WSN Lab Report Group 2

1st Chi Xia
Chair of Communication Networks
Technical University of Munich
Munich, Germany
go25kuw@tum.de

2nd Kaikang Huang
Chair of Communication Networks
Technical University of Munich
Munich, Germany
go49jan@mytum.de

3rd Zhihao Deng
Chair of Communication Networks
Technical University of Munich
Munich, Germany
zhihao.deng@tum.de

***Abstract*—Urban underground parking often suffers from inefficient space utilization, causing unnecessary congestion and time waste for drivers searching for available parking spots. This project proposes a smart parking system based on a wireless sensor network (WSN) to detect parking space availability in real-time. By deploying multiple sensor nodes equipped with light and distance sensors on individual parking slots, the system accurately detects vehicle presence through sensor data fusion. The sensor network adopts a cluster-based multi-hop topology, employing a routing protocol optimized for energy-efficient and event-driven communication. Cluster head nodes are dynamically selected based on application-specific metrics, enhancing the scalability and efficiency of the network. A graphical user interface (GUI) developed using Qt provides real-time visualization of parking slot status and network topology. This intelligent system reduces parking search time, optimizes space utilization, and demonstrates adaptive and energy-efficient WSN operations in dynamic environments.**

***Index Terms*—Wireless sensor networks, Smart parking, Cluster-based routing, Energy efficiency, Multi-hop communication, Real-time monitoring, Sensor data filtering.**

I. Introduction

The increasing density of urban areas poses significant challenges for parking infrastructure, particularly in underground facilities where limited visibility and space availability complicate efficient vehicle management. Drivers often spend excessive time searching for vacant spaces, contributing to unnecessary traffic within parking structures and negatively impacting overall traffic flow.

To address these inefficiencies, modern parking solutions are progressively integrating sensing and wireless communication technologies. Wireless sensor networks offer a promising approach by providing decentralized, scalable systems capable of monitoring environmental conditions and reporting data in real time. By equipping individual parking slots with embedded sensor nodes, the occupancy status of each space can be monitored continuously and relayed across the network.

This project develops a smart parking system leveraging a multi-hop WSN to enable distributed vehicle detection and data forwarding. The system design focuses on scalability, energy efficiency, and real-time responsiveness, ensuring practical deployment in constrained underground environments. In addition, a graphical user interface is implemented to provide users and system administrators with an intuitive visualization of parking slot availability and network status.

The following sections present the system architecture, sensor hardware, communication protocols, and interface design that together enable this intelligent parking solution.

II. Hardware System and Sensors

The hardware platform for the wireless sensor network consists of multiple **nRF52840 Development Kits (DK)** from Nordic Semiconductor. Each board integrates an IEEE 802.15.4 compliant wireless transceiver and a microcontroller based on the ARM Cortex-M4F architecture, providing sufficient computational capability and low-power wireless communication for embedded sensing applications.

In the presented system, a total of nine nRF52840 DK boards are utilized to form the complete network. One board is configured as the master node, which is directly connected to a PC via serial interface and serves as the central coordinator of the network. Three nodes are dynamically selected as cluster heads, responsible for forwarding data from other nodes and enabling multi-hop communication within the network. The remaining five boards operate as sensor nodes, deployed at individual parking slots to monitor vehicle presence through onboard sensors. These sensor nodes focus solely on data collection and transmit their readings through the cluster heads to the master node.

Each sensor node is equipped with two types of sensors to accurately detect the presence of a vehicle in a parking slot:

- **Light Sensor: Phidgets 1000 Lux Sensor** The light sensor measures ambient light intensity within a range of 1 to 1000 lux and produces a linear analog voltage output proportional to the detected illuminance. In this application, the sensor is mounted on the floor of each parking slot to detect changes in ambient light caused by the presence of a vehicle. When a car is parked directly above the sensor, the measured light intensity decreases significantly due to shadowing effects. A continuously low light sensor reading sustained over a certain period indicates a high probability that the parking slot is occupied. This approach enables a non-intrusive and contactless detection method for assessing parking slot availability.
- **Distance Sensor: MaxBotix LV-MaxSonar-EZ1 Sonar Sensor** The ultrasonic distance sensor measures the distance to objects directly above the sensor. The LV-MaxSonar-EZ1 offers a typical sensing range from 15

cm to 645 cm with centimeter-level resolution. In this application, the distance sensor detects the vertical clearance above the parking slot. A significant decrease in measured distance indicates that a vehicle is positioned above the sensor, complementing the light sensor reading for occupancy detection.

The combination of the light sensor and the distance sensor allows for robust detection of parking slot occupancy, minimizing false positives caused by temporary lighting changes or environmental noise. The sensor readings are periodically collected by the nRF52840 DK board and transmitted over the wireless network to the master node for aggregation and visualization.

III. Network Architecture and Communication Protocol

The wireless sensor network adopts a hierarchical, cluster-based topology to enable scalable and energy-efficient communication. A customized routing protocol manages route discovery and data forwarding across multiple node types, supporting reliable multi-hop communication within the parking system.

A. RIP based Protocol

Our protocol follows a RIP-based design, where each node maintains a routing table of neighborhoods containing distance metrics to known destinations. The proposed system classifies nodes into three functional types to support a hierarchical and energy-efficient network architecture:The network consists of three node types: (1) Master Node – acts as the control plane, initiating routing discovery, managing cluster head (CH) selection, collecting topology via DAO packets, and interfacing with GUI; (2) Cluster Head (CH) – serves as a gateway, aggregating data from worker nodes and forwarding it to the master via multi-hop paths; (3) Worker Node – low-power sensing units deployed to monitor parking slots and report data to their assigned CHs.

The protocol could be classified into two main phases: Routing Discovery and Stable Phase.

1) Phase 1: Routing Discovery: Each node initializes its local routing table with only its own address as a valid entry. Memory pools and list structures for routing entries are also initialized. The master node periodically broadcasts a DIO (DODAG Information Object) packet containing its own address, a sequence number, and an initial hop count.

Upon receiving a DIO packet, a worker node performs the following operations:

- Records the source address, hop count, and RSSI into its local routing table.
- Forwards the DIO packet to neighboring nodes with an incremented hop count and updated source address.
- Sends a DAO (Destination Advertisement Object) packet upstream to the master, reporting its routing information.

The master node collects DAO packets from across the network to build a complete topology view and initialize the adjacency matrix for the global routing structure.

2) Phase 2: Stable Phase: After routing discovery, the master node maintains a global view of the network topology and performs cluster head selection based on metrics such as node degree, energy, and connectivity. The master then broadcasts the cluster head assignments to the network, thereby forming a hierarchical architecture.

In this stable phase:

- Worker nodes send DAO packets to their assigned cluster heads.
- Cluster heads aggregate and periodically forward DAO packets to the master node via multi-hop routing paths.
- Each DIO and DAO packet includes a sequence ID; the master monitors the sequence updates to detect node activity.

If a new node attempts to join the network while the system is operating in a stable state, it first listens to the communication channel for 3 seconds to detect any existing network activity. After this listening phase, the node actively broadcasts a DIO message to announce its presence and initiate the joining process. Upon receiving this message, one of the existing nodes in the network forwards the packet to the master node according to the current routing table. The master node then re-executes the cluster head selection process and broadcasts an updated routing table to the entire network, ensuring that the topology reflects the addition of the new node.

B. Routing Algorithm

The core concept of this project is a cluster-based scheme. This type of algorithm typically divides the entire process into two stages: head selection and subnode allocation. Based on the collected information, the routing table is then updated.

1) Head Chosen: The first step is to obtain a stable adjacency matrix, which represents the connectivity among nodes. Based on this matrix, a hop-n matrix is calculated. In this project, due to the limited number of nodes, only hop-1, hop-2, and hop-3 connections are considered during the head selection process.

In addition to the adjacency matrix, other criteria—such as each node's battery status and its average received RSSI—are also taken into account. To achieve energy efficiency, the total or average number of hops across the network should be minimized. Therefore, nodes that can reach others with fewer hops are considered more valuable.

Since the system is battery-powered, it is preferable to concentrate energy consumption on the head nodes. This criterion also plays a critical role in the selection process. Ultimately, the chosen head nodes must satisfy all these requirements.

- Can connect to as many as nodes using as few as hops.
- Battery state should be healthy.
- Can connect to master node with help of as few as subnodes.

As the system continuously monitors both the status of each node and the adjacency matrix, it can dynamically reselect head nodes to ensure a more balanced energy consumption, thereby extending the overall system lifetime.

2) Sub-node Allocation: After selecting the head nodes, the next challenge is how to allocate the remaining nodes to each head. The simplest case involves nodes that can directly connect to a head. However, the algorithm must also ensure fairness during this allocation process. To achieve this, it considers both the battery level of each head and the number of subnodes already assigned to it. This balanced allocation strategy enhances the overall robustness of the system.

Some nodes may be unable to connect directly to any head. In such cases, a node will connect to an already assigned subnode, based on the battery condition of that subnode. Since the system is continuously maintained, the allocation may be updated if a subnode's battery status changes.

3) Reactive Network: The network is designed to be reactive, meaning that a node transmits sensor data only when a sudden change in the recorded values occurs. Given the target scenario, the data typically remains within an acceptable range during most of the system's active time. Therefore, transmitting unremarkable or stable data is unnecessary, which helps conserve energy and prolong the system's operational lifespan.

IV. Graphical User Interface Implementation

The graphical user interface (GUI) is developed using Qt and serves as the primary interaction platform for system administrators. It establishes a serial connection to the master node, which functions as the central coordinator of the wireless sensor network. By opening the COM port of the master node, the GUI continuously listens for incoming messages. These messages, generated by the master node, contain various types of information essential for real-time monitoring and visualization of the network.

The master node transmits four distinct categories of messages over the serial connection:

1) **Sensor data messages:** These messages contain the node ID, sensor type, sensor reading, and the current battery level of the node. An example message follows the format `Node: 1 SensorType: 1 Value: 12 Battery: 80`, where sensor type 1 refers to the light sensor and sensor type 2 to the distance sensor. Upon receiving such a message, the GUI evaluates the occupancy status of the corresponding parking slot by comparing the sensor readings against predefined thresholds. If the slot is determined to be available, the graphical representation of the parking slot is set to green; otherwise, if a car is detected, the slot is marked in red. Additionally, the current battery level of the node is displayed as a percentage in the interface.
2) **Topology update messages – new links:** These messages notify the GUI of new links within the network topology, for example, `NewLink 1 -> 2`. Upon receiving this message, the GUI dynamically adds a directed edge from the source node to the destination node in the topology graph, reflecting the latest routing structure maintained by the master node.
3) **Topology update messages – node failure:** If a node becomes unresponsive for a certain period, the master node detects the failure and sends a message such as `LinkLost: 1`. The GUI then removes the corresponding node from the topology graph, along with all associated links where the node served as either the source or destination.
4) **Cluster head assignment messages:** The master node periodically evaluates network energy conditions and assigns new cluster heads by transmitting a message such as `ClusterHead: 2 3 6`. The GUI processes this message by classifying the listed nodes as cluster heads, placing them on the second stage of the topology graph, and visually distinguishing them from normal nodes. All remaining sensor nodes, excluding the master node, are positioned in the third stage of the graph and assigned the class "normal".

The network topology is visualized in a hierarchical three-stage structure. The first stage contains the master node, fixed as node 0. The second stage displays the dynamically assigned cluster head nodes, and the third stage contains all other sensor nodes. Whenever topology-related messages are received, the GUI adapts the node positions and re-renders the topology graph to reflect the current network state.

The GUI layout is organized into two main display areas:

- On the **left side**, the latest sensor values for each parking slot are displayed, providing users with a clear overview of parking space availability. Additionally, a message log window in the center of the interface continuously displays all incoming messages transmitted by the master node, facilitating real-time observation of system events. An example of this layout is shown in Fig. 1.
- On the **right side**, the **network topology graph** is rendered, showing the nodes and their interconnections. This visualization automatically updates in response to network events such as the addition of new links, node failures, and changes in cluster head assignments. The initial layout of the topology graph adopts some of the graphical rendering concepts from the Elastic Nodes example in the Qt framework [3], which demonstrates dynamic node placement and edge drawing in a scalable and interactive manner. An example of the topology visualization is shown in Fig. 2.

Beyond monitoring, the GUI also supports control functionalities. Specifically, it allows users to adjust the transmitting power of the master node. This feature enables optimization of communication range and energy consumption during system operation.

For development and debugging purposes, the GUI employs extensive `qDebug` logging. During message processing, the GUI prints detailed debug information in the Qt Creator debug window, including the number of words contained in the received message and the specific content of each word. This facilitates efficient debugging and validation of message parsing and system logic.

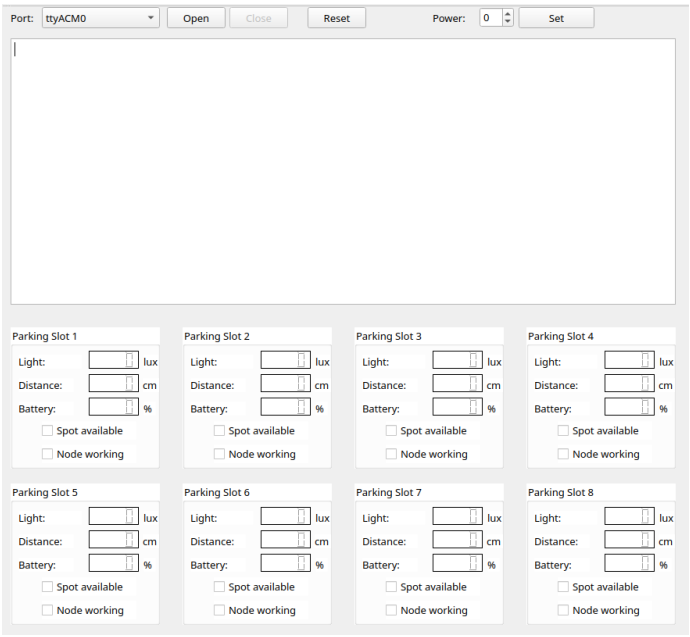

Fig. 1. The left panel of the GUI displaying parking slot sensor values and the message log window during initialization.

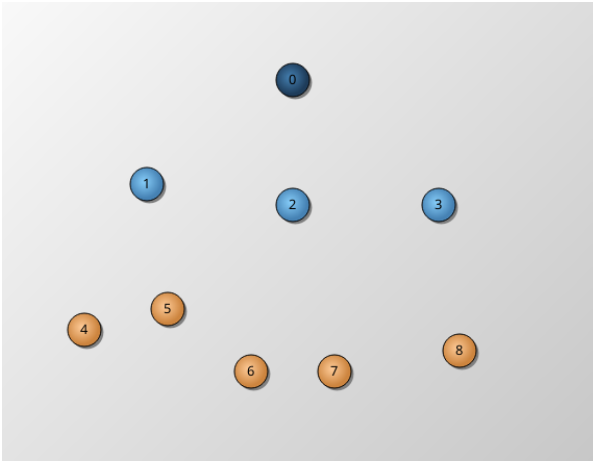

Fig. 2. The right panel of the GUI visualizing the dynamic network topology.

V. Application-Specific Enhancements

In addition to the basic wireless communication and sensing functionalities, the system implements several application-specific enhancements tailored to the smart parking scenario. These enhancements optimize the network's routing performance and improve the robustness of the parking slot occupancy detection.

A. Low Power Protocol Design

Energy efficiency is a fundamental design goal in our smart parking system. The proposed routing protocol and network architecture incorporate two energy-aware mechanisms.

First, during the routing discovery and topology construction phases, the master node evaluates the residual battery level of each node and uses this information as a key metric in the selection of CHs. Nodes with higher energy reserves are preferentially assigned to serve as CHs, taking on additional responsibilities such as data aggregation and multi-hop forwarding. This strategy ensures that energy-intensive tasks are offloaded from weaker nodes, promoting balanced energy consumption across the network.

Second, the use of a hierarchical architecture significantly reduces overall communication overhead. Sensor nodes only communicate with their designated CHs rather than sending data directly to the master node. The CHs locally integrate and forward data in batches via multi-hop routing paths. This reduces the number of transmissions over long distances, mitigates congestion in high-density deployments, and lowers per-node energy usage.

Overall, the combination of energy-aware cluster head assignment, hierarchical data aggregation enables the system to operate reliably under strict energy constraints. These design choices make the protocol highly suitable for long-term deployment in real-world smart parking scenarios.

B. Sensor Data Filtering in the GUI

To improve the stability of the parking slot occupancy indication, a filtering mechanism is applied within the GUI. The raw sensor data transmitted from the master node may exhibit transient fluctuations caused by environmental changes or sensor noise. Without filtering, these fluctuations could lead to rapid and misleading changes in the displayed parking slot status.

To address this, the GUI processes the received sensor values and applies a time-based smoothing filter. The system updates the parking slot status only when the sensor readings consistently indicate a state change over a predefined duration of 5 seconds. This filtering prevents frequent toggling between occupied and available states, ensuring that the GUI reflects the actual occupancy status more reliably, and enhancing the overall user experience.

VI. Conclusion

This project presents the design and implementation of a smart parking system based on a wireless sensor network. By integrating light and distance sensors with nRF52840 DK boards, the system reliably detects the occupancy status of individual parking slots. A multi-hop network architecture is established using the RIP based protocol and connectivity-aware LEACH algorithm, enabling efficient data forwarding and cluster-based communication. A Qt-based graphical user interface provides real-time visualization of parking slot availability and network topology, applying application-specific data filtering to enhance system stability. The project demonstrates a scalable and adaptable approach to smart parking management, highlighting the potential of WSN technologies in addressing practical urban infrastructure challenges.

REFERENCES

[1] N. G. Palan, B. V. Barbadekar, and S. Patil, “Low energy adaptive clustering hierarchy (LEACH) protocol: A retrospective analysis,” in *Proc. 2017 Int. Conf. Inventive Systems and Control (ICISC)*, 2017, pp. 1–12, doi: 10.1109/ICISC.2017.8068715.

[2] W. B. Heinzelman, A. Chandrakasan, and H. Balakrishnan, “An application-specific protocol architecture for wireless microsensor networks,” *Ad Hoc Networks*, vol. 1, no. 4, pp. 493–502, 2002. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1084804512002482

[3] The Qt Company, ”Elastic Nodes Example - Qt Widgets,” Qt Documentation, 2024. [Online]. Available: https://doc.qt.io/qt-6/qtwidgets-graphicsview-elasticnodes-example.html. [Accessed: July 8, 2025].