RoachBot: Autonomous Bug Detection and Capture for SJSU Dormitories

Suresh Ravuri

San Jose State University, Department of Computer Engineering

May 2025

Abstract

San Jose State University (SJSU) residence halls periodically report cockroach sightings, which negatively affect student health, comfort, and overall living conditions. Traditional chemical pest-control methods, although common, pose risks to indoor air quality and carry environmental drawbacks. To address this issue, we developed **RoachBot**, an autonomous mobile robotic system designed to detect, track, and pursue insect-like agents in real time within dorm-style environments. Leveraging LiDAR-based Simultaneous Localization and Mapping (SLAM), AprilTag visual detection, A* path planning, and finely tuned PID motion control, RoachBot demonstrates robust tracking behavior and adaptive environmental response. This report details the design, methodology, execution, and evaluation of RoachBot as deployed in a simulated SJSU dorm scenario.

I. Introduction

Maintaining a hygienic living environment is a critical operational requirement for university housing, particularly in dense residential communities such as those at San Jose State University. Cockroach infestations, even at low levels, can degrade resident well-being and introduce allergens. Human-driven pest removal is inefficient and unscalable, motivating the exploration of autonomous robotic alternatives.

The inspiration for RoachBot originates from the "Push-Button Kitty" episode of *Tom and Jerry*, in which a robotic predator systematically identifies and chases a fast-moving target. While whimsical in origin, this analogy guided the development of a practical robotic system capable of autonomously detecting and pursuing pests in real time. Because modeling real insect behavior is challenging, we employ a remote-controlled vehicle tagged with AprilTags to simulate unpredictable bug motion.

RoachBot integrates multiple subsystems—computer vision, SLAM, motion planning, and control—into a cohesive architecture capable of responding to dynamic indoor environments. The robot was tested in a convex arena approximating an SJSU dorm room with unobstructed floor layout, enabling a controlled evaluation of its chase performance.

II. Methodology

The RoachBot system is composed of several interdependent modules. Each contributes a specific capability required for detection, localization, path planning, and motion execution. The following subsections describe each component in detail, with emphasis on conceptual clarity and system integration.

A. Hardware Components

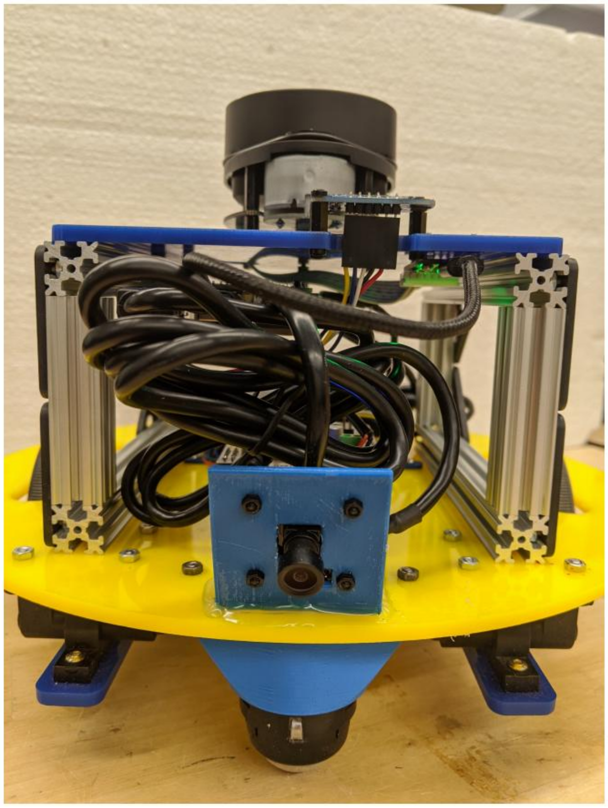
RoachBot is built on the MBot platform, which provides a compact yet powerful foundation for autonomous navigation tasks. The major components include:

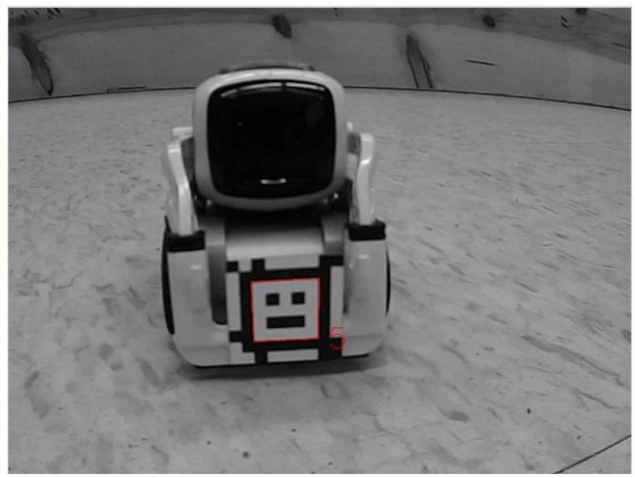
- **Raspberry Pi 3**: Handles camera input, AprilTag detection, and local preprocessing tasks.
- **BeagleBone Black**: Executes low-level motor control loops and interfaces with wheel encoders.
- **RPLIDAR A1**: Generates 360° distance scans used for SLAM-based mapping and localization.
- **Two USB Cameras (720p, 100° FOV)**: Mounted forward and rear to maximize visual coverage and reduce blind spots.
- **Differential Drive Motors**: Provide mobility; controlled through PID algorithms for stability.
- **Mock Bug Vehicle**: A remote-controlled car carrying an AprilTag, representing a bug moving unpredictably.
- **Arena Structure**: Wooden boundaries forming a convex environment to simulate simplified SJSU dorm constraints.

This configuration ensures that sensing, computation, and actuation operate cooperatively under real-time constraints.

B. Vision Tracking

Computer vision is the primary detection mechanism for identifying and locating the bug surrogate. AprilTags provide a simple, robust, and computationally lightweight method for tracking fast-moving objects.





1. **Detection Pipeline**:
   - Camera captures incoming frames.
   - AprilTag library extracts ID, orientation, and pixel-space coordinates.
   - Pose is computed relative to the camera frame.
2. **Coordinate Transformation**:
   Because SLAM uses a world-aligned map, detected bug positions must be transformed from the camera frame into the world coordinate frame. Calibration matrices derived from camera extrinsics enable this transformation.
3. **Dual-Camera Setup Advantages**:
   - Increases field of view, helping prevent bug loss.
   - Simplifies searching, as either camera can detect the target.

- Reduces time spent reorienting the robot.

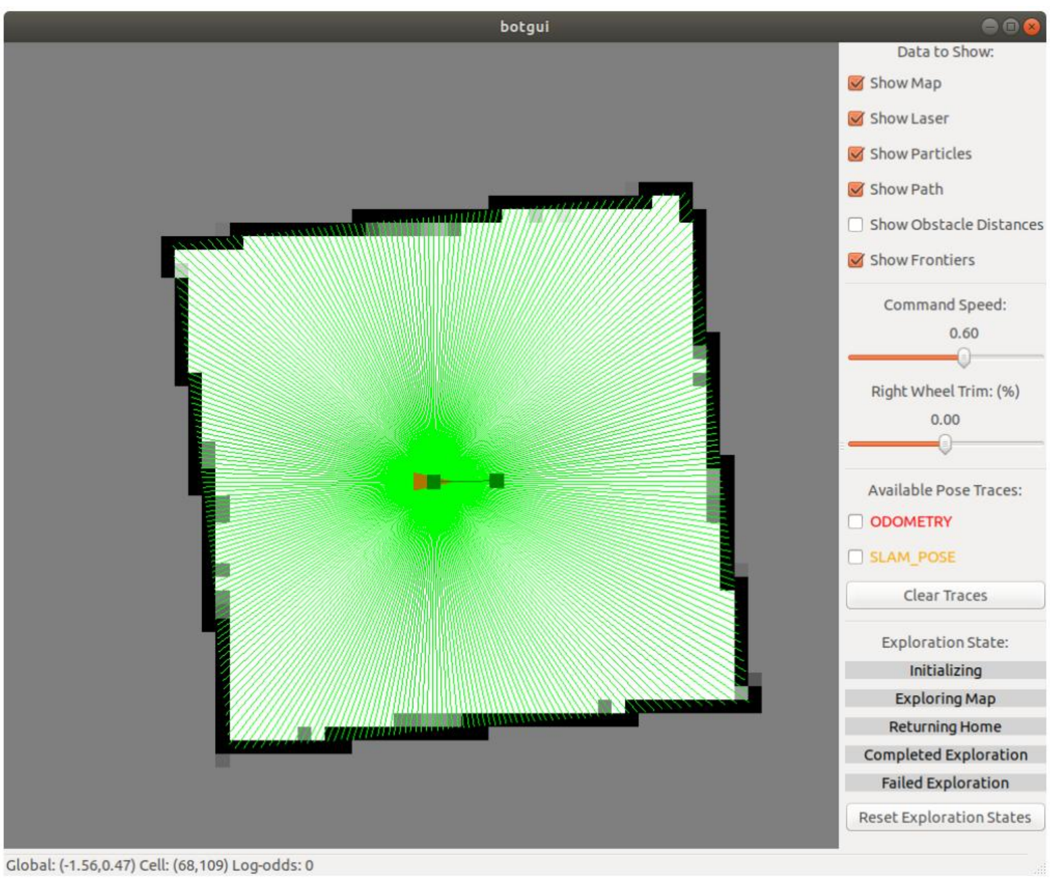
4. **Finite State Machine (FSM) Integration**:
   The vision system communicates with the FSM to drive state transitions between **Detect**, **Search**, and **Follow**, depending on whether the AprilTag is visible.

C. SLAM and Localization

SLAM allows RoachBot to simultaneously build a map and localize itself within that map. This is essential for reliable chase performance.

- **Map Generation**: LiDAR scans produce occupancy grids, identifying free space and boundaries.
- **Pose Estimation**: SLAM fuses LiDAR and odometry data to estimate the robot's positional coordinates and heading.
- **Environment Assumptions**: The system assumes a convex dorm layout with no obstructive furniture, simplifying localization and reducing computational demand.

Accurate SLAM ensures the robot does not drift away from the true environment and prevents wall collisions.



D. PID Motion Control

Precise motion requires careful adjustment of motor commands to ensure stability and responsiveness.

- **Proportional Term**: Reduces immediate error.
- **Integral Term**: Addresses accumulated bias or drift.
- **Derivative Term**: Smoothens motion and reduces oscillation.

Tuning aimed for a slightly under-damped response—fast enough to track the target but not so aggressive as to overshoot or create instability in tight dorm-like corridors.

E. Motion Planning Using A*

The A* algorithm generates the shortest viable path toward the moving bug while respecting environment boundaries.

- **Inputs**: SLAM occupancy grid, robot pose, bug pose.
- **Output**: Ordered set of path waypoints.
- **Dynamic Updating**: As the bug moves, new goal points are continually computed.

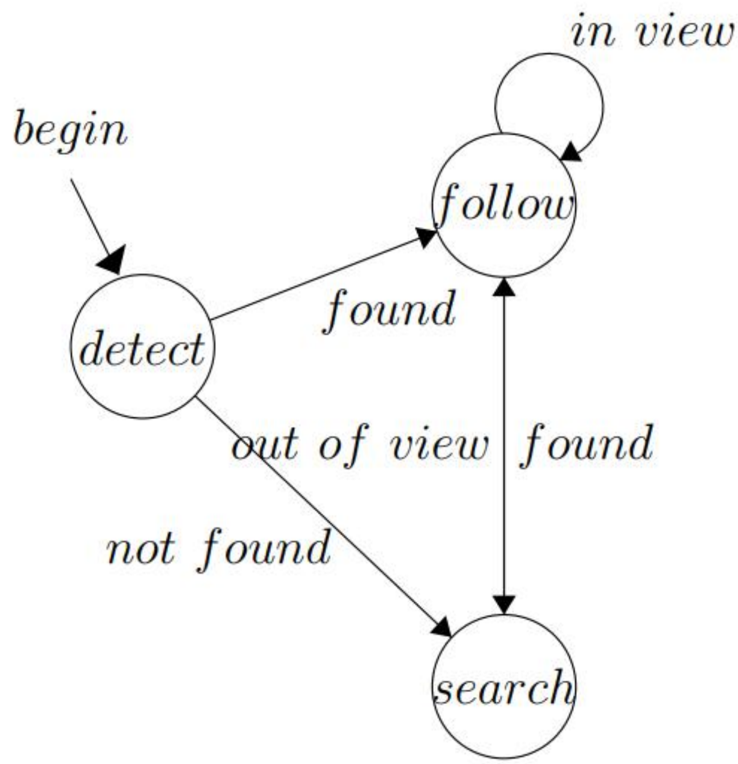
A* was selected for its simplicity, reliability, and suitability for grid-based maps typical of indoor environments.

F. Finite State Machine

The state machine governs behavioral transitions:

1. **Detect**: Search for AprilTag in current camera feed.
2. **Search**: Rotate in place when the target is lost to reacquire visibility.
3. **Follow**: Pursue the target while updating its location and dynamically recalculating motion paths.

The FSM ensures predictable operation and clear separation of perception and action workflows.


in view
begin
follow
found
detect
out of view
found
not found
search

III. Schedule Planning (Aug 2024 – May 2025)

The project followed an academic-year timeline aligned with SJSU's schedule:

Phase	Duration	Description
System Concept & Requirements	Aug 2024	Defined pest-control problem, constraints, and robot capabilities.
Hardware Assembly & Calibration	Sep 2024	Installed cameras, LiDAR, mounts; calibrated sensors; verified connectivity.
SLAM Development	Oct 2024	Integrated LiDAR with SLAM stack; tested mapping in simulated dorm layouts.
Vision Tracking & AprilTag Integration	Nov 2024	Implemented AprilTag detection; validated pose estimation.
FSM & Motion Control Development	Dec 2024	Built state machine and PID loops; executed basic chase logic.
A* Path Planning Integration	Jan 2025	Merged SLAM maps with A* planning for dynamic bug pursuit.
Full-System Integration	Feb 2025	Connected Python high-level logic with C++ motion planners via LCM.
Testing & Iterative Improvements	Mar–Apr 2025	Conducted chase tests; tuned PID; refined search behavior.
Final Demonstration & Documentation	May 2025	Presented project simulation for SJSU environment.

IV. Discussion

Our final robot successfully demonstrated the ability to track and follow a moving target along both linear and complex trajectories. The adaptive search mechanism allowed the robot to intelligently rotate in the direction the target exited the field of view, thereby reducing reacquisition time. A reference demonstration video illustrating similar behavior from earlier iterations of the system can be viewed at: https://www.youtube.com/@Suresh.Ravuri

The effectiveness of the RoachBot system is evaluated using the following criteria:

A. Ability to Detect Bugs

A robust bug-detection system must respond to various real-world conditions:

- Target initially visible within the camera frame.
- Target located within the SLAM-mapped region but not in view.
- Target temporarily obstructed or moving unpredictably.
- Target exiting and re-entering the field of view.

In all but the "hidden behind obstacle" scenario, RoachBot's detection and reacquisition logic performed reliably. Since our tests simulate a convex SJSU dorm environment, hidden-target cases were outside project scope.

B. Responsiveness to External Stimuli

The system demonstrated adaptability to sudden changes such as:

- Target removal from the scene.
- Sudden changes in target speed or direction.
- Temporary sensor occlusions.

The FSM reliably transitioned into **Search** mode when the target disappeared and returned to **Detect** or **Follow** modes upon reacquisition. This confirms that the robot maintains situational awareness.

C. Ability to Avoid Obstacles

Although designed for convex environments with minimal obstructions, RoachBot effectively avoided arena walls using SLAM-based localization. During testing, it did not exhibit unsafe collisions or erratic motion, ensuring viability for dorm-style layouts.

V. Possible Follow-Up Projects

Several enhancements could evolve RoachBot into a field-ready residential pest-control robot.

A. Functionality in Non-Convex Environments

Real dorm rooms contain furniture, occlusions, and multi-room layouts. Improving RoachBot for such environments would require:

- Obstacle-aware navigation.
- Patrolling behaviors instead of stationary spinning.
- More complex SLAM and exploration algorithms.

B. Removal of AprilTag Assistance

AprilTags simplify detection but are unrealistic for real insects. Future versions should integrate:

- Deep-learning-based object detection.
- On-device inference optimizations.
- Additional sensing modalities like thermal or infrared imaging.

C. Added Dimensionality

Current navigation is 2-D. Bugs on walls or elevated surfaces require:

- 3-D mapping.
- Multi-axis camera systems.
- Reengineered chassis or articulated arms.

These changes introduce significant mechanical, computational, and algorithmic complexity.

D. Arm Improvement

The initial Rexarm manipulator was removed due to motor faults and inconsistent feedback signals. Future designs could incorporate:

- A reliable multi-servo arm with torque sensing.

- A vacuum-based capture mechanism.
- Modular attachable end-effectors.

Such improvements are necessary for interacting with bugs in tight or elevated spaces.

VI. Conclusion

RoachBot demonstrates the feasibility of deploying a low-cost autonomous robot to detect and pursue insect-like agents in residential environments such as SJSU dormitories. By combining SLAM-based localization, AprilTag vision tracking, A* planning, and PID control, the system achieves real-time chase behaviors and adaptive target reacquisition.

While current functionality is limited to simplified convex spaces and AprilTag-marked targets, the project establishes a strong foundation for future expansion into realistic residential scenarios involving robotic perception, obstacle-aware navigation, and physical pest capture.

RoachBot demonstrates that a low-cost, autonomous robotic system can be effectively deployed in a dorm-like environment to chase and potentially remove pests. With future upgrades, such a system could serve real-world applications in pest control, smart homes, and beyond.