AUTOMATIC STREET LIGHT FAULT DETECTION SYSTEM USING LIGHT DEPENDENT RESISTOR

KARTHIKEYAN T[1], ARUN R G[2], BALAJI S S[3], DEEPESH KUMAR S[4], GOKUL HARI R[5]
[1,2,3,4,5] *Department of Computer Science and Engineering, Knowledge Institute of Technology, India*
Email: {tkcse[1], 2k20cse006[2], 2k20cse015[3], 2k20cse027[4], 2k20cse048[5]}@kiot.ac.in

Abstract

Downtime in streetlights could pose a significant threat during nighttime, and current reliance on public reports or periodic surveys done by technicians in periodic intervals to report such anomalies is time-consuming and inefficient. This paper suggests the usage of modern IoT solution to efficiently report the faults as they appear. This solution utilizes a Light Dependent Resistor (LDR) sensor integrated into an inexpensive board such as Arduino UNO connected to a Modular Network Module to detect changes in light intensity or the absence of light instantly. The collected information, including a unique identification number for each street light pole, is then transmitted to a regional hub for precise location determination and technician dispatch and also relayed to the nearest Electricity Board (EB) Office mainframe. The system can scale beyond just fault detection and can act as a backbone for expanded infrastructure, providing a versatile framework for future enhancements. The expected outcomes include faster response times, reduced costs, and overall improved urban lighting infrastructure.

Keywords: IoT, Urban Lighting Infrastructure, Real-time Fault Detection, LDR, Modular Network Module

1. INTRODUCTION

In modern urban environments, the efficient operation and maintenance of street lighting infrastructure are critical for ensuring public safety and enhancing overall urban livability. However, the traditional methods of detecting and addressing faults in streetlights present significant challenges, often resulting in delays, inefficiencies, and increased costs. Typically, the information regarding damaged or faulty streetlights is relayed to the Electricity Board (EB) office and subsequently to technicians through two primary channels: public reports or periodic surveys conducted by technicians [1].

Unfortunately, both approaches suffer from inherent drawbacks. Public reports, while potentially numerous, are often unreliable and difficult to prioritize among the vast number of reports received. On the other hand, periodic surveys conducted by technicians are time-consuming and inefficient, often taking days or even months to complete due to the sheer scale of urban environments. This inefficiency is exacerbated by the absence of viable alternatives, leaving municipalities with few options for improving the maintenance pipeline.

Recognizing the need for optimization, this paper proposes a novel solution leveraging wide-scale Internet of Things (IoT) technologies to revolutionize the detection and reporting of faults in street lighting infrastructure. The proposed solution centres around the utilization of Light Dependent Resistor (LDR) sensors, which detect reductions in light intensity or the absence of light. These sensors are seamlessly integrated into an inexpensive board, which transmits real-time information through a Modular Network Module utilizing GSM or other appropriate networking technologies to a regional hub covering a radius of several kilometers.

Critical to the effectiveness of the proposed solution is the transmission of comprehensive information, including unique identification numbers assigned to each streetlight and relevant telemetry data from the pole. The regional hub acts as a central relay point, forwarding this information to the nearest EB Office mainframe. Here, the unique identification numbers are resolved to determine the precise location of the faulty streetlight, enabling expedited dispatch or alerting of technicians for prompt repair.

By combining advanced sensor technologies with robust communication networks, this paper aims to streamline fault detection and reporting processes, significantly reducing response times, and improving overall urban lighting infrastructure. The integration of IoT technologies not only addresses the shortcomings of existing maintenance practices but also lays the groundwork for future enhancements and innovations in urban lighting management.

2. LITERATURE REVIEW

Numerous studies have been carried out in the field of Urban infrastructure and its modernization using IoT. We will discuss the relevant works below, which have inspired this paper.

Study by Dizon and Pranggono [2] discussed the use of smart street lights in the city of Sheffield to emphasise and enhance the public safety and reduce the power consumption of the urban lighting system which at that point have been predominantly controlled by static management system. Crucially they concluded that the dimming schemes that are being implemented in the city did save energy up to 50% compared to the conventional scheme.

An experimental study conducted by Chalfin et al. (2022) [3] in the city of New York found that there is correlation between crime rate and availability of street lights during nigh-time, this is mainly due to increased visibility provided by the street lights at night times which is the major hotbed for crimes around the world. This experiment proves that the streetlights that are operational act as major deterrent and maintaining it and reducing downtime must be a high priority. Hence, using an automated fault system extends beyond just increasing speed of repair but also increases the public safety significantly.

Study work done by M. Kanthi and Ravilla, D [4] suggests a system that uses LDR and LIR to detect presence of on road entities and turn on the streetlights accordingly. The study concluded with the results of power savings of the proposed system being up to 53.45%, 44.76%, 39.39%, and 32.25% respectively for 10%, 20%, 30%, and 50%. The communication they use is a nRF24L01 transceiver module, and the system is configured in a master node and slave node fashion. This method inspired the use of regional hub and all the lights (acting as a slave node) reporting to the regional hub. As such, very little computation is performed on the light nodes. Therefore, we can optimize the node computation devices to their least required level to reduce cost and energy.

The project work done by Ashok Kumar et al. (2020) [5] proposed a system that also use LDRs to detect the faults in the streetlights. It should be noted that this project is a major inspiration for our system. The system makes intelligent use of environmental light to detect the conditions of the day-night cycle. This component inspired the use of environmental LDR in our system. The system they proposed also used GPS and cloud for location tracking and storage, which we believe is quite excess for this type of application. So, we used id for location tracking and local database for storage in our system. They also have a common controller to which all streetlight in an small area would connect to physically, this can be quite difficult on implementing in large scale, so we opted for Independently functioning modules that are installed separately on each light pole.

Sravani et al. (2021) [6] suggested the use of LoRa and IEEE 802.11 wireless modules to be used for a similar type of system which they proposed in their study work. We found this type of communication would be beneficial on most of the cases.

From the review work, we can deduce that a system of this application must be adaptable such that it can work regardless of communication medium used, loosely coupled to other streetlights and each unit should do the minimum amount of work possible, such that it can be optimized in terms of hardware requirements.

3. PROPOSED WORK

To address all the advancements and shortcomings of previous works, we propose a system that utilizes the central hub reporting, environmental sensors and units decoupled from network modules to fully take advantage of the solutions proposed.

The system mainly consists of three components: Light pole unit which reads out the sensor output and compresses them out to a 32-bit integer and transmits them through the installed network module using MQTT Protocol, An MQTT broker which in our case is HiveMQ and the regional hub.

Each light pole would be assigned an ID statically to eliminate the need of GPS for location tracking as we can easily map an static object to an location in our database.

In our implementation we have used GSM as the networking module since it was most convenient with respect to our current implementation requirements. We acknowledge that the use of GSM module is excess and an alternative module such as LoRa module or NB-IoT module depending on the environment it should operate on. Using the later modules would facilitate the use of daisy-chained communication leading to much lesser signal noise bleeding into public reception. Nevertheless, our choice of the GSM module adheres to our guiding principle of selecting the most suitable network module for a given environmental category.

The broker we used HiveMQ public broker endpoint [7] to illustrate the working of the project, however, an local endpoint using open source MQTT implementations like Eclipse Mosquito [8] can be used easily to eliminate outside dependencies.

The Overall Flow of the system would look like flow chart shown in Figure 1.

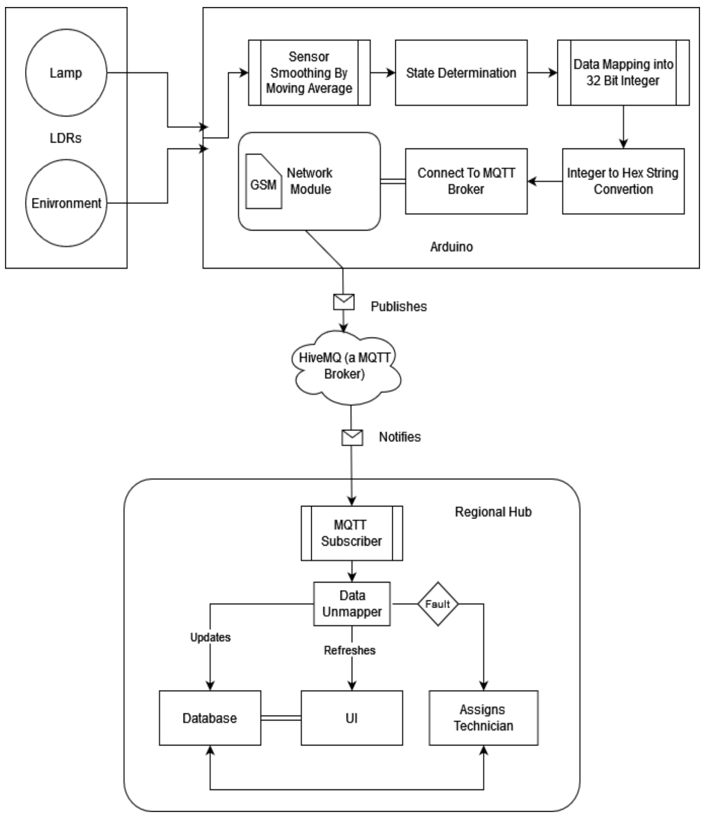

Fig.1. System Flow Chart

Below we will discuss major components present in the system and how they function with respect to implementation of our system.

3.1 LDR

There are two LDR Sensors, Lamp LDR which is placed such that it faces the bulb of the light pole, and the environment LDR which is exposed to the sky and placed such that it is isolated from the Lamp LDR. The two LDR Sensors are simultaneously read with minimal delay such as 5ms. This process is repeated for atleast 20 times for each main loop and the average of the readings is considered for further processing. This type of smoothening is called as rolling average and is done to eliminate fluctuations in the sensors, this smoothening algorithm is very simple and is good enough for this application [9]. The Environment LDR gives the context like night, dusk, dawn and day, and Lamp LDR must be greater than environment during night and dusk.

3.2 ARDUINO

In our system we have used Arduino UNO as our computational device, since it is inexpensive and the program can be ported to other Microcontroller board quite easily [10]. As previously discussed the board reads the values, eliminates the fluctuations and determines the state using a predicate. But, it also performs additional operations to efficiently transmit the data to the regional hub.

Below we will discuss how the board transmits the data to the MQTT broker in detail.

3.2.1 Data Masking:

Data Masking is done to pack as much data as possible within a unit of storage, therefore this step is more akin to data packing or data compression, the masking part is the one that achieves it, hence the name. The MCU would publish four pieces of data to the broker server: ID of the light pole, Lamp LDR value, Environment LDR value and finally the state of the lamp. The regional hub wouldn't manage more than 1000 light poles within its territory; therefore, the ID of the light pole can be represented in a 10-bit number instead of 4 bytes. The ADC in Arduino board would only resolve up to 10-bit (0-1023) so the values of both LDR is essentially capped at 0-1023 and finally the state/condition of the lamp would only have three states: working, sleeping and fault, which can be represented in 2-bit number. Adding all the space of the data, we get 10+10+10+2 = 32 bits, which is an unsigned long. The data are bit-shifted and masked before payload publishing. The Figure 2 will show the structure of the masked long in binary form.

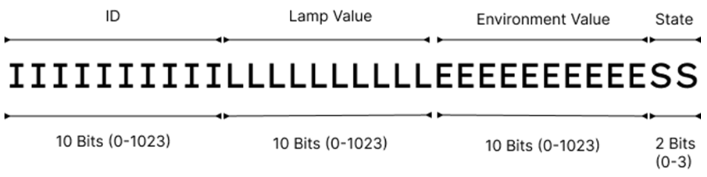

Fig.2. Bit Masked 32-Bit Integer

The masked integer is then converted to its Hexadecimal format, which is then converted to byte string reducing the number of bytes required to transmit data further.

3.2.2 Connecting to MQTT Broker:

For every predefined period of time such as 5 seconds or more, the unit will try to connect to MQTT broker to publish the hex string data on a predefined topic. The Topic would be used by all the light pole units to publish, and every region has their topic which would be subscribed by their respective regional hub. The publish is deliberately left as a tuneable variable to reduce and assign each region's level of granularity of live updates.

3.2.3 Sending Data Through Networking Module:

As previously stated, we have used a GSM Module called SIM800L for transmitting the data. However, this is not set in stone and can be easily swapped out for other efficient modules with little to no code change.

3.3 MQTT BROKER

Functioning as a communication intermediary, the MQTT Broker serves as a centralized hub for message exchange between the sensor module and downstream recipients. This middleware component receives published messages from the sensor module and facilitates seamless dissemination to subscribed clients which in this case: the regional hub. We used HiveMQ Public Broker endpoint for our implementation, but the brokers such as Eclipse Mosquito can be run locally in the regional hub to eliminate external dependencies and to improve the efficiency of data processing.

3.4 REGIONAL HUB

The regional hub can be any computational device that can support running python programs and in case of local running of MQTT brokers, meets the requirements to running said software as well. This is quite simple requirements which can be accomplished even by Raspberry Pi 5 [11] systems. However, for our implementation we have used a laptop as regional hub. The regional hub does the following tasks.

3.4.1 MQTT Subscriber:

The regional hub runs a separate thread of MQTT subscriber program that listens for notifications of the published messages, then when an message is received it runs an task for that particular message in an separate thread.

3.4.2 Data Unmasker:

The data unmasker decompressed the data stored in the binary hex string and splits out the individual elements: Unique ID, Lamp LDR Value, Environment LDR value and State of the light pole. Then it updates the database and the GUI according to the message.

3.4.3 Database:

The system employs MySQL database for this purpose, which houses data pertaining to the streetlights, including the mapping of their unique identifiers to their respective locations. It also maintains a record of registered technicians and repair tasks, thereby facilitating efficient task assignment and tracking.

In addition to this, the MySQL database also serves as a time-series database, storing sensor data published by the streetlights. This allows for historical data analysis and trend identification, which can be instrumental in predictive maintenance and resource planning.

The choice of MySQL for this project was primarily driven by familiarity rather than efficiency. While MySQL is a robust and widely used database system, it may not be the most efficient choice for time-series data, especially when the system needs to handle multiple write operations concurrently. MySQL also lack robust support for data aggregation, which is often required when dealing with time-series data.

Given these limitations, further improvement would be to considering alternatives that are better suited for handling time-series data. PostgreSQL, when used in conjunction with its time-series database extension, could offer improved performance and more powerful data aggregation capabilities. Other dedicated time-series databases like InfluxDB could also be considered, given their optimized architecture for handling time-series data.

Another suggested approach is to use MongoDB as a collector for sensor data, while periodically updating the MySQL tables. This hybrid approach could leverage the strengths of both systems, with MongoDB handling the high-volume sensor data and MySQL managing the less frequently updated data.

3.4.4 Desktop UI:

The Desktop UI displays the information of the individual light poles such as their state and their location. The locations will be displayed in an Map situated in the left side of the UI and states are colour coded to distinguish them easily.

4. IMPLEMENTATION

The implementation of the system would require the device or components listed in Table 1.

Table.1. Device Requirements

Devices	Quantity
Arduino UNO	1
Bread Board Large	1
SIM800L	1
Basic LED	2
LDR Sensor	2
Resistors 1K Ohms	10
Resistors 10K Ohms	10
Jumper Cables	20
LM2596 Convertor	1
LCD RG1602A Display	1

The connections are to be made according to the schematic displayed in the Figure 3.

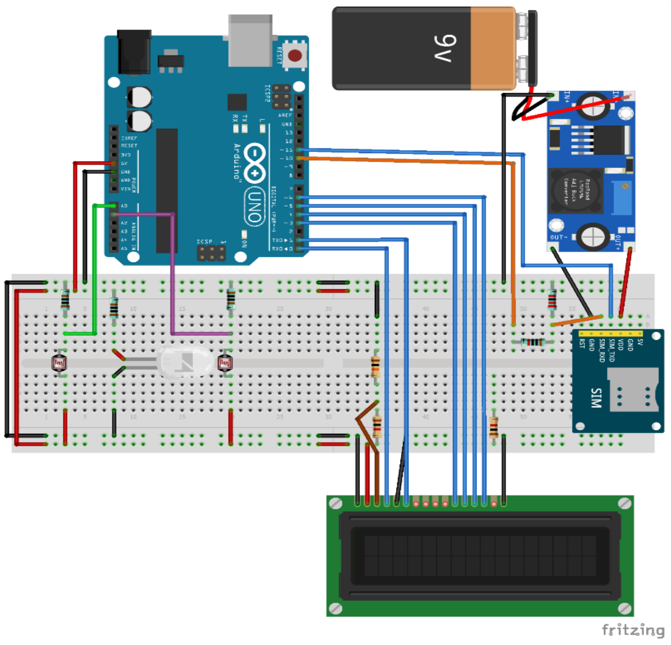

Fig.3. System Schematic

The LED is used for mimicking the Lamp in an streetlight. Since the LDR are connected in voltage divider circuit the resistance of the second resistor must be one magnitude higher than the LDR ohm range to get a readable voltage that makes sense during night and daytime [12]. Given the SIM800L module’s requirement for a substantial current supply, specifically two amperes, it is imperative to provide a stable and sufficient power source to ensure its optimal operation. This consideration is essential to prevent any power-related disruptions in the module’s functionality. An LCD display also connected to display the information of each LDR sensor’s intensity values and transmission information. The Final assembly would like Figure 4.

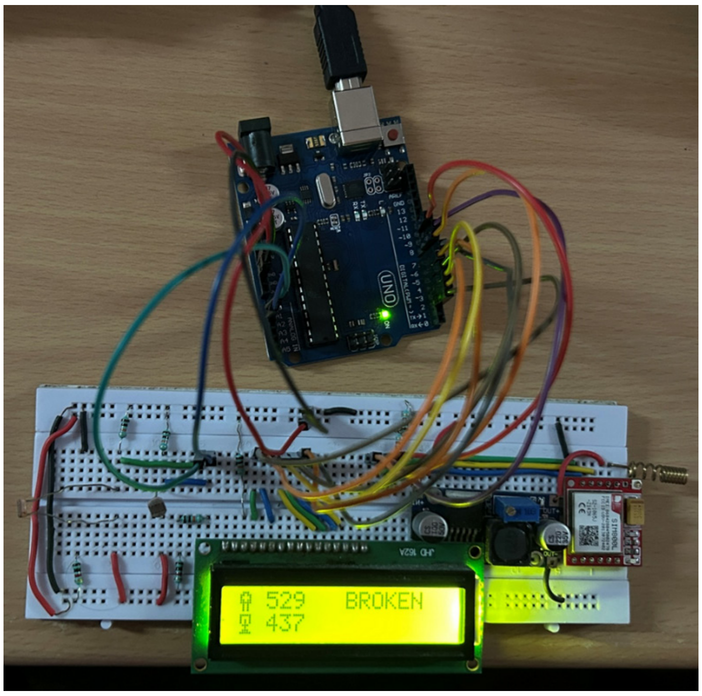

Fig.4. Physical Implementation

The GUI interface in the regional hub is created using CustomTKinter, which library wrapper for Tkinter library in python. The GUI features an Map displaying all the location of the managed sensors and an list of light poles along with their state in color coded information.

5. RESULT

The Figure 4 shows the result of how the unit would act when run and the Figure 5 shows the regional hub GUI result during the execution.

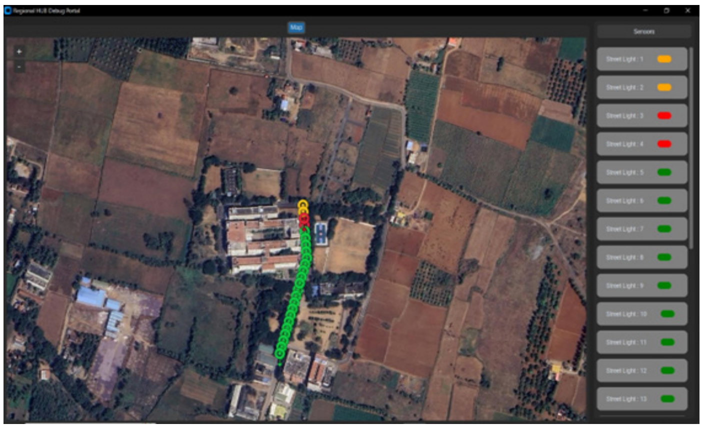

Fig.5. Desktop GUI

The map on the left side displays the location information along with a colour coded pin on their location. Each Sensors are listed in the sensor list situated on the right side of the GUI. Each sensor listing is clickable and when on click will go to pin of the sensor.

6. CONCLUSION

In conclusion, the proposed streetlight monitoring system addresses the challenges outlined in the problem statement by leveraging modern IoT technologies to enhance urban lighting infrastructure. Through the implementation of the system described in the proposal, we have provided a solution that enables real-time detection of streetlight faults, precise location tracking, and efficient maintenance processes. By utilizing Light Dependent Resistor (LDR) sensors and modular network modules, we have introduced a cost-effective and scalable approach to streetlight management. This system not only optimizes the process of reporting faults but also lays the foundation for future enhancements and innovations. With its ability to improve response times, reduce costs, and enhance overall urban lighting infrastructure, the proposed solution offers a transformative approach to streetlight maintenance in both urban and rural areas.

REFERENCES

[1] Intelilight, "Automatic Street Lighting," [Online]. Available: https://intelilight.eu/automatic-street-lighting/.

[2] E. Dizon and B. Pranggono, "Smart streetlights in Smart City: a case study of Sheffield," *Journal of Ambient Intelligence and Humanized Computing,* 2022.

[3] A. Chalfin, B. Hansen, J. Lerner and L. Parker, "Reducing Crime Through Environmental Design: Evidence from a Randomized Experiment of Street Lighting in New York City," *Journal of Quantitative Criminology,* vol. 38, no. 1, March 2022.

[4] M. D. R. Kanthi, "Smart streetlight system using mobile applications: secured fault detection and diagnosis with optimal powers," 2023.

[5] A. K. Nanduri, S. K. Kotamraju, G. L. Sravanthi, S. R. Babu and K. V. K. V. L. P. Kumar, "IoT based Automatic Damaged Street Light Fault Detection Management System," *International Journal of Advanced Computer Science and Applications(IJACSA),* 2020.

[6] N. Sravani and Y. L. a. G. Nirmala, "Street Light Controlling and Monitoring of Fault Detection using LoRa," *International Journal for Modern Trends in Science and Technology,* 2021.

[7] HiveMQ, "HiveMQ Public MQTT Broker," [Online]. Available: https://www.hivemq.com/mqtt/public-mqtt-broker/.

[8] Eclipse Foundation, "Eclipse Mosquito," [Online]. Available: https://mosquitto.org/.

[9] T. O'Haver, "Smoothing," [Online]. Available: http://terpconnect.umd.edu/~toh/spectrum/Smoothing.html.

[10] "Arduino Uno Rev3," [Online]. Available: https://www.arduino.cc/en/Main/ArduinoBoardUno.

[11] Raspberry Pi, "Raspberry Pi 5," [Online]. Available: https://www.raspberrypi.com/products/raspberry-pi-5/.

[12] A. Meyer, "Simple Light Reading With LDR + Arduino," [Online]. Available: https://adam-meyer.com/arduino/LDR.