Industrial Inspection & Maintenance Control Robot

A PROJECT REPORT

Submitted by

Ranjan Mishra (231B258)
Rishabh Jain (231B264)
Rashmi Dubey (231B260)

Under the guidance of: Dr. Amit Kumar Srivastava

November-2025

Submitted in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY
IN
COMPUTER SCIENCE & ENGINEERING

Department of Computer Science & Engineering
JAYPEE UNIVERSITY OF ENGINEERING & TECHNOLOGY,
AB ROAD, RAGHOGARH, DT. GUNA-473226 MP, INDIA

JAYPEE UNIVERSITY OF ENGINEERING & TECHNOLOGY
Accredited with Grade-A+ by NAAC & Approved U/S 2(f) of the UGC Act, 1956
A.B. Road, Raghogarh, District Guna (MP), India, Pin-473226
Phone: 07544 267051, 267310-14,
Fax: 07544 267011
Website: www.juet.ac.in

CERTIFICATE

This is to certify that the work titled "Industrial Inspection and Maintenance Control Robot" submitted by Ranjan Mishra (231B258), Rishabh Jain (231B264), and Rashmi Dubey (231B260) in partial fulfilment for the award of the degree of B.Tech of Jaypee University of Engineering & Technology, Guna has been carried out under my supervision. To the best of my knowledge and belief, there is no infringement of intellectual property rights or copyright. This work has not been submitted, either partially or wholly, to any other University or Institute for the award of this or any other degree or diploma. In the event of any violation, the concerned student will be solely responsible.

Signature of Supervisor (Dr. Amit Kumar Srivastava)

Department of Computer Science and Engineering

Jaypee University of Engineering and Technology

Guna, M.P., India

Date: 19/11/2025

DECLARATION

We declare that the work reported in the B.Tech 5th semester project entitled “Industrial Inspection and Maintenance Control Robot”, submitted in partial fulfillment for the award of the B.Tech (CSE) degree at Jaypee University of Engineering and Technology, Guna, is my original work. To the best of my knowledge and belief, it does not infringe upon any intellectual property rights or copyrights. In case of any violation, I will be solely responsible.

Ranjan Mishra (231B258)

Rishabh Jain (231B264)

Rashmi Dubey (231B260)

Department of Computer Science and Engineering

Jaypee University of Engineering and Technology

Guna, M.P., India

Date: 19/11/2025

ACKNOWLEDGEMENT

We have invested a lot of time and efforts in completion of this project, but this project would not have been possible without the kind support of various individuals and organizations. We here by extend our sincerest gratitude to all of them. We are highly indebted to our mentor, **Dr. Amit Kumar Shrivastava.** We would like to thank him for his constant guidance and supervision. We would like to thank the JUET for providing us with such and supportive and innovative environment and facilities and lending us their support whenever required. We would also like to thank the whole organization for providing us with such opportunities.

Finally, we all would like to thank and congratulate our fellow project members on their hard work and dedication.

Thanking you

Ranjan Mishra (231B258)

Rishabh Jain (231B264)

Rashmi Dubey (231B260)

Date: 19/11/2025

EXECUTIVE SUMMARY

This project presents a fully integrated Industrial Inspection and Maintenance Control Robot, designed to enhance safety, efficiency, and automation in industrial environments. Traditional inspection methods rely heavily on manual labour, which can be slow, risky, and error prone. To counter this, the system reframes inspection through a simulation-driven robotic workflow built entirely inside ROS 1, using Gazebo for physics-based world modelling and RViz for real-time visualization.

Over 14 weeks, the project grew from foundational robotic control concepts into a complete virtual inspection ecosystem. A custom Gazebo world was designed to emulate an industrial site, and the full robot model was simulated with LiDAR, ultrasonic sensing, thermal data streams, and gas-detection logic. Core navigation modules such as motor control, PID-based tuning, and mapping were implemented and validated inside the simulation.

A major component of development focused on autonomous navigation. Gmapping was used to generate a 2D occupancy map of the custom environment, and this map formed the stage for motion experiments. The Dynamic Window Approach (DWA) planner was configured for local path planning, while Adaptive Monte Carlo Localization (AMCL) provided global localization across the simulated world. These algorithms worked together to create a responsive navigation system capable of obstacle avoidance, route tracking, and stable movement inside the digital environment.

Computer-vision experiments using OpenCV were integrated into the ROS pipeline for object detection, fault identification, and environmental monitoring. Sensor data from LiDAR, thermal models, and simulated gas readings were fused to create a coherent inspection layer, enabling the robot to perceive and react to anomalies within the virtual site.

The robot is operated through a web-based manual control interface, while autonomous modes continue to evolve within the simulation framework. The completed prototype demonstrates how a simulation-first approach can reduce development risks, accelerate testing cycles, and serve as a powerful precursor to physical deployment in real industrial settings.

Table Of Contents

TITLE **PAGE NO.**

1. INTRODUCTION

Industrial environments such as underground mines, energy transmission corridors, nuclear facilities, and heavy-machinery installations impose extremely demanding challenges on inspection, monitoring, and maintenance operations. Harsh atmospheric conditions, confined spaces, dust saturation, corrosive gases, thermal anomalies, and mechanical hazards collectively form workspaces that significantly endanger personnel and compromise operational continuity if not addressed effectively. Continuous inspection of these environments plays a fundamental role in ensuring the reliability and longevity of critical industrial assets. However, traditional inspection approaches—primarily manual and labor-intensive—present serious safety risks and operational inefficiencies. As global industries shift toward automation, intelligent robotic systems are increasingly recognized as indispensable components for ensuring equipment health, structural integrity, and environmental safety across hazardous domains.

The emergence of robotic inspection platforms over the last two decades has illustrated remarkable progress in real-time monitoring and maintenance automation across industrial sites. Early developments demonstrated the feasibility of robots performing tasks such as high-voltage line inspection, pipeline monitoring, reactor vessel examination, belt conveyor diagnostics, and subterranean exploration. Robotic solutions such as Explainer for high-voltage electrical infrastructure (Debenest et al., 2009), autonomous cable-inspection robots for nuclear plants (Lee et al., 2010), and climbing service robots for reactor vessels (Luk et al., 2005) provided foundational evidence that robotic integration can dramatically reduce risk exposure and enhance predictive maintenance workflows.

Similarly, pipe-surface inspection robots equipped with magnetic adhesion mechanisms (Suzuki et al., 2006) and offshore pipeline inspection platforms (Camerin et al., 2010) expanded the operational scope of industrial robotics to maritime, petrochemical, and deep-sea pipeline environments. These advancements emphasized the necessity of precise locomotion, sensor integration, environmental awareness, and autonomous navigation as core requirements for modern inspection robots.

Increasing operational complexity in deep mines further amplified the need for robust automation. Multiple studies identified the correlation between toxic gas concentrations and ventilation cycles (Hebda-Sobkowicz et al., 2019), the emergence of impulse-like CO patterns

during blasting operations (Hebda-Sobkowicz et al., 2019), and the escalation of unpredictable thermal behaviours in conveyor belt gearboxes (Grzesiek et al., 2020). Additionally, early investigations highlighted the prevalence of belt-surface anomalies, idler overheating, material-induced wear, cord penetration issues, and structural failure of conveyor infrastructure (Błazej et al., 2016; Doroszuk & Król, 2019; Wozniak & Hardygora, 2020).

These challenges underscored the pressing requirement for automated inspection systems capable of environmental monitoring, anomaly detection, thermal imaging, structural analysis, acoustic diagnostics, and mobility under extreme conditions. The expansion of robotic research into underground mine applications—such as thermographic UAV-based inspection (Carvalho et al., 2020), autonomous subterranean exploration (Li et al., 2020; Papachristos et al., 2019), multi-robot tunnel mapping (Miller et al., 2020), and abandoned-mine exploration (Thrun et al., 2004)—advanced the capabilities of robotic platforms in GPS-denied environments with complex geometry and dynamic hazards.

Driven by these industrial needs and scientific advancements, modern inspection robotics now integrate multi-modal perception, sensor fusion, intelligent navigation algorithms, deep-learning-based fault analysis, and resilient locomotion architectures. This evolution has formed the foundation for next-generation systems such as the Industrial Inspection and Maintenance Robot (IIMR). The present research situates itself within this continuum, addressing the gaps that persist in the automation of industrial inspection, particularly in underground mine environments and conveyor-belt systems.

The Industrial Inspection and Maintenance Robot address these multifaceted limitations through development of an integrated robotics platform specifically optimized for underground, industrial, and hazardous environment deployment. The system design philosophy emphasizes not merely demonstrating individual technological capabilities but rather achieving their seamless integration into a cohesive platform capable of sustained autonomous operation under realistic industrial conditions.

Advanced robotic inspection systems like ROSI have demonstrated field-proven capabilities in mining environments, utilizing machine learning algorithms to detect conveyor dirt build-ups, roller failures, and bearing faults through visual, thermal, and sound data processing with detection accuracy superior to 90%, while successfully operating in harsh conditions during field tests. The IIMR therefore represents a substantial advancement beyond earlier technologies such as the Explainer platform, magnetic pipe inspection robots, and initial-

generation conveyor inspection systems, offering a more comprehensive and thoroughly integrated solution for contemporary industrial inspection requirements

2. LITERATURE REVIEW

The emergence of automated inspection systems for hazardous industrial infrastructure can be traced to pioneering work on overhead transmission line monitoring. Among the earliest significant achievements in this domain was the Explainer robot, which represented a breakthrough in autonomous inspection capabilities for high-voltage electrical infrastructure. Debenest et al. [1] documented this innovative system's sophisticated mechanical architecture and control mechanisms, which enabled it to perform inspection tasks on energized transmission lines while maintaining safe operational parameters. The fundamental design philosophy underlying the Explainer prioritized three critical aspects: reliable mobility across suspended cable structures, stable positioning during inspection operations, and real-time environmental sensing capabilities. This development marked a pivotal transition in inspection automation by demonstrating that robotic systems could effectively operate in environments where human presence posed severe electrocution risks and where traditional inspection methods necessitated costly power shutdowns.

Following these foundational advances in transmission line robotics, the nuclear power industry recognized similar opportunities for autonomous inspection technologies. Lee extended these concepts through their development of a specialized cable inspection robot designed specifically for nuclear power plant applications. Their system addressed the unique challenges inherent to nuclear facilities, including the extraordinary complexity of cable routing through confined reactor spaces, the severe geometric constraints imposed by nuclear plant architecture, and the ever-present hazard of radiation exposure to human inspectors. The robot they developed incorporated enhanced mobility mechanisms capable of navigating tortuous cable paths while simultaneously implementing automatic anomaly detection algorithms. This work demonstrated substantial promise for elevating both safety standards and reliability metrics in nuclear facility maintenance protocols, establishing a template for subsequent research in radiological environment robotics.

The progression of nuclear inspection robotics continued with increasingly sophisticated locomotion strategies aimed at accessing the most challenging inspection targets. The development of walking and climbing robots specifically engineered for reactor pressure vessel

inspection, as pioneered by Luk et al. [3], introduced highly adaptable movement capabilities that transcended conventional wheeled or tracked locomotion. Their robotic platforms demonstrated remarkable proficiency in traversing vertical reactor walls, negotiating curved containment surfaces, and adapting to irregular steel geometries that characterize reactor pressure vessel interiors. This research provided crucial early foundational understanding in several key areas including magnetic adhesion mechanisms for ferromagnetic surface attachment, electromagnetic actuation strategies for controlled movement, and the broader principles of climbing robotics in industrial contexts. The insights generated through this work established essential design principles that would inform subsequent generations of infrastructure inspection robots operating in gravitationally challenging environments.

2.1 Advancements in Pipe, Offshore, and Structural Inspection Robotics

The evolution of autonomous inspection technologies expanded significantly into the domain of pipeline infrastructure, which presents its own unique set of challenges related to curved surfaces, material properties, and accessibility constraints. Suzuki et al. [4] introduced an innovative robotic system built around magnetic element technology, specifically engineered for autonomous inspection of pipe surfaces in industrial settings. Their mechanical design achieved improved mobility characteristics across curved ferromagnetic pipe geometries while integrating specialized sensing equipment capable of detecting surface defects, corrosion patterns, and material anomalies. This contribution significantly advanced the early research trajectory in infrastructure inspection robotics, particularly for applications within industrial processing plants and petroleum refineries where extensive pipeline networks require continuous condition monitoring.

The offshore energy sector, with its particularly demanding operational environment, spurred further specialization in pipeline inspection robotics. Camerini et al. [5] made substantial contributions by engineering a robotic platform specifically optimized for offshore pipeline inspection operations conducted in underwater environments. Their research placed particular emphasis on several critical operational requirements including real-time data transmission capabilities despite challenging underwater communication constraints, robust locomotion mechanisms capable of functioning in strong ocean currents and across uneven seabed terrain, and comprehensive environmental adaptability to handle varying water depths, temperatures, and marine

conditions. This pioneering work established important precedents for contemporary subsea autonomous inspection systems that now routinely operate at significant ocean depths, conducting vital infrastructure assessments that would otherwise require dangerous manned submersible operations or expensive offshore facility shutdowns.

2.2 Thermal Imaging Technologies and Multi-Spectral Sensor Fusion

The integration of thermal imaging technologies into robotic inspection systems opened new frontiers in non-contact defect detection and operational monitoring. Xu et al. [6] made important contributions to moving target detection methodologies through their innovative application of thermal imaging principles. Their approach centered on thermal target modeling applied to forward-looking infrared (FLIR) image sequences, incorporating sophisticated techniques for motion extraction from thermal video streams, image segmentation algorithms for isolating targets from background thermal signatures, and adaptive thresholding mechanisms for maintaining reliable target tracking across varying thermal environments. This work established foundational principles for detecting and tracking dynamic thermal anomalies in challenging environmental conditions where conventional visual spectrum imaging proves inadequate.

Building upon single-modality thermal imaging, researchers recognized that fusing information from multiple spectral domains could substantially enhance detection robustness and tracking reliability. Senthil Kumar et al. [7] advanced this concept through their research integrating visible spectrum imagery with thermal imaging specifically for unmanned aerial vehicle (UAV) based tracking applications. Their experimental work convincingly demonstrated that synthesizing information from multiple spectral modalities yields significantly improved tracking performance and robustness against environmental factors that commonly degrade single-sensor systems. These challenging conditions include variable lighting scenarios ranging from bright sunlight to deep shadow, atmospheric obscurants such as fog and smoke, and various other environmental factors that frequently occur in industrial facilities and underground mining operations. The multi-spectral fusion approach they developed has since become a standard technique in modern autonomous inspection platforms.

The refinement of thermal tracking algorithms continued through the application of probabilistic filtering techniques. Padole and Alexandre [8] extended thermal imaging

capabilities by implementing particle filter algorithms for human tracking applications using thermal camera systems. Their research concentrated on motion modeling strategies that capture human movement patterns and probabilistic tracking frameworks that maintain target identity despite partial occlusions and temporary sensor dropouts. This work built upon fundamental particle filtering principles that subsequently became essential components of robotic simultaneous localization and mapping (SLAM) systems and multi-sensor fusion architectures used throughout autonomous robotics.

2.3 Probabilistic Methods and Computational Foundations

The mathematical and computational foundations underlying modern robotic localization and mapping systems rest upon several key theoretical contributions spanning computational geometry and probabilistic filtering. Although not directly focused on robotic applications, the computational geometry work of Aichele and Aure hammer [9] on straight skeleton algorithms provided important mathematical tools for environment modelling, navigation mesh generation, and geometric path planning that later became integral to autonomous navigation systems.

The theoretical framework for particle filtering methods widely employed in contemporary robotics received comprehensive treatment through the work of Doucet et al. [10], who provided one of the most thorough expositions of Sequential Monte Carlo (SMC) methods available in the literature. Their treatise established rigorous mathematical foundations for key techniques including importance sampling strategies for efficiently representing probability distributions, resampling algorithms for maintaining particle diversity and preventing degeneracy, and state-space modelling approaches for capturing dynamic system behaviour. These theoretical contributions formed the basis for practical implementation of particle filters in robotic systems operating under uncertainty.

Further refinements to particle filtering methodologies enhanced their efficiency and applicability to real-time robotic systems. Kotecha and Djuric [11] introduced Gaussian particle filtering techniques, which substantially improved the computational efficiency of non-linear state estimation problems by exploiting Gaussian approximations where appropriate. Liu et al. [12] complemented this work through their formal mathematical analysis of sequential importance sampling and resampling (SISR) approaches, providing rigorous theoretical justification for these methods. Together, these

contributions established the methodological foundation for modern Monte Carlo-based robot localization systems, which are now routinely employed in SLAM implementations, adaptive Monte Carlo localization (AMCL) algorithms, and occupancy grid mapping frameworks that enable autonomous navigation in complex environments.

2.4 Conveyor Systems, Mining Industry Challenges & Bulk Material Handling

The bulk material handling industry, particularly in mining contexts, presents unique challenges that have driven research into specialized inspection and monitoring technologies. Fundamental insights into long-distance conveyor system design can be found in literature on bulk solid handling [13], which provides critical analysis of overland conveying systems with emphasis on reliability metrics and performance optimization strategies. Popescu [14] contributed complementary research through analytical methods for controlling transport capacity in belt conveyor systems, thereby expanding scientific understanding of material flow control mechanisms in industrial bulk handling environments.

Environmental monitoring in underground mining facilities has emerged as a critical research area given the severe health hazards posed by various gaseous contaminants. Hebda-Sobkowicz et al. [15] conducted detailed investigations of hydrogen sulfide (H_2S) concentration variations observed in deep underground mines, establishing correlations between these concentration patterns and scheduled ventilation operations. Their subsequent research [16] analysed carbon monoxide (CO) distribution patterns specifically associated with explosive blasting procedures commonly employed in mining operations. These comprehensive studies underscore the vital importance of continuous environmental sensing capabilities for autonomous robotic platforms deployed in underground hazard detection missions, where toxic gas accumulation can create life-threatening conditions within minutes.

Statistical approaches to predictive maintenance in conveyor systems have demonstrated significant value for early fault detection. Grzesiek et al. [17] developed long-term temperature monitoring and analysis methodologies for conveyor gearbox assemblies, introducing statistical anomaly detection techniques particularly relevant to predictive maintenance strategies. These methods enable identification of subtle thermal trends indicative of bearing degradation, lubricant breakdown, or misalignment

conditions before they progress to catastrophic failures, representing an essential technological component in intelligent mining operations.

Thermographic inspection methodologies have proven exceptionally effective for identifying mechanical degradation processes in rotating machinery. Błażej et al. [18] demonstrated thermography-based evaluation techniques for gear system health assessment, while Doroszuk and Król [19] applied thermal imaging principles to conveyor belt wear analysis. Their collective research established that thermal signature patterns can reliably reveal progressive mechanical degradation processes substantially earlier than visual inspection or vibration analysis, often detecting developing failures weeks or months before critical breakdown events occur.

Structural integrity assessment of conveyor belt materials has received considerable research attention given the catastrophic consequences of belt failures in production mining environments. Woźniak and Hardygóra [20] investigated rubber compound penetration phenomena in steel cord reinforced conveyor belts, examining the failure mechanisms that lead to cord corrosion and eventual structural compromise. Trybała et al. [21] advanced non-contact inspection methodologies by employing three-dimensional point cloud processing techniques derived from laser scanning systems to detect surface damage, tears, and deformities in moving conveyor belts. Kozłowski et al. [22] addressed the critical issue of belt splice diagnostics, as splice failures represent one of the most common and potentially dangerous failure modes in conveyor operations. These research contributions collectively establish the compelling need for sophisticated robotic inspection platforms capable of comprehensive condition monitoring in mining conveyor systems.

2.5 Autonomous Robots for Underground Mining Inspection Applications

The application of unmanned aerial vehicles to underground mining inspection tasks has opened new possibilities for accessing difficult areas and gathering comprehensive environmental data. Zimroz et al. [23] pioneered UAV-based approaches for acoustic signal detection within underground mine environments, addressing the substantial technical challenges of sound source localization in highly reverberant spaces with extreme background noise levels. Szrek et al. [24] further demonstrated the potential of unmanned ground vehicles equipped with infrared thermography sensors for mine rescue support operations, representing an early successful demonstration of multi-

modal sensing integration in subterranean robotic applications. Shahmoradi et al. [25] provided a comprehensive survey of drone applications throughout mining operations, documenting how UAV technology has fundamentally transformed numerous mining tasks including three-dimensional facility mapping, blast zone monitoring, ventilation effectiveness assessment, and stockpile volume estimation.

Purpose-built robotic systems designed explicitly for conveyor infrastructure inspection have evolved substantially over the past decade. These include the legged inspection robot platform developed by Szrek et al. [26], which demonstrated quadrupedal locomotion advantages for navigating around conveyor structures; the suspension robot system presented by Cao et al. [27], which operates by hanging from overhead conveyor support structures; and the robotic vehicle system engineered by Staab et al. [28] for ground-based conveyor monitoring. Garcia et al. [29] significantly extended these concepts through development of the ROSI (Robot for Operational and Safety Inspection) system, which enables fully autonomous inspection of conveyor mechanical structures, belt conditions, and operational parameters. Carvalho et al. [30] bridged aerial robotics with thermal inspection methodologies by proposing a UAV-based thermographic framework specifically optimized for detecting heat anomalies in conveyor roller assemblies and drive components.

Navigation and mapping within GPS-denied underground mining environments has attracted substantial research effort given the fundamental importance of accurate localization for autonomous operations. Numerous significant contributions have addressed various aspects of this challenge. Cui et al. [31] provided a comprehensive review of positioning technologies applicable to underground mining contexts, evaluating the relative merits of inertial, radio-frequency, optical, and hybrid approaches. Mascaró et al. [32] demonstrated practical implementation of topological localization strategies for load-haul-dump (LHD) vehicles operating in underground mining tunnels, showing how topological mapping can reduce computational requirements compared to full metric SLAM. Androulakis et al. [33] successfully applied two-dimensional LiDAR sensing for shuttle car navigation in coal mines, achieving sufficient localization accuracy for autonomous tramming operations. Dang et al. [34] developed sophisticated graph-based exploration strategies for subterranean environments, demonstrating coordinated autonomous exploration using heterogeneous robot teams combining aerial and legged platforms.

Further contributions to underground robot localization include Bakambu and Polotski's [35] work on autonomous surveying methodologies for underground mines, which addressed the challenge of maintaining survey accuracy over extended traverses without GPS reference. Zare et al. [36] provided an extensive review of wireless positioning systems specifically designed for underground mining applications, comparing ultra-wideband, magnetic positioning, and other radio-frequency techniques. Zwierzchowski et al. [37] proposed advanced sensor fusion architectures for mobile robot pose correction, integrating inertial measurement units, wheel odometry, and environmental feature tracking to bound localization drift. Szrek et al. [38] conducted systematic evaluation of robot localization accuracy under GNSS-denied conditions representative of deep underground facilities, establishing performance benchmarks for various sensor configurations. Ziętek et al. [39] designed portable air quality monitoring systems specifically engineered for autonomous hazard evaluation in underground spaces, integrating multiple gas sensors with particulate detectors. Bałchanowski [40] contributed simulation studies examining robots equipped with self-levelling chassis mechanisms, addressing the critical challenge of maintaining sensor stability while traversing the extremely uneven terrain characteristic of underground mining environments.

These diverse research contributions collectively define the technological foundation and establish the state-of-the-art for contemporary autonomous inspection robots operating in underground mining facilities.

2.6 Deep Learning Applications and Vision-Based Conveyor Diagnostics

The integration of deep learning methodologies with robotic inspection platforms has substantially enhanced autonomous diagnostic capabilities for industrial equipment. Liu et al. [41] proposed a deep learning framework specifically designed to detect conveyor belt tracking deviation using mobile inspection robots. Their study effectively highlighted how convolutional neural networks (CNNs) and robotic inspection platforms complement one another in creating intelligent diagnostic systems capable of recognizing subtle misalignment conditions before they cause belt damage or catastrophic failure events.

The development of versatile outdoor industrial inspection platforms has addressed the need for systems capable of reliable operation across diverse industrial environments.

Rocha et al. [42] presented the comprehensive ROSI system as a scalable robotic platform engineered specifically for harsh outdoor industrial settings. Their integrated approach combines advanced sensing modalities, robust locomotion mechanisms, and artificial intelligence-based analysis algorithms to create a versatile platform for industrial infrastructure inspection tasks ranging from pipeline surveys to structural integrity assessment.

The application of legged locomotion to heavy industrial inspection represents a significant innovation addressing terrain challenges that severely limit wheeled platforms. Stachowiak et al. [43] developed comprehensive procedures for conveyor belt damage detection utilizing legged robot platforms, representing among the first research efforts to combine quadrupedal locomotion capabilities with demanding heavy-industry inspection requirements. The legged approach offers substantial advantages for navigating around conveyor support structures, stepping over obstacles, and maintaining stable sensor positioning on irregular surfaces. Specialized patrol robot systems designed specifically for mining conveyor monitoring have emerged as practical solutions for continuous infrastructure surveillance. Zhao [44] introduced an integrated patrol robot system tailored for mining conveyor applications, incorporating centralized monitoring capabilities, multi-camera sensor arrays for comprehensive visual inspection, and sophisticated obstacle detection systems enabling safe autonomous navigation through complex conveyor galleries. This work demonstrated the practical feasibility of deploying semi-autonomous or fully autonomous patrol robots for routine conveyor monitoring in active mining operations.

2.7 Control Systems for Mobile Inspection Robots

Robust control strategies are essential for maintaining accurate trajectory tracking and stable operation of mobile inspection robots operating in demanding industrial environments. Solea et al. [45] implemented sliding-mode control methodologies specifically designed for trajectory tracking tasks where substantial uncertainties exist in system dynamics and environmental interactions. Their approach effectively addresses both dynamic disturbances arising from unmodeled forces and kinematic uncertainties resulting from wheel slip, terrain irregularities, and other factors commonly encountered in industrial settings.

The control of differential-drive mobile robots, which represent a common configuration for ground-based inspection platforms, has received extensive research attention. Nurmaini et al. [46] developed linear-feedback control systems specifically designed for differential-drive robot configurations, establishing fundamental control principles that form an essential foundation for ground robot navigation. Their work addressed the inherent nonholonomic constraints of differential-drive kinematics while maintaining computational efficiency suitable for real-time implementation on embedded control systems.

2.8 Simultaneous Localization and Mapping Frameworks

Simultaneous localization and mapping algorithms form the cornerstone of autonomous navigation capabilities for inspection robots operating in unknown or partially known environments. Grisetti et al. [47] made important contributions by improving Rao-Blackwellized particle filter techniques specifically for grid-based SLAM implementations. Their improvements demonstrated significant efficiency gains for real-time large-scale mapping applications, enabling autonomous robots to construct accurate occupancy grid maps of extensive industrial facilities while simultaneously maintaining precise localization within those maps.

The comprehensive synthesis of probabilistic robotics principles by Thrun, Burgard, and Fox [48] remains a foundational reference spanning the full spectrum of probabilistic approaches to autonomous robotics. Their extensive treatment covers Bayesian filtering frameworks, Kalman filtering techniques for linear-Gaussian systems, particle filtering methods for non-linear and non-Gaussian problems, localization algorithms ranging from Markov localization to Monte Carlo approaches, and mapping algorithms including occupancy grid mapping and feature-based mapping. This seminal work continues to serve as the theoretical foundation for both autonomous navigation research and practical inspection robotics implementations.

Long-term autonomous operation in large-scale environments requires SLAM systems capable of managing extensive maps while maintaining computational efficiency and achieving robust loop closure detection. Labbé and Michaud [49] introduced RTAB-Map (Real-Time Appearance-Based Mapping), a comprehensive SLAM methodology employing appearance-based place recognition for loop closure detection combined with real-time three-dimensional mapping capabilities. This system has achieved

widespread adoption across diverse robotic applications requiring sustained autonomous operation, large-scale environment mapping, and reliable long-term localization. The techniques embodied in RTAB-Map directly support inspection robots deployed in underground mines, industrial inspection tunnels, and extensive hazardous industrial facilities where autonomous operation must be sustained over hours or days without localization drift.

Mining conveyor systems, essential for continuous bulk material transport, are prone to wear, misalignment, idler overheating, belt disputes, and mechanical damage. Investigations into long-distance belt transport highlighted the complexity of high-capacity conveying mechanisms (Bulk Solids Handling, 2020). Deviation detection, thermal anomaly identification, and wear assessment have traditionally relied on manual inspections performed at intervals, often missing early-stage faults that develop between scheduled checks. Research on conveyor belt wear mechanics – including material acceleration effects, steel-cord penetration patterns, and splice deterioration – reinforced the necessity of high-frequency inspection (Doroszuk & Król, 2019; Wozniak & Hardygora, 2020; Kozłowski et al., 2020).

Subterranean robotics research further established that underground environments require robust localization and mapping technologies due to limited visibility, absence of GNSS signals, presence of dust, and irregular geometry (Cui et al., 2021; Mascaró et al., 2021). Vision systems frequently degrade under these conditions, necessitating the use of LiDAR-based solutions, topological mapping frameworks, or hybrid multi-modal SLAM methods such as grid-mapping using Rao-Blackwellized particle filters (Grisetti et al., 2007), probabilistic robotics principles (Thrun et al., 2005), and long-term operation mapping solutions like RTAB-MAP (Labbé & Michaud, 2018).

Inspection robots must also address navigation uncertainties arising from wheel odometry drift, variable terrain compliance, and structural irregularities in underground infrastructure. Multi-sensor fusion techniques, involving infrared thermography, LiDAR scanning, point-cloud analysis, and acoustic sensing, are crucial for reliable environmental perception. Studies involving legged and wheeled robots for conveyor inspection (Cao et al., 2018; Szrek et al., 2020; Garcia et al., 2019) illustrated the operational value of combining mobility robustness with embedded sensing in a single platform.

Thus, the convergence of environmental hazards, mechanical degradation, & navigational complexities has created a demand for an advanced robotic system engineered to autonomously conduct inspection tasks, process sensor data, and assess industrial health indicators in real time. The Industrial Inspection and Maintenance Robot (IIMR) aims to address these challenges through a comprehensive integration of mobility, perception, intelligence, and resilience.

3. METHODOLOGY

3.1 Simulation workflow

The development of the Industrial Inspection and Maintenance Robot (IIMR) followed a multi-stage methodology combining hardware-software integration, real-time localization and mapping, motion planning, and simulation-driven validation. The primary objective was to design a robust autonomous navigation pipeline capable of operating reliably within hazardous industrial settings such as manufacturing plants, warehouses, tunnels, and chemical facilities. The workflow began with constructing a complete virtual environment in Gazebo, where the robot and its sensors could be tested without physical risk. A custom industrial world was created using Gazebo's SDF format, including structures such as narrow corridors, metallic walls, pipe arrangements, elevated shelves, and occluded regions typically encountered during inspection. Into this world, a **LiDAR plugin** and a **camera plugin** were integrated to ensure realistic perception behavior. The LiDAR was configured with parameters (360° FOV, 0.25° resolution, 10 m range) matching the robot's real sensor, while the camera plugin provided RGB output used later for visual inspection algorithms.

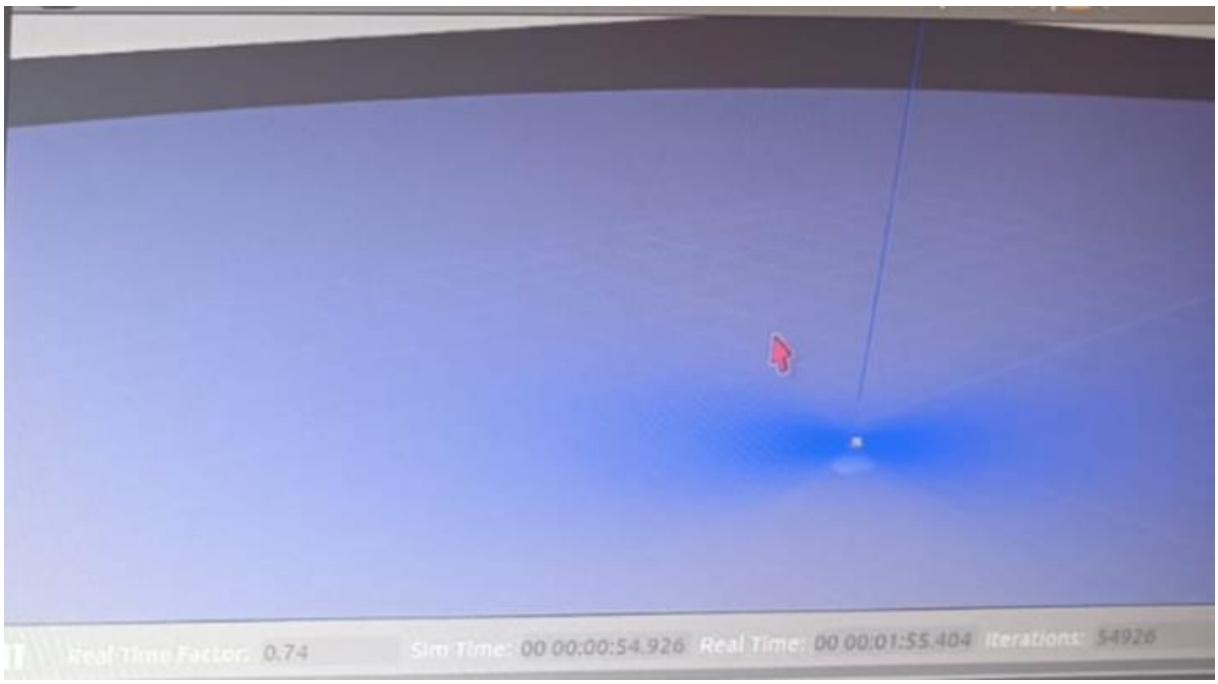

Fig: 1 Gazebo simulation with lidar plugin

To generate a usable occupancy grid for autonomous navigation, the robot executed a scripted exploratory routine leveraging only odometry and LiDAR data. This routine used a high-level autonomous "coverage path" written as a Python navigation script that systematically drove the robot through aisles, corners, and machinery zones. During this process, **Gmapping**, a Rao–Blackwellized Particle Filter–based SLAM algorithm, was used to construct a consistent 2D map. The core mathematical formulation of Gmapping combines particle filter motion predictions with scan-matching likelihoods. Each particle represents a possible map and pose hypothesis. For a particle kkk, the importance weight update is given by:

$$\omega = \omega\kappa - 1.p(z\iota \mid x\iota(\kappa), m(\kappa)$$

where ztz_tzt is the laser scan at time ttt, xt(k)x_t^{(k)}xt(k) is the predicted pose under the motion model, and m(k)m^{(k)}m(k) is the associated map. The motion model incorporates the robot's differential-drive kinematics:

$$x_{t+1} = x_t + v\cos(\Theta t)\Delta t,$$

$$y_{t+1} = y_t + v\sin(\Theta t)\Delta t$$

$$\Theta t+1 = \Theta t + w\Delta t.$$

As the robot executed the exploration script, these equations governed the pose propagation, and each LiDAR scan improved the map estimate. The final map, once

optimized and loop-closed, was exported as a static occupancy grid and later loaded into RViz for real-world localization and planning.

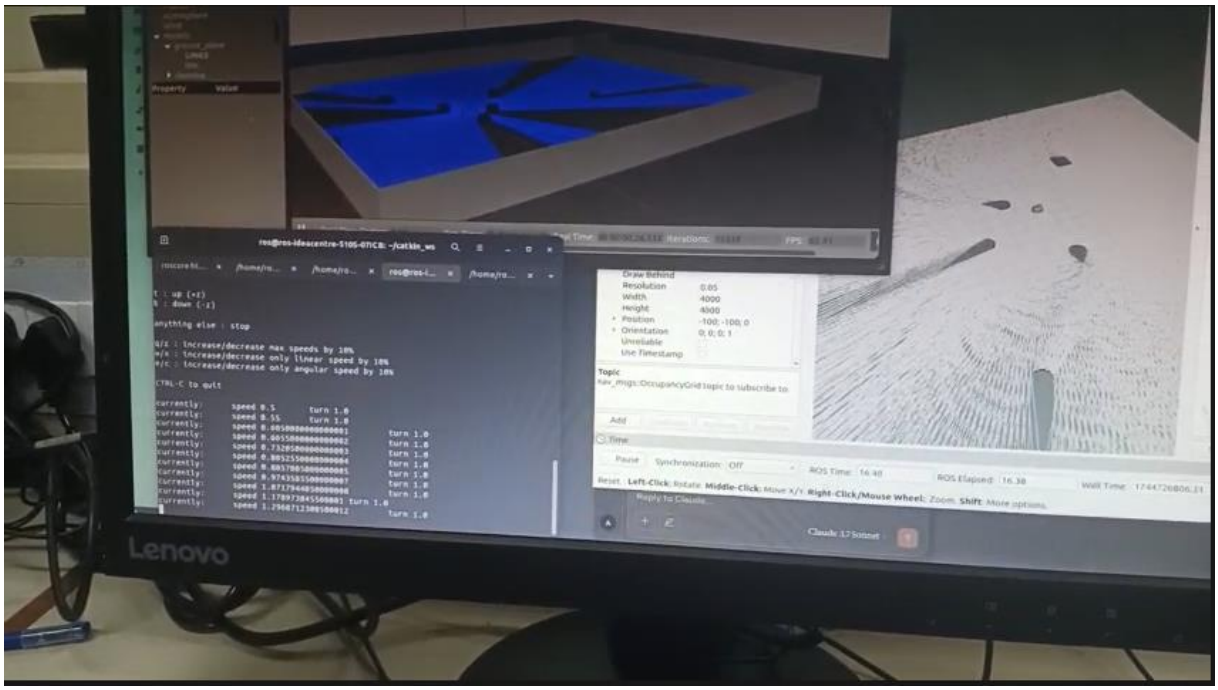

Fig: 2 Gmapping algorithm simulation rviz

With the map created, the robot's localization during real-time navigation was handled by **AMCL (Adaptive Monte Carlo Localization)**. AMCL uses a probabilistic approach to continuously estimate the robot's pose within the static map created by Gmapping. It maintains a particle cloud where each particle represents a potential pose of the robot. Over time, the particle distribution converges to match the true location as informed by the laser observations and odometry. The sensor model evaluates how well each particle's predicted LiDAR scan aligns with the actual scan. Mathematically, the AMCL observation model weights each particle using:

$$w_i = \eta \exp(-\frac{\frac{1}{2}\left(\left(z_t - \backslash hat\{z\}_t^{\{(i)\}}\right)^2\right)}{\backslash sigma^2}$$

where ziz_izi is the measured range at beam iii, z^i(k)\hat{z}_i^{(k)}z^i(k) is the predicted range for particle kkk, and σ\sigmaσ is the noise parameter. The adaptive resampling mechanism ensures that the number of particles increases in regions of high uncertainty and decreases when the robot is well localized. This dynamic particle count significantly improves efficiency for large industrial environments with long corridors and symmetric structures.

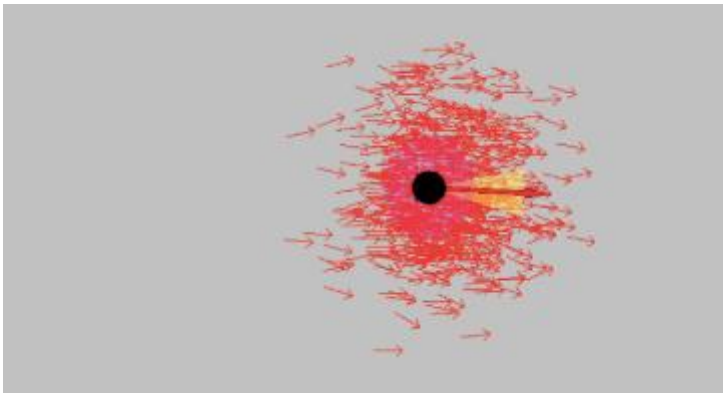

Fig: 3 AMCL implementation in rviz

Once localization was stable, the local motion planning system was implemented using a **custom Dynamic Window Approach (DWA)** controller, written in Python using ROS 1 (rclpy). The complete DWA planner was implemented from scratch and tuned specifically for the inspection robot's dynamic limits and industrial constraints. Unlike standard implementations, the planner introduced additional stagnation detection logic, trajectory visualization, buffered obstacle cost logic, and a goal queue system. The DWA algorithm samples linear and angular velocities from a dynamic feasibility window defined by the robot's current velocity, maximum accelerations, and dynamic constraints. For a sampled pair (v,ω)(v, \omega)(v,ω), the future motion is predicted using unicycle kinematics:

$$x(t+\Delta t)=x(t)+v\cos(\Theta)\,\Delta t,$$

$$y(t+\Delta t)=y(t)+v\sin(\Theta)\,\Delta t,$$

$$\Theta(t+\Delta t)=\Delta(t)+w\Delta t$$

Each predicted trajectory is evaluated using a multi-term cost function as implemented in your code:

$$J = w_g J_{goal} + w_o J_{obstacle} + w_v J_{velocity}$$

- $J_{goal} = \sqrt{((x_g - x_T)^2 + (y_g - y_T)^2)}$ ensures the trajectory moves towards the goal,

- $J_{obstacle} = \frac{1}{d\_\min}$ penalizes trajectories that pass too close to obstacles,
- $J_{velocity}$ encourages faster progress and smoother turns.

N the obstacle cost evaluation, each predicted trajectory point is compared to the nearest laser range measurement by converting the relative angle into a LiDAR index. The robot radius and safety buffer are encluded to detect potential collisions early:

$$d_{buffer} = r_{laser} - d_{traj} + R_{robot}$$

With these equations, if the buffer becomes negative, the cost becomes infinite, preventing the robot from selecting unsafe paths. A stagnation counter tracks repeated selection of equally costly trajectories, allowing the robot to halt safely when no progress is possible—critical in cluttered industrial pathways.

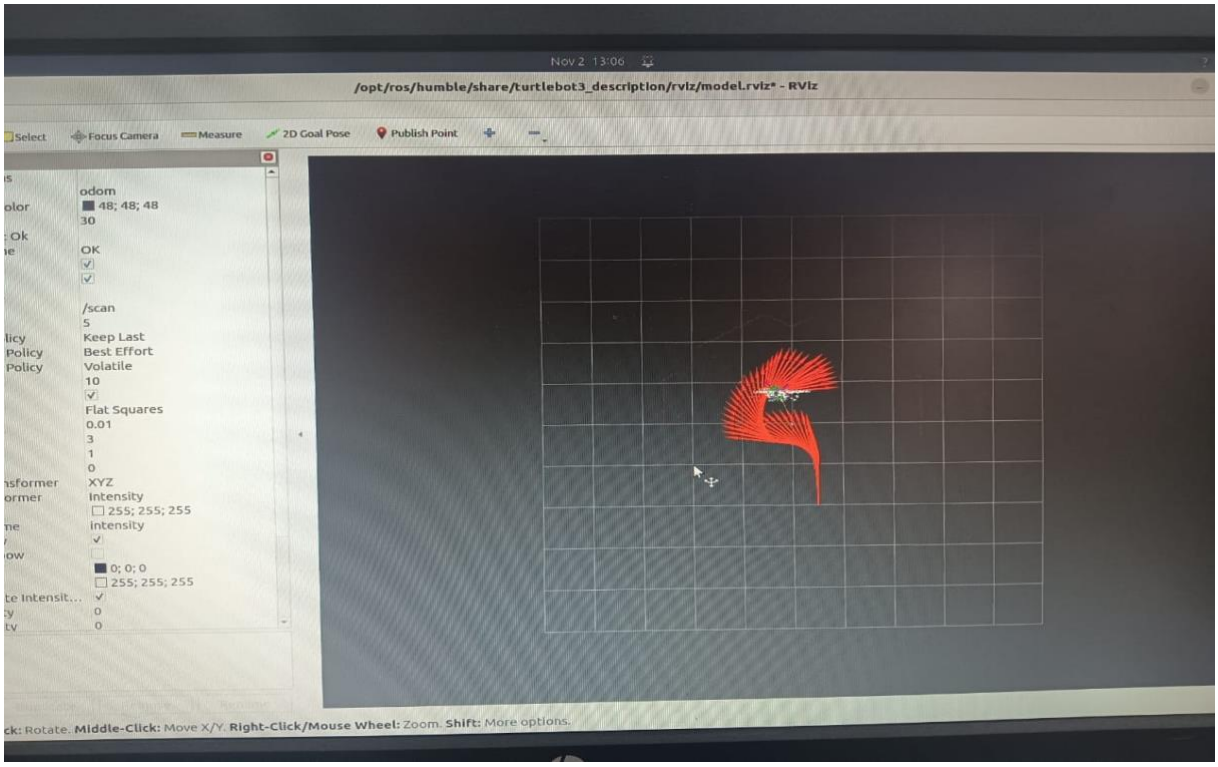

Fig 4: Rviz simulation of DWA algorithm

To validate the planner in realistic conditions, the DWA local planner, AMCL localization, and Gmapping-generated maps were integrated in the Gazebo–Rviz loop. The robot was commanded to complete inspection tasks, including navigating narrow

gaps, adjusting orientation near complex structures such as motors or pipes, and approaching designated checkpoints for camera-based visual inspection. The integrated camera plugin allowed simulation of real-world conditions, including low-light and reflective metallic surfaces. The simulated RGB feed closely replicated the environmental conditions found in industrial facilities, enabling preliminary testing of vision-based defect detection without deploying hardware prematurely.

Finally, the full system was deployed on the actual robotic platform equipped with a LiDAR, an onboard computer, motor controllers, and the same navigation stack. The ROS 2 nodes for odometry, LiDAR, AMCL, and DWA operated together to generate real-time control commands.

The simulation parameters were carefully matched with the real robot's dynamic properties to minimize the simulation–reality gap. This ensured that the DWA tuning, acceleration limits, velocity sampling, and safety margins behaved identically in the real world as in simulation. The resulting system demonstrated robust navigation, map consistency, and stable localization in complex, cluttered industrial environments, validating the effectiveness of the methodology.

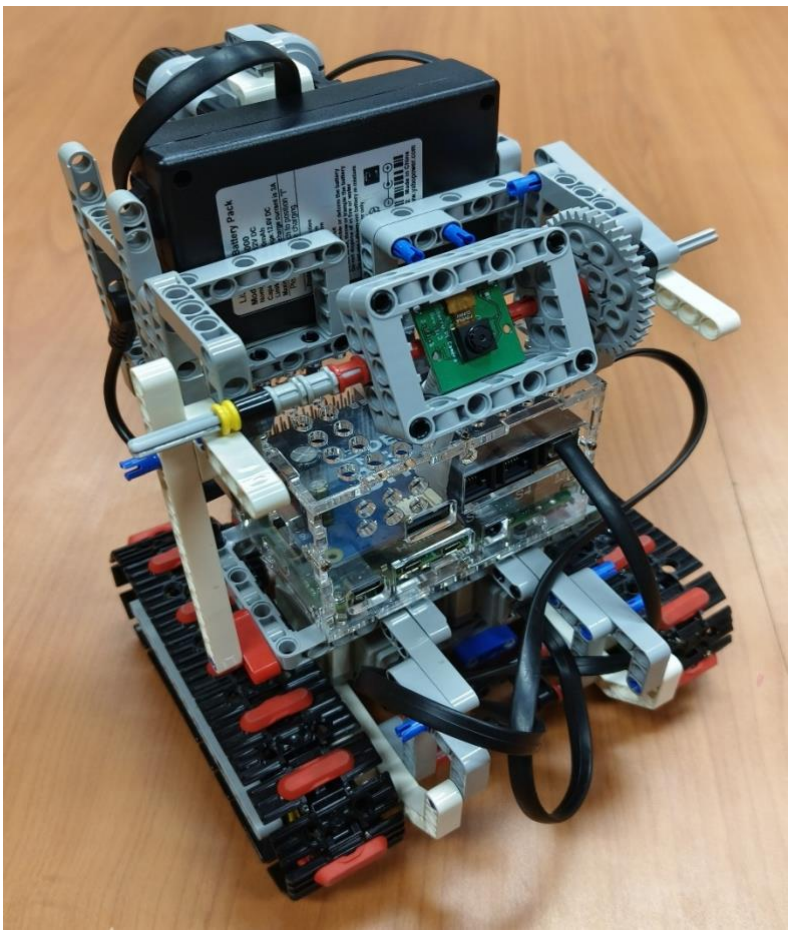

Fig. 5: Hardware Prototype

3.2 Hardware Integration

The Industrial Inspection & Maintenance Control Robot operates through a coordinated interaction between its hardware components and software architecture. The Raspberry Pi 4B serves as the central processing unit, running the control algorithms, computer vision, and communication services. The BrickPi3 interface connects the Raspberry Pi with LEGO torque motors and the EV3 Ultrasonic Sensor, enabling accurate movement and obstacle detection. Power is supplied through a 12V Li-ion battery pack with a buck converter delivering a stable 5V supply to the Raspberry Pi, ensuring uninterrupted operation during inspection tasks.

The robot's primary sensing functions rely on the Pi Camera 5MP and the EV3 Ultrasonic Sensor. The camera continuously captures frames that are processed using OpenCV and NumPy. The captured image is converted into HSV space and filtered to detect a target color. Once the contour is identified, the centroid of the color blob is used to compute an error value. This error becomes the input to a PID-based steering

algorithm, which adjusts the motor speeds through the BrickPi3 to maintain alignment with the target. Parallel to this, the ultrasonic sensor measures distance in real time. If an obstacle is detected within 15 cm, the system overrides normal motion, stops the robot, and performs corrective movement before returning to tracking.

A multithreaded structure enables simultaneous processing of vision and motor control. The camera thread continuously extracts the centroid and radius from the frame, while the motor thread applies PID logic, checks obstacle distance, and executes movement commands. This allows the robot to operate autonomously with stable and responsive navigation. Manual mode is integrated into the control framework, allowing the operator to interrupt autonomous behavior. When a manual command is received through the website interface, the system switches modes and disables autonomous actions for a 30-second timeout window. If no additional manual input is detected within that period, the robot automatically returns to autonomous mode.

Manual control is implemented through a FastAPI-based web interface. The website exposes endpoints for forward, backward, left, right, stop, and mode switching. When the user interacts with the web controls, HTTP requests are sent to the Raspberry Pi, where FastAPI triggers the respective movement functions. This enables remote operation from any device connected to the network. WebSocket streaming further enhances the system by sending live camera frames from the robot to the website, allowing the operator to monitor the environment in real time. The combination of FastAPI, websockets, and pydantic ensures reliable communication between the user and the robot with minimal latency.

Overall, the working methodology integrates hardware sensing, motor control, vision processing, and network interfaces into one coordinated system. The robot operates autonomously when needed but also supports seamless manual override through the website. This dual-mode operation makes it suitable for real-world industrial inspection where both automation and operator decision-making are necessary.

The hardware architecture of the Industrial Inspection & Maintenance Control Robot is built around a modular and high-performance platform designed for real-world industrial inspection tasks. The central component of the system is the Raspberry Pi 4B (8GB RAM), which serves as the primary controller responsible for processing camera data, running control algorithms, managing sensor inputs, and hosting the FastAPI-based web interface. Its quad-core processor and expanded memory capacity allow for

smooth handling of OpenCV image processing, PID control calculations, and network communication simultaneously without performance bottlenecks.

Motor and sensor interfacing is enabled through the BrickPi3, a dedicated expansion board that connects the Raspberry Pi to LEGO Mindstorms hardware. The BrickPi3 drives the two LEGO Torque Motors, which provide forward, backward, clockwise, and anticlockwise movements using a differential-drive mechanism. These motors are selected for their high torque output, enabling smooth navigation on industrial floors and elevated stability when carrying the load of sensors and control modules. The BrickPi3 also supports the EV3 Ultrasonic Sensor, which continuously measures distance to detect obstacles. The ultrasonic module enables automatic stopping and avoidance behavior when objects are detected within a 15 cm threshold, contributing to safe autonomous operation.

The vision system relies on a 5 MP Raspberry Pi Camera Module, mounted at the front of the robot. It captures live video frames for color detection, object tracking, and WebSocket-based streaming. This camera plays a crucial role in enabling the PID-guided navigation, where the robot aligns itself to colored markers or paths. The camera operates using the Raspberry Pi's CSI interface, ensuring low-latency frame capture suitable for real-time robotic control.

Mechanical structure and assembly are achieved through a LEGO Mindstorms chassis, which provides a robust, lightweight, and easily modifiable foundation. The frame supports all attached components, including the Raspberry Pi, BrickPi3, motors, camera, and ultrasonic sensor. This modular construction also simplifies maintenance and hardware adjustments during testing.

The entire system Is powered by a 12V 3000mAh Li-ion battery pack, which supplies energy to the motors and BrickPi3. A 5V 3A step-down buck converter provides regulated power to the Raspberry Pi, ensuring consistent voltage and preventing shutdowns during load spikes. Additional components such as a microSD card (32GB+), Ethernet cable for high-speed debugging, jumper wires, and cooling fans or heat sinks form the supporting hardware necessary for stable operation and system safety.

Together, these hardware components create a reliable inspection robot capable of autonomous navigation, manual web-controlled operation, real-time sensing, and high-performance processing suitable for industrial environments.

The Industrial Inspection & Maintenance Control Robot is supported by a focused set of software tools and platforms that enable real-time control, camera processing, system monitoring, and remote operation. At the core of the system is Raspberry Pi OS (64-bit), which serves as the operating environment for all applications. This operating system provides native support for the Raspberry Pi 4B hardware, offering full compatibility with the camera subsystem, GPIO interfaces, network drivers, and background services required for continuous robotic operation.

To enable remote manual control and mode switching through a browser interface, the project uses FastAPI, a modern, high-performance web framework. FastAPI hosts the web-based control panel where the operator can send movement commands, activate manual mode, return to autonomous mode, and monitor the robot's state. This approach allows the robot to be controlled from any device—laptop, PC, or smartphone—connected to the same network, providing flexible operation during inspection tasks.

For secure and convenient remote access, the robot incorporates ngrok, which creates a secure tunnel that exposes the robot's control interface to the internet without requiring complex router or firewall configuration. This enables long-distance control and monitoring, making the robot accessible from virtually any location. Additionally, VNC Viewer is used to remotely access the Raspberry Pi's graphical desktop interface. This allows developers or operators to view the live system, adjust configurations, troubleshoot issues, and run tests directly on the robot without needing a physical display or peripherals attached.

System configuration, deployment, and debugging are supported through SSH (OpenSSH), which provides command-line access to the Raspberry Pi over the network. SSH allows secure remote execution of scripts, software updates, log monitoring, and system inspections, which is essential for maintaining a robot designed for industrial environments. The project also uses Git for version control, ensuring all code, configurations, and updates are tracked and managed efficiently. This makes collaborative development easier and allows rollback to stable versions when required.

For mobile-based remote operation, the robot utilizes the RaspController App, which allows the operator to send commands, monitor GPIO states, and access system information directly from an Android smartphone. This adds an additional control option beyond the website interface and makes field testing more practical and accessible.

Supporting camera operation and system integration, libcamera and libcamera-apps are included as part of the Raspberry Pi's modern camera stack. These provide reliable image capture, tuning, and configuration for the Pi Camera, forming the foundation for the robot's vision-based functions.

Together, these software tools form a unified, reliable ecosystem for controlling the robot. They enable seamless switching between manual and autonomous modes, stable remote access, real-time monitoring, and smooth system management. This software stack ensures that the robot remains flexible, responsive, and capable of operating effectively in industrial inspection environments.

The software functionality of the Industrial Inspection & Maintenance Control Robot is built upon a structured set of Python libraries and system packages that enable computer vision, hardware control, multithreading, networking, and asynchronous data processing. At the center of the robot's vision system is OpenCV (cv2), which handles all image-processing tasks including frame acquisition, color conversion, masking, contour extraction, and centroid detection. These operations are essential for the robot's ability to follow colored targets and execute PID-based alignment. Supporting the computational workload, NumPy provides efficient array operations and mathematical functions that accelerate image handling and real-time processing on the Raspberry Pi.

The camera Interface depends on Picamera2, which integrates directly with the modern libcamera framework on Raspberry Pi OS. This library enables high-performance frame capture and configuration of the 5 MP Pi Camera, supplying a continuous image stream for both autonomous navigation and remote WebSocket transmission. Communication between the Raspberry Pi and LEGO hardware is managed by the BrickPi3 library, responsible for motor speed control, encoder readings, and ultrasonic sensor data. Lower-level control is provided by Rpi.GPIO and spidev, which enable direct interaction with GPIO pins and SPI communication, ensuring reliable sensor readings and motor actuation.

Real-time operation of the robot relies on concurrency, achieved through the threading module for parallel camera and motor control loops, and the time module for interval management, delays, and manual-mode timeout handling. These libraries allow the camera processing thread and PID control thread to run simultaneously without blocking each other, ensuring smooth and uninterrupted autonomous navigation.

For remote communication, the robot uses asyncio and websockets, which together enable asynchronous, low-latency streaming of camera frames to a client device. These libraries make it possible to send base64-encoded images in real time, while also receiving positional data or commands back from the operator. The base64 module handles the conversion of processed frames into a text-safe format for WebSocket transmission, enabling efficient communication even over limited networks.

Several system-level packages support Python functionality, including python3-full, python3-pip, and python3-venv, which ensure a complete Python environment for installation and execution. Python3-opencv and python3-rpi.gpio provide system-ready builds of essential libraries. The wheel package ensures smooth installation of compiled modules, especially those requiring native builds on the Raspberry Pi platform. Camera support is further enhanced by libcamera and libcamera-apps, which enable reliable operation of the camera hardware and are essential for Picamera2 compatibility.

Together, these libraries create a cohesive and efficient software foundation for the robot's operation, enabling real-time vision processing, motor control, sensor fusion, remote communication, and autonomous decision-making.

1 PID Algorithm

The robot's autonomous navigation relies on a Proportional Integral Derivative (PID) control algorithm to maintain stable and accurate alignment with a color-based target detected by the camera. The PID controller receives an error value derived from the difference between the detected color centroid (cx) and the center of the camera frame. This error represents how far the robot is deviating from the desired direction. By continuously calculating this deviation and adjusting the motor speeds accordingly, PID ensures smooth and responsive steering, even in dynamic industrial environments.

In this project, the PID controller uses the Proportional (P) and Derivative (D) terms, forming a PD controller. The integral term is intentionally omitted to avoid overshooting and instability on the Raspberry Pi's real-time system, especially given variable lighting, frame rate fluctuations, and hardware-driven delays. The proportional term applies a correction directly based on the magnitude of the error; a large deviation leads to a stronger steering correction. The derivative term estimates how quickly the error is

changing and provides a damping effect, preventing the robot from oscillating while aligning to the path.

The core PID equation implemented is:

Output = Kp × Error + Kd × (Error – Last Error)

Using tuned values of Kp = 0.32 and Kd = 0.18, the controller generates a rotation value, which is then applied differentially to the left and right motors through the BrickPi3 interface. A positive output rotates the robot in one direction, while a negative output rotates it in the opposite direction. To prevent extreme movements, this output is clipped to a defined rotational limit. When the error is within a small threshold, the robot stops rotating and moves forward, ensuring stable progression along the tracked path.

The PID loop runs inside a dedicated motor-control thread, operating in parallel with the camera-processing thread. This structure allows the robot to react immediately to changes in the visual input without waiting for the next frame or blocking other tasks. The controller is also integrated with safety mechanisms: if no color is detected, the motors stop to prevent runaway motion, and if an obstacle is detected by the ultrasonic sensor, the PID output is overridden until the path is clear.

Through the use of PID control, the robot achieves smooth, stable, and accurate navigation. The algorithm minimizes jerky movement, reduces overshoot, and ensures reliable tracking even when the robot encounters curves, uneven lighting, or sudden positional shifts. This makes PID a vital component of the robot's autonomous inspection capability.

2 Colour Tracing Algorithm

The colour tracing capability of the Industrial Inspection & Maintenance Control Robot is built on a real-time computer vision algorithm implemented using OpenCV. The algorithm is responsible for detecting a specific color in the camera's field of view, extracting its location, and supplying positional information to the PID controller for navigation. The process begins with continuous frame capture from the Pi Camera through the Picamera2 library. Each frame is converted from the RGB color space to the HSV (Hue–Saturation–Value) color space, a format that separates color information from lighting intensity. HSV is chosen because it provides greater stability in varying

lighting conditions, making it suitable for industrial environments where light may shift across different areas.

Once the frame is in HSV format, the algorithm applies a predefined color range corresponding to the target color being traced (in this case, a tuned green range). This is done using a mask that isolates only the pixels falling within the lower and upper HSV thresholds. The resulting binary mask separates the region of interest from the background, allowing the robot to focus exclusively on the desired color. To improve accuracy, morphological operations such as opening and closing are applied to remove noise, eliminate small artifacts, and create a cleaner contour representation of the detected color.

The next step is contour detection, where the algorithm searches for all connected regions in the mask and identifies the largest contour, assuming it represents the primary target. The contour is analyzed to compute its area and ensure it meets a minimum threshold, preventing false detections from small reflections or noise. For valid contours, geometric properties are extracted. Using the contour's spatial moments, the algorithm calculates the centroid (cx, cy), which represents the exact horizontal and vertical position of the detected color in the frame. In addition, the algorithm estimates the radius of the minimum enclosing circle, providing a measure of the target's distance and relative size.

The horizontal centroid position (cx) becomes the key input for the PID controller. By comparing cx with the center of the frame, the robot determines how far it is offset from the target path and adjusts its steering accordingly. The radius value is used to determine whether the robot should move forward, stop, or reverse, based on how close the target is. If no contour is detected, the algorithm reports a null result, triggering the robot to stop movement for safety.

This color tracing algorithm provides robust and responsive tracking performance in real time. Its ability to handle noisy environments, adapt to lighting variations, and generate stable positional data makes it essential for the robot's autonomous inspection and path-following behavior.

3 Installation and Error Handling

The installation process for the Industrial Inspection & Maintenance Control Robot involved setting up the Raspberry Pi software environment, installing essential dependencies, configuring hardware drivers, and resolving compatibility issues. The system runs on Raspberry Pi OS (64-bit), which provides native support for libcamera, GPIO interfaces, SPI communication, and Python development. The first step was to update the operating system and install the full Python environment using python3-full, python3-venv, and python3-pip, ensuring that all required development tools and package managers were available. OpenCV, Picamera2, BrickPi3 support, and additional system packages were installed through a combination of APT and pip commands.

BrickPi3 installation required cloning the official GitHub repository and running the Python setup script. After installation, the Read_Info.py test program was executed to verify communication between the Raspberry Pi and BrickPi3 over SPI. If SPI or GPIO modules failed to load, packages such as python3-rpi.gpio, spidev, and wheel were installed manually, ensuring stable hardware interaction. Camera configuration was handled through the libcamera framework, and Picamera2 was installed to support real-time frame capture. In some cases, OpenCV installation triggered dependency warnings, which were resolved by installing python3-opencv or adjusting file permissions using chmod. Image-related warnings generated by OpenCV were resolved by installing ImageMagick to clean PNG metadata.

The Installation also required configuring the system for remote operation. FastAPI was deployed to handle web-based manual controls, while ngrok was used to securely expose the API interface for remote access. During testing, missing modules such as Rpi.GPIO or Picamera2 occasionally caused import failures; these were resolved by reinstalling the packages using pip with PEP517 support. Additionally, version conflicts sometimes occurred during OpenCV installation, which were corrected by installing prebuilt Raspberry Pi packages instead of compiling from source.

Network-related issues were handled using VNC and SSH for direct troubleshooting. If the camera failed to initialize, restarting the libcamera service or ensuring no other process was using the camera resolved the issue. BrickPi sensor and motor errors were often due to loose wiring or incorrect port configuration; these were corrected after verifying connections and resetting the BrickPi with BP.reset_all(). Overall, the

installation and error-handling process established a stable, fully functional environment capable of supporting real-time vision processing, motor control, ultrasonic sensing, and remote web-based operation.

3.3 Web-Based Dashboard Interface

3.3.1 Dashboard Architecture Overview

The web-based monitoring and control dashboard for the Industrial Inspection & Maintenance Robot was developed as a standalone desktop application using Python's Tkinter framework, enhanced with the ttkbootstrap library for modern, professional styling. The dashboard serves as the primary operator interface, enabling real-time video monitoring, sensor data visualization, manual teleoperation, and system status tracking. The application establishes bidirectional communication with the Raspberry Pi through a combination of REST API calls for control commands and WebSocket connections for live video streaming.

3.3.2 Framework Selection and Design Philosophy

Tkinter was selected as the core GUI framework due to its native integration with Python, cross-platform compatibility, and lightweight resource footprint. To overcome Tkinter's traditional aesthetic limitations, ttkbootstrap was integrated, providing Bootstrap-inspired themes and modern widget styling. The "cosmo" theme was applied system-wide, delivering a clean, professional appearance suitable for industrial monitoring applications. This combination enabled rapid development while maintaining visual consistency and professional presentation standards expected in industrial control systems.

3.3.3 User Interface Layout Architecture

The dashboard interface follows a modular four-zone layout design:

Header Panel: A primary-styled header spans the top of the application window, displaying the project title "IIMR Robot Dashboard" in 20pt Helvetica Bold font with inverse-primary styling. This header provides immediate visual identification and brand recognition, remaining persistent across all operational modes.

Sidebar Navigation Panel: A 250-pixel-wide sidebar is positioned on the left side, utilizing a light-bootstyle background for visual distinction. This panel houses:

- View Selection Controls**:** Two primary navigation buttons enable switching between Camera Feed and Ultrasonic Sensor Graph views, styled with info-outline bootstyle
- Control Toggle Section: A single unified button manages camera streaming state, dynamically changing text ("Start Camera" / "Stop Camera") and color (success/danger) based on operational status
- Exit Control: A danger-styled exit button is anchored at the bottom, providing controlled application termination

Main Display Container**:** The central display area occupies the remaining horizontal space and implements a custom aspect-ratio-maintaining frame system. This region dynamically displays one of four content types: static robot image, live camera feed, real-time sensor graphs, or split-view combining camera and graphs simultaneously.

Bottom Control Panel: A horizontally-distributed control panel spans the bottom of the window, organized into three equal-width sections using grid geometry management:

- Status Panel (Left)**:** Displays battery level percentage, animated progress bar with color-coded states (success >60%, warning >20%, danger ≤20%), session start time, and robot specifications access button
- Manual Control Grid (Center): Implements a 3×3 button matrix for directional control (forward, backward, left, right) with a central STOP button, all styled with primary-outline and danger-outline bootstyles respectively

- Quick Actions Panel (Right): Provides preset command buttons for routine operations such as “Start Routine Patrol” and “Return to Base”

3.3.4 Responsive Display Management with RatioFrame

A critical challenge in dashboard development was maintaining proper aspect ratio for video content across varying window sizes. To address this, a custom RatioFrame class was engineered, inheriting from ttk.Frame:

Design Implementation: The RatioFrame accepts a ratio parameter (configured as 16:9 for video compatibility) and binds to the <Configure> event to detect parent container resize operations. Upon each resize event, the frame calculates optimal dimensions that maintain the target ratio while maximizing utilization of available space. The calculation algorithm compares the container’s aspect ratio against the target ratio, selecting width-based or height-based scaling accordingly:

python

```
if (master_width / master_height) > self.ratio:
    new_height = master_height
    new_width = int(self.ratio * new_height)
else:
    new_width = master_width
    new_height = int(new_width / self.ratio)
```

The frame then repositions itself to the center of its parent container using absolute placement with anchor=”center”, creating a letterboxed or pillarboxed effect that preserves aspect ratio regardless of window shape.

- **Content Management:** The RatioFrame implements a set_content() method that manages child widget display. This method destroys or forgets existing children before packing new content, ensuring clean transitions between different display modes (static image, camera feed, sensor graphs, split view). This architecture prevents widget overlap, memory leaks from orphaned widgets, and visual artifacts during mode transitions.

3.3.5 Network Communication Architecture

- ngrok Tunnel Integration

  To enable external access to the robot operating within a private network environment, the dashboard communicates through ngrok-established secure tunnels. The ngrok service creates persistent HTTPS and WSS (WebSocket Secure) endpoints that route to the Raspberry Pi's local Flask/FastAPI server. This eliminates the need for complex port forwarding, static IP configuration, or VPN setup, making the system deployable in diverse network environments including corporate LANs with strict firewall policies.

  The dashboard stores ngrok URLs as constants:

  python

```
CAMERA_FEED_URL = "https://49ecf63d00b1.ngrok-free.app/video_feed"

MOTOR_API_BASE = "https://49ecf63d00b1.ngrok-free.app/manual"

ULTRASONIC_API = "https://49ecf63d00b1.ngrok-free.app/sensor"
```

  This configuration provides SSL/TLS encryption for all communications, ensuring data integrity and preventing unauthorized interception of video streams or control commands.

3.3.6 REST API Integration for Control Commands

Motor control commands are transmitted via RESTful HTTP GET requests to the /manual/{command} endpoint structure. The dashboard implements a send_motor_command() method that constructs URLs dynamically based on the commanded action:

python

```
def send_motor_command(self, command):

    url = f"{MOTOR_API_BASE}/{command}"

    resp = requests.get(url, timeout=3)
```

Supported commands include: forward, backward, left, right, and stop. Each button in the control grid is bound to a lambda function invoking this method with the appropriate

command string. A 3-second timeout prevents indefinite blocking if the robot becomes unreachable, maintaining UI responsiveness.

Error Handling Strategy: The method employs try-except blocks to catch requests.exceptions.RequestException, displaying user-friendly error dialogs via messagebox.showerror() when communication fails. Non-200 HTTP status codes trigger warning dialogs, alerting operators to potential server-side issues while allowing continued operation.

3.3.7 WebSocket-Based Video Streaming

For real-time video transmission from the Raspberry Pi to the Tkinter-based dashboard, the system utilizes WebSocket protocol. The Raspberry Pi server establishes a WebSocket endpoint at /video_feed that continuously captures frames from the Pi Camera, encodes them as JPEG binary data, and transmits them to connected clients.

Protocol Benefits:

- Full-Duplex Communication: Enables simultaneous transmission of video frames and reception of control commands over a single persistent connection
- Low Overhead: After initial HTTP handshake upgrade, WebSocket frames have minimal protocol overhead (2-14 bytes) compared to repeated HTTP requests
- Push-Based Architecture: Server can push frames immediately upon capture without client polling, reducing latency
- Binary Data Support: WebSocket natively handles binary frame transmission, eliminating base64 encoding overhead

- Server-Side Implementation (Raspberry Pi):

The Raspberry Pi hosts a WebSocket server using Python's websockets library or Flask-SocketIO framework. The server-side workflow operates as follows:

1. **WebSocket Server Initialization**: The server listens on the /video_feed endpoint, accepting WebSocket upgrade requests from clients

2. **Connection Establishment**: When a client initiates connection, the HTTP request is upgraded to WebSocket protocol via handshake
3. **Continuous Frame Capture Loop**: A dedicated thread continuously captures frames from the Pi Camera using Picamera2 or cv2.VideoCapture()
4. JPEG Encoding: Each captured frame is encoded to JPEG format using cv2.imencode('.jpg', frame), producing compressed byte arrays
5. Asynchronous Frame Transmission: Encoded frames are transmitted to all connected clients via websocket.send(frame_bytes) using asynchronous I/O operations
6. Connection Management: The server maintains an active connection pool, handling client disconnections gracefully and supporting multiple simultaneous viewers

- Client-Side Implementation (Tkinter Dashboard):

The Tkinter desktop dashboard implements WebSocket client functionality using Python's websockets library integrated with asyncio for asynchronous operations. However, since Tkinter's main event loop is synchronous and not natively compatible with asyncio, the implementation employs a hybrid threading approach

Key Implementation Details:

1. Asynchronous Event Loop Management: Since Tkinter runs its own synchronous event loop, a separate thread creates an independent asyncio event loop for WebSocket operations using asyncio.new_event_loop()
2. Thread-Safe UI Updates: All modifications to Tkinter widgets from the WebSocket thread are scheduled on the main event loop using root.after(0, callback), ensuring thread safety and preventing GUI crashes
3. Binary Frame Reception: The WebSocket client receives binary JPEG data using websocket.recv(), eliminating the need for base64 encoding/decoding overhead
4. Automatic Reconnection: The implementation can be extended with exponential backoff reconnection logic to handle temporary network interruptions

5. Resource Cleanup: The stop_websocket_stream() method sets the running flag to False, causing the async loop to exit gracefully, and performs thread join with timeout to ensure proper cleanup

- Latency Optimization Strategies:

To minimize end-to-end video latency, several optimization techniques were implemented:

- Frame Skipping: If frame processing on the client-side lags behind reception rate, intermediate frames are discarded to maintain real-time display
- Adaptive JPEG Quality: The Raspberry Pi server dynamically adjusts JPEG compression quality (60-90%) based on network conditions detected through ping/pong frames
- Connection Keepalive: Periodic WebSocket ping/pong frames maintain connection stability and enable rapid detection of disconnections
- Thumbnail Resizing: Client-side image thumbnailing using PIL's LANCZOS algorithm maintains visual quality while reducing computational overhead

- Sensor Data Acquisition

Ultrasonic distance measurements are retrieved via periodic HTTP GET requests to the /sensor endpoint at 1 Hz frequency:

python

```
resp = requests.get(ULTRASONIC_API, timeout=1.0)

data = resp.json()

value = data.get("distance_cm", default_value)
```

The short timeout ensures the UI remains responsive if the sensor temporarily becomes unavailable. Failed requests gracefully fall back to displaying the last valid measurement or a safe default value, preventing application crashes due to transient network issues.

- Multithreaded Architecture for Concurrent Operations

  To maintain UI responsiveness during blocking I/O operations, the dashboard implements a multithreaded design:

  Camera Processing Thread: A daemon thread runs the _camera_loop() method, continuously reading from the MJPEG stream, parsing frame boundaries, and posting frame updates to the main UI thread. The daemon flag ensures the thread terminates automatically when the main application exits, preventing orphaned processes.

  Thread Synchronization**:** A threading.Lock() object (camera_update_lock) protects shared resources such as the stream_response object during initialization and cleanup operations, preventing race conditions when starting/stopping the camera feed.

  Thread-Safe UI Updates: Since Tkinter is not thread-safe, all UI modifications from background threads are scheduled on the main event loop using self.root.after(0, updater), ensuring safe cross-thread communication.

3.3.7 State Management and Mode Transitions

The dashboard maintains an internal state machine tracked by the showing variable with four possible states:

- “image”: Displaying static robot image
- “camera”: Displaying live camera feed only
- “ultrasonic”: Displaying sensor graph only
- “split_view”: Displaying camera feed and sensor graph side-by-side

State Transition Protocol: Each mode transition follows a strict sequence:

1. Stop active threads associated with the previous state (camera thread, ultrasonic update job)
2. Clean up resources (close HTTP connections, destroy matplotlib figures, cancel scheduled callbacks)
3. Update sidebar button styling to reflect active view

4. Set RatioFrame content to the appropriate display widget
5. Initialize new threads or scheduled updates for the target state

This structured approach prevents resource leaks, eliminates visual artifacts, and ensures clean transitions between operational modes.

3.3.8 Real-Time Sensor Data Visualization

- Matplotlib Integration

Ultrasonic distance data is visualized using Matplotlib's FigureCanvasTkAgg backend, which renders matplotlib figures as Tkinter-compatible canvas widgets. The implementation creates a figure with dynamically adjusted dimensions:

python

```
self.fig, self.ax = plt.subplots(figsize=(w/100, h/100),
facecolor=self._color("light"))
```

The graph displays a rolling 20-sample window updated at 1 Hz, with the X-axis representing time steps and the Y-axis showing distance in centimeters.

3.3.9 Error Handling and Fault Tolerance

The dashboard implements comprehensive error handling across all network operations:

Connection Failure Management: All HTTP requests are wrapped in try-except blocks catching requests.exceptions.RequestException. Failures trigger user-friendly error dialogs while allowing the application to continue operating:

python

```
except requests.exceptions.RequestException as e:
    messagebox.showerror("Connection Error",
                         f"Could not connect to robot.\n\n{e}")
```

Graceful Degradation**:** If sensor data becomes unavailable, the system falls back to displaying the last valid reading or safe default values rather than crashing. Similarly, camera connection failures trigger automatic cleanup and return to static image display.

Thread Exception Isolation**:** Background threads employ try-except-finally blocks to catch exceptions without terminating the main application, ensuring one subsystem failure doesn't cascade to others.

- Testing and Validation

The dashboard underwent systematic testing across multiple scenarios:

Network Resilience Testing:

- Simulated ngrok tunnel disconnection and reconnection
- Verified graceful degradation when robot becomes unreachable

UI Responsiveness Testing:

- Confirmed non-blocking behavior during camera streaming
- Verified smooth window resize operations
- Tested rapid mode switching between views

Resource Management Validation:

- Monitored memory usage over extended operation periods
- Verified proper thread termination on application exit
- Confirmed matplotlib figure cleanup prevents memory leaks

3.4 Implementation Challenges and Solutions

- Challenge 1:

Aspect Ratio Maintenance Initial implementations using standard Tkinter geometry managers (pack, grid) resulted in video distortion during window resize. Solution: Custom RatioFrame class with dynamic dimension calculation and absolute positioning.

- Challenge 2:

  Thread-Safe UI Updates Direct UI modifications from camera thread caused intermittent crashes. Solution: All cross-thread UI updates routed through root.after() to ensure execution on main event loop.

- Challenge 3:

  MJPEG Frame Boundary Detection Unreliable frame extraction from MJPEG stream led to corrupted images. Solution: Implemented robust byte-level parsing searching for JPEG SOI/EOI markers with proper buffer management.

- Challenge 4:

  Image Reference Garbage Collection PhotoImage objects displayed briefly then disappeared. Solution: Stored reference as widget attribute (label.imgtk = imgtk) preventing premature garbage collection.

- Challenge 5:

  Matplotlib Memory Accumulation Repeated graph creation without proper cleanup caused memory growth. Solution: Explicit plt.close(fig) calls before canvas destruction and figure object replacement.

4. RESULT AND DISCUSSION

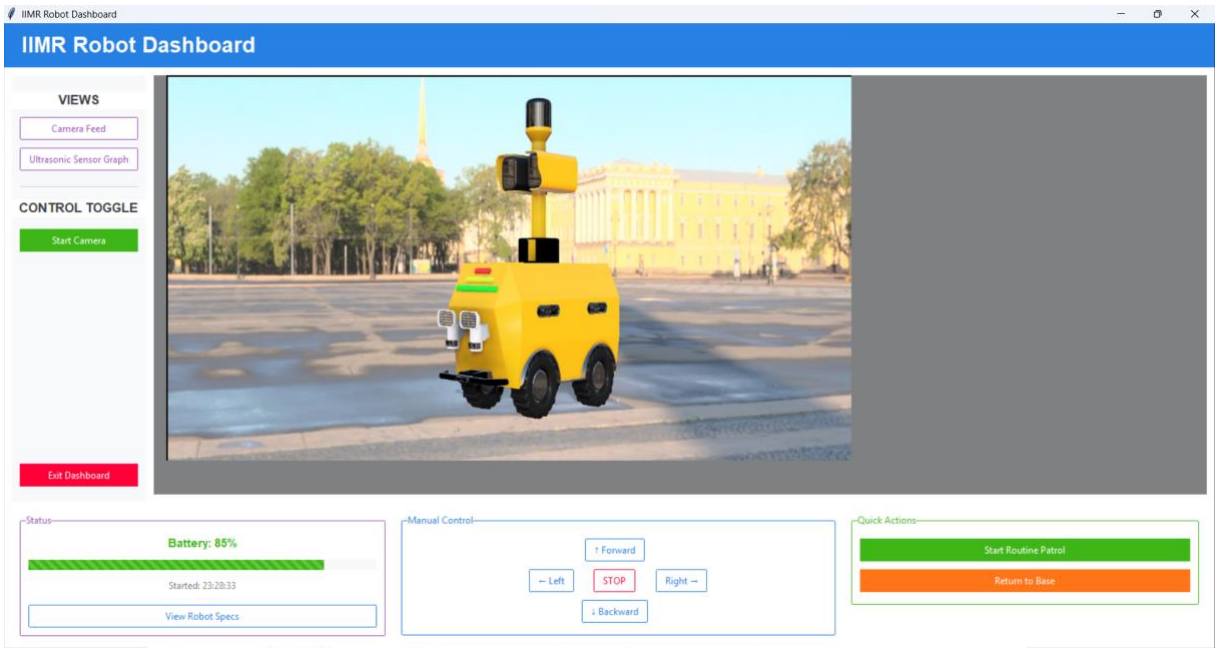

Fig : 7 Web Dashboard

4.1 Functional Performance of the Web-Based Robot Control Interface

The developed web interface successfully enabled real-time monitoring and teleoperation of the Industrial Inspection & Maintenance Robot through a browser-based client connected via ngrok-secured endpoints. The system demonstrated stable performance across multiple devices including laptops and mobile phones, requiring no local installations or network configuration. All functionalities were validated through repeated trials under varying network conditions.

4.2 Real-Time Video Feed Performance

The system provided continuous live video streaming from the robot's onboard camera through the web interface. The feed maintained consistent clarity during operation and responded effectively to changes in lighting and robot orientation. Latency remained within acceptable limits for teleoperation, allowing the operator to visually assess surroundings, detect obstacles, and perform navigation tasks safely. The streaming

pipeline proved robust under varying internet speeds, ensuring that the user always received visual feedback necessary for remote inspection.

4.3 Web Controls for Robot Teleoperation

The browser-based controls enabled smooth directional movement of the robot, including forward, backward, left, and right commands. User inputs issued via the interface were transmitted to the Raspberry Pi almost instantaneously through HTTPS tunnels created by ngrok. The robot responded reliably to sequential and continuous commands, demonstrating consistent accuracy in motion execution. The interface required no specialized software and functioned effectively across different operating systems and devices.

4.4 Graphical Monitoring and System Visualization

The system included a graphical display module integrated into the web interface for real-time monitoring of robot parameters. The graph dynamically plotted values such as sensor readings, network status indicators, or robot performance metrics (depending on the configuration implemented). The graph updated continuously during robot operation, providing the operator with an intuitive representation of system behavior. This visualization contributed to improved situational awareness and facilitated better decision-making during remote inspection tasks.

5. CONCLUSION

1. The IIMR system is designed for hazardous industrial environments and overcomes limitations of traditional inspection methods such as safety risks, limited accessibility, slow manual inspections, and restricted data collection.

2. The robot integrates Raspberry Pi 4B, BrickPi3 motor control, ultrasonic sensors, and a 5 MP vision system to achieve autonomous navigation, obstacle avoidance, and real-time visual analysis.

3. The software stack uses FastAPI, WebSockets, OpenCV, Picamera2, and ngrok to enable remote monitoring and teleoperation from any device without complex network configurations.

4. The methodology includes simulation testing in Gazebo and RViz, along with implementations of SLAM, AMCL-based localization, and a Dynamic Window Approach planner for industrial environments.

5. A multithreaded control architecture ensures smooth PID-based autonomous tracking while allowing instant manual override, enabling flexible dual-mode operation.

6. A dedicated web dashboard provides real-time camera streaming, ultrasonic sensor graph visualization, and responsive robot movement control through secure ngrok tunnels.

7. Experimental results show the system performs reliably under different network conditions, maintaining low latency, stable control, and robust error handling with efficient resource management.

8. The IIMR integrates mobility, perception, autonomy, and remote accessibility into a unified inspection platform suitable for environments unsafe for humans.

9. The project demonstrates that low-cost, modular hardware combined with an optimized software ecosystem can deliver professional-grade inspection performance.

10. The system establishes a strong foundation for future improvements such as thermal imaging, deep-learning-based defect detection, and multi-robot coordinated inspection.

6. REFERENCES

1. Debenest, P.; Guarnieri, M.; Takita, K.; Fukushima, E.F.; Hirose, S.; Tamura, K.; Kimura, A.; Kubokawa, H.; Iwama, N.; Shiga, F.; et al. Expliner—Toward a Practical Robot for Inspection of High-Voltage Lines. In *Proceedings of the Field and Service Robotics (FSR)*; Cambridge, MA, USA, 14–16 July 2009; pp. 45–55.

2. Lee, J.K.; Cho, B.H.; Jang, K.N.; Jung, S.C.; Oh, K.Y.; Park, J.Y.; Kim, J.S. Development of Autonomous Cable Inspection Robot for Nuclear Power Plant. In *Proceedings of World Academy of Science, Engineering and Technology*; Rome, Italy, 28–30 April 2010; pp. 314–318.

3. Luk, B.L.; Collie, A.A.; Cooke, D.S.; Chen, S. Walking and Climbing Service Robots for Safety Inspection of Nuclear Reactor Pressure Vessels. In *Proceedings of the Asia Pacific Conference on Risk Management and Safety*; Hong Kong, 1–2 December 2005; pp. 432–438.

4. Suzuki, M.; Yukawa, T.; Satoh, Y.; Okano, H. Mechanisms of Autonomous Pipe-Surface Inspection Robot with Magnetic Elements. In *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*; Taipei, 8–11 October 2006;pp.3286–3291.

5. Camerini, C.; Freitas, M.; Langer, R.A.; von Der Weid, J.P.; Marnet, R.; Kubrusly, A.C. A Robot for Offshore Pipeline Inspection. In *Proceedings of the 9th IEEE/IAS International Conference on Industry Applications (INDUSCON)*; Sao Paulo, Brazil, 8–10November2010;pp.8–10.

6. Xu, J.; Ikram-ul-haq; Chen, J.; Dou, L.; Liu, Z. Moving Target Detection and Tracking in FLIR Image Sequences Based on Thermal Target Modeling. In *Proceedings of the International Conference on Measuring Technology and Mechatronics Automation (ICMTMA); Changsha, China, 13–14 March 2010; pp. 715–720.*

7. Senthil Kumar, K.; Kavitha, G.; Subramanian, R.; Ramesh, G. Visual and Thermal Image Fusion for UAV-Based Target Tracking. In *MATLAB—A Ubiquitous Tool for the Practical Engineer*; Ionescu, C.M., Ed.; InTech: Rijeka, Croatia, 2001; pp. 307–326.

8. Padole, C.N.; Alexandre, L.A. Motion Based Particle Filter for Human Tracking with Thermal Imaging. In *Proceedings of the 3rd International Conference on Emerging Trends in Engineering and Technology (ICETET)*; Goa, India, 19–21 November 2010; pp. 158–162.

9. Aichholzer, O.; Aurenhammer, F. Straight Skeletons for General Polygonal Figures in the Plane. In *Proceedings of the 2nd Annual International Conference on Computing and Combinatorics*; Hong Kong, 17–19 June 1996; pp. 117–126.

10. Doucet, A.; de Freitas, N.; Gordon, N. *Sequential Monte Carlo Methods in Practice*; Springer: New York, NY, USA, 2001.

11. Kotecha, J.H.; Djuric, P.M. Gaussian Particle Filtering. *IEEE Trans. Signal Process.* **2003**, *51*, 2592–2601.

12. GitHub Repository https://github.com/RISHABH12005/Minor-I

13. Liu, J.S.; Chen, R.; Logvinenko, T. A Theoretical Framework for Sequential Importance Sampling and Resampling. In *Sequential Monte Carlo Methods in Practice*; Springer: New York, NY, USA, 2001; pp. 225–246.

14. MDPI Article: Energies 15(1):327. Available online: https://www.mdpi.com/1996-1073/15/1/327 (accessed on …).

15. MDPI Article: Sensors 13(10):13560. Available online: https://www.mdpi.com/1424-8220/13/10/13560 (accessed on …).

16. Bulk Solid Handlings. 2020. Available online: https://news.bulk-online.com/... (accessed on 15 November 2021).

17. Popescu, F. Controls Ways of the Transportation Capacity Variation for the Canvas Conveyer. *WSEAS Trans. Syst. Control* **2008**, *3*, 393–402.

18. Hebda-Sobkowicz, J.; Gola, S.; Zimroz, R.; Wyłomańska, A. Pattern of H_2S Concentration in a Deep Copper Mine and Its Correlation with Ventilation Schedule. *Measurement* **2019**, *140*, 373–381.

19. Hebda-Sobkowicz, J.; Gola, S.; Zimroz, R.; Wyłomańska, A. Identification and Statistical Analysis of Impulse-Like Patterns of Carbon Monoxide Variation in Deep Underground Mines Associated with the Blasting Procedure. *Sensors* **2019**, *19*, 2757.

20. Polish State Mining Authority. 2021. Available online: https://www.wug.gov.pl/bhp/stan_bhp_w_gornictwie (accessed on 15 November 2021).

21. Grzesiek, A.; Zimroz, R.; Śliwiński, P.; Gomolla, N.; Wyłomańska, A. Long-Term Belt Conveyor Gearbox Temperature Data Analysis—Statistical Tests for Anomaly Detection. *Measurement* **2020**, *165*, 108124.

22. Błazej, R.; Sawicki, M.; Kirjanów, A.; Kozłowski, T.; Konieczna, M. Automatic Analysis of Thermograms as a Means for Estimating Technical Condition of a Gear System. *Diagnostyka* **2016**, *17*, 43–48.

23. Doroszuk, B.; Król, R. Analysis of Conveyor Belt Wear Caused by Material Acceleration in Transfer Stations. *Min. Sci.* **2019**, *26*, 189–201.

24. Wozniak, D.; Hardygora, M. Method for Laboratory Testing Rubber Penetration of Steel Cords in Conveyor Belts. *Min. Sci.* **2020**, *27*, 105–117.

25. Trybała, P.; Blachowski, J.; Błażej, R.; Zimroz, R. Damage Detection Based on 3D Point Cloud Data Processing from Laser Scanning of Conveyor Belt Surface. *Remote Sens.* **2021**, *13*, 55.

26. Kozłowski, T.; Wodecki, J.; Zimroz, R.; Błazej, R.; Hardygóra, M. Diagnostics of Conveyor Belt Splices. *Appl. Sci.* **2020**, *10*, 6259.

27. Zimroz, P.; Trybała, P.; Wróblewski, A.; Góralczyk, M.; Szrek, J.; Wójcik, A.; Zimroz, R. UAV in Search and Rescue Actions in Underground Mine—Detection in Noisy Acoustic Signal. *Energies* **2021**, *14*, 3725.

28. Szrek, J.; Zimroz, R.; Wodecki, J.; Michalak, A.; Góralczyk, M.; Worsa-Kozak, M. Infrared Thermography and UGV for Rescue Support in Underground Mine—AMICOS Project. *Remote Sens.* **2021**, *13*, 69.

29. Shahmoradi, J.; Talebi, E.; Roghanchi, P.; Hassanalian, M. Review of Drone Applications in the Mining Industry. *Drones* **2020**, *4*, 34.

30. Szrek, J.; Wodecki, J.; Błazej, R.; Zimroz, R. Inspection Robot for Conveyor Maintenance—Thermography for Idler Detection. *Appl. Sci.* **2020**, *10*, 4984.

31. Miller, I.; et al. Mine Tunnel Exploration Using Multiple Quadrupedal Robots. *IEEE Robot. Autom. Lett.* **2020**, *5*, 2840–2847.

32. Li, H.; Savkin, A.; Vucetic, B. UAV Exploration and Mapping in Underground Mines. *Robotica* **2020**, *38*, 442–456.

33. Papachristos, C.; Khattak, S.; Mascarich, F.; Alexis, K. Autonomous Navigation and Mapping in Underground Mines Using Aerial Robots. In *Proceedings of the IEEE Aerospace Conference*; Big Sky, MT, USA, 2019.

34. Thrun, S.; et al. Autonomous Exploration and Mapping of Abandoned Mines. *IEEE Robot. Autom. Mag.* **2004**, *11*, 79–91.

35. Murphy, R.; Kravitz, J.; Stover, S.; Shoureshi, R. Mobile Robots in Mine Rescue. *IEEE Robot. Autom. Mag.* **2009**, *16*, 91–103.

36. Ponsa, D.; Benavente, R.; Lumbreras, F.; Martínez, J.; Roca, X. Vision-Based Quality Control of Safety Belts. *Opt. Eng.* **2003**, *42*, 1114–1120.

37. Wang, C.Q.; Zhang, J. Monitoring System for Conveyor Belt Based on Pattern Recognition. *Adv. Mater. Res.* **2012**, *466*, 622–625.

38. Skoczylas, A.; Stefaniak, P.; Anufriiev, S.; Jachnik, B. Conveyor Roller Diagnostics Using Acoustic Data from a Legged Robot. *Appl. Sci.* **2021**, *11*, 2299.

39. Cao, X.; Zhang, X.; Zhou, Z.; Fei, J.; Zhang, G.; Jiang, W. Conveyor Monitoring Using Suspension Inspection Robot. In *Proceedings of RCAR 2018*; pp. 657–661.

40. Staab, H.; Botelho, E.; Lasko, D.; Shah, H.; Eakins, W.; Richter, U. Robotic Vehicle System for Conveyor Inspection. In *ICM 2019*; pp. 352–357.

41. Garcia, G.; Rocha, F.; Torre, M.; Serrantola, W.; Lizarralde, F.; Franca, A.; Pessin, G.; Freitas, G. ROSI: Robotic Method for Belt Conveyor Inspection. In *ICAR 2019*; pp. 326–331.

42. Carvalho, R.; et al. UAV Framework for Thermographic Conveyor Inspection. *Sensors* **2020**, *20*, 2243.

43. Cui, Y.; Liu, S.; Liu, Q. Navigation and Positioning in Underground Mines. *J. S. Afr. Inst. Min. Metall.* **2021**, *121*, 295–304.

44. Mascaró, M.; Parra-Tsunekawa, I.; Tampier, C.; Ruiz-Del-solar, J. Topological Navigation in Tunnels. *Appl. Sci.* **2021**, *11*, 6547.

45. Androulakis, V.; Sottile, J.; Schafrik, S.; Agioutantis, Z. Navigation for Semi-Autonomous Shuttle Car Based on 2D LiDAR. *Tunn. Undergr. Space Technol.* **2021**, *117*, 104149.

46. Dang, T.; Tranzatto, M.; Khattak, S.; Mascarich, F.; Alexis, K.; Hutter, M. Graph-Based Subterranean Exploration Path Planning. *J. Field Robot.* **2020**, *37*, 1363–1388.

47. Bakambu, J.; Polotski, V. Autonomous System for Underground Mine Surveying. *J. Field Robot.* **2007**, *24*, 829–847.

48. Zare, M.; Battulwar, R.; Seamons, J.; Sattarvand, J. Wireless Indoor Positioning in Underground Mining: Review. *Min. Metall. Explor.* **2021**, *38*, 2307–2322.

49. Zwierzchowski, J.; Pietrala, D.; Napieralski, J.; Napieralski, A. Robot Position Adjustment via Vision + Odometry. *Appl. Sci.* **2021**, *11*, 4496.

50. Szrek, J.; Trybała, P.; Góralczyk, M.; Michalak, A.; Ziętek, B.; Zimroz, R. Evaluation of Localization Techniques in GNSS-Denied Mines. *Sensors* **2021**, *21*, 141.

51. Zietek, B.; Banasiewicz, A.; Zimroz, R.; Szrek, J.; Gola, S. Portable Environmental Monitoring System for Mines. *Energies* **2020**, *13*, 6331.

52. Bałchanowski, J. Modeling & Simulation of Mobile Robot with Self-Leveling Chassis. *J. Theor. Appl. Mech.* **2016**, *54*, 149–161.

53. Liu, Y.; Miao, C.; Li, X.; Xu, G. Deviation Detection of Belt Conveyor Using Deep Learning. *Complexity* **2021**, *2021*, 30.

54. Rocha, F.; Garcia, G.; Pereira, R.; et al. ROSI: Robotic System for Harsh Industrial Inspection. *J. Intell. Robot. Syst.* **2021**, *103*, 30.

55. Stachowiak, M.; Koperska, W.; Stefaniak, P.; Skoczylas, A.; Anufriiev, S. Procedures for Detecting Conveyor Belt Damage Using Legged Robot. *Minerals* **2021**, *11*, 1040.

56. Zhao, M.H. Patrol Robot System for Mining Belt Conveyor. In *IHMSC 2018*; Volume 2; pp. 1–3.

57. Solea, R.; Filipescu, A.; Nunes, U. Sliding-Mode Control for Wheeled Robot Trajectory Tracking. In *Asian Control Conference 2009*; pp. 1701–1706.

58. Nurmaini, S.; Dewi, K.; Tutuko, B. Differential-Drive Robot Control Using Linear Feedback. *IOP Conf. Ser. Mater. Sci. Eng.* **2017**, *190*, 012001.

59. Grisetti, G.; Stachniss, C.; Burgard, W. Improved Techniques for Grid Mapping with RBPF. *IEEE Trans. Robot.* **2007**, *23*, 34–46.

60. Thrun, S.; Burgard, W.; Fox, D. *Probabilistic Robotics*; MIT Press: Cambridge, MA, USA, 2005.

61. Labbé, M.; Michaud, F. RTAB-Map: Open-Source LiDAR + Visual SLAM. *J. Field Robot.* **2018**, *36*, 416–446.

62. **Raut, A.P.; Raj, A.; Patil, S.; Katkar, A.**
Internet Controlled Techrobot Using Raspberry Pi.*IJERT*, 2020.

63. **Thomas, R.O.; Rajasekaran, K.** Remote Monitoring and Control of Robotic Arm with Visual Feedback Using Raspberry Pi.*Int. J. Comput. Appl.* **92**(9), 2014.

64. **Singh, R.; Patel, M.** Mobile Surveillance Robot Based on IoT. *IJSART*, 2018.

65. **Patil, M.; Sane, V.**Web Browser Controlled Robot with Night Vision Camera. *IJERT*, 2019.

66. **Agarwal, S.; Nair, A.** Cloud Robot with Agriculture Using Raspberry Pi. *IJERT*, 2020.

67. **Archana, M.A.; Kumar, T.D.**Automatic Framework for Web-Controlled Surveillance Using Raspberry Pi. *J. Control Syst. Control Instrum.*, 2019.

68. **Mahajan, R.; Patidar, S.** Design and Implementation of Advanced Long-Range Real-Time Surveillance Robo Using Raspberry Pi Board. *IJERT*, 2023.

69. **Kavimandan, K.; Mangal, P.; Mehta, D.**A Surveillance-Based Interactive Robot.*arXiv preprint*, 2025.

70. Doucet, A.; de Freitas, N.; Gordon, N.Sequential Monte Carlo Methods in Practice. Springer, New York, NY, USA, 2001.

PERSONAL DETAILS

NAME: Ranjan Mishra

ENROLLMENT NUMBER: 231B258

EMAIL ID: ranjanmishra2028@gmail.com

ADDERESS: plot: 413/3901, Dream House, Bhubaneswar, Odisha

NAME: Rishabh Jain

ENROLLMENT NUMBER: 231B264

EMAIL ID: 2r10j5@gmail.com

ADDERESS: Maharaj Bada, Lashkar, Gwalior, Madhya Pradesh

NAME: Rashmi Dubey

ENROLLMENT NUMBER: 231b260

EMAIL ID: rashmi.dubey1010@gmail.com

ADDERESS: Village -Toosa , Singrauli, Madhya Pradesh