Swarm robotics for area coverage problems

Paper: ASurvey on Swarm Robotics for Area Coverage Problem

(algorithms-17-00003.pdf)

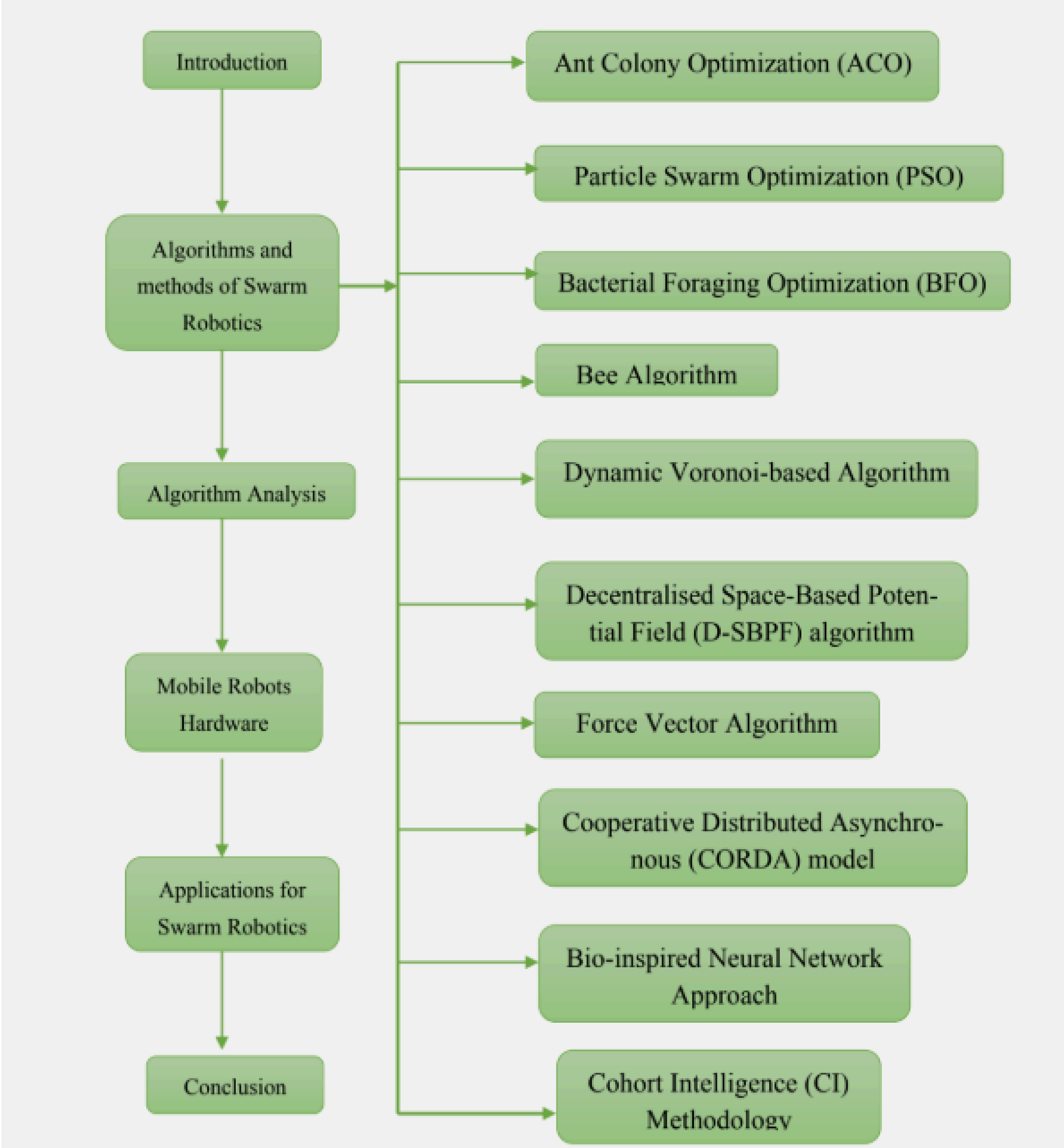

Figure 1. Conceptual framework of the work.

1. Introduction to Swarm Robotics and Area Coverage Problem

The **area coverage problem** is a vital research area that significantly benefits from **swarm robotics**. It involves obtaining or updating information within a specific spatial robotic domain. The main challenge is effectively covering an area.

Key applications where area coverage is essential include:

- **Exploration** (e.g., planetary exploration, uncovering unknown environments)
- **Surveillance** (e.g., monitoring expansive regions for intelligence or security)
- **Mapping** (e.g., creating collective maps of environments)
- **Foraging** (e.g., searching for food sources or targets)
- **Mine detection and demining**
- **Wildfire fighting**
- **Agriculture** (e.g., precision farming, pest management, broad-acre farming)
- **Manufacturing** (e.g., logistics, cleaning, maintenance)
- **Search and rescue**

Swarm robotics is a field within multi-robot systems and is an application of **swarm intelligence**. It is **bio-inspired**, stemming from observations of common animal behaviors where individuals act autonomously. The core idea is that each robot follows simple rules based on local sensory inputs and local communication with neighbors.

Swarm robotics offers several characteristics that make it suitable for various problem solutions:

- **Synergistic**
- **Distributed**
- **Robust**
- **Operating in real-time**
- **Universal**
- **Practical**
- **Optimality**
- **Reliability**
- **Scalability**

Research in swarm robotics encompasses aggregation, area coverage, target search, and cooperative handling.

2. Methodology of the Survey

This survey focused on research papers published between **2015 and 2022**. This timeframe was chosen due to:

- **Tremendous technical developments** in swarm robotics controllers, sensing technologies, and hardware capabilities.
- The emergence of swarm robotics as a **separate and significant field** within robotics.
- Advancements in **algorithmic frameworks** (optimization, coordination strategies, swarm intelligence systems).
- Transition from theoretical study to **real-world applications** in various domains.
- An **upsurge in academic publications, conferences, and group projects**.
- A period of **maturity** where scientists improved previous efforts and tested new approaches.
- Ensuring the reviewed articles are **up-to-date and relevant** to current trends.

Search Criteria:

- **Academic Databases**: Google Scholar, IEEE Xplore, PubMed, and ScienceDirect.
- **Search Queries**: "Swarm robotics," "area coverage," and similar topics.
- **Inclusion Criteria**: Publications between 2015 and 2022, focusing on swarm robotics-based area coverage problems (exploration, monitoring, mapping, foraging, etc.). Priority was given to **high-quality peer-reviewed journals and conference papers**, and only **freely available full texts** were included for accessibility and transparency.

Classification Process: Articles were categorized based on:

- **Major Use Case**: Exploration, task allocation, coordination, specialized applications (agricultural, surveillance, industrial tasks).
- **Algorithmic Methods**: Such as Particle Swarm Optimization (PSO) and Ant Colony Optimization (ACO).
- **Robot Coordination**.
- **Centralization**.
- **Obstacle Handling**.
- **Hardware Requirements**.

This classification aimed to help readers understand the various facets of swarm robotics, assessing each work's merits, shortcomings, and overall impact. Data extracted included title, authors, publication year, essential algorithms, application domain, coordination mechanisms, hardware details, strengths/limitations, and results presentation. Emphasis was placed on works with noteworthy methodology, experimental validations, or practical applications.

3. Algorithms and Methods for Swarm Robotics

Most algorithms in swarm robotics for area coverage are inspired by **biological entities' coordination**. Algorithms are broadly classified into **stochastic** and **deterministic**.

- **Deterministic approaches** use mathematical models relying on local optimum and gradient.
- **Stochastic techniques** use mathematical properties with minimal dependency on gradients and local optimums, often considered more user-friendly for optimizing robots. The choice of algorithm depends on the problem's nature.

3.1. Metaheuristic Algorithms—Swarm Intelligence

These algorithms are crucial for covering large areas, drawing cues from cooperative natural systems like ant colonies, beehives, and bird flocks, promoting decentralized decision-making.

- **Ant Colony Optimization (ACO)**:
    - Inspired by ants searching for food, depositing pheromones in their footpath. Ants follow trails with the highest quality/quantity of food.
    - Researchers adapt this by using **artificial intelligence robot-to-robot communication** as a substitute for natural pheromones.
    - **Applications**: Controlling individual robot behavior for **area coverage** using **decentralized control laws**. Studies analyze the effect of pheromone diffusion, evaporation, and noise on performance.
    - **Variants**:
        - **Cellular Automata Ant Memory Model (CAAM)**: For **foraging search tasks** in known environments, where robots communicate via **inverted pheromones** (repulsive force) and use **short-term memory** (inspired by Tabu Search) to avoid unnecessary explorations.
            - **Cellular Automata (CA):** The environment is discretized into a grid. Each robot moves from cell to cell based on a set of rules.
            - **Inverted Pheromones:** Unlike traditional ant algorithms that use **attractive pheromones**, CAAM uses **repulsive pheromones**. Robots leave a trail to **discourage others from revisiting the same area**, encouraging exploration.
            - **Short-term Memory (Tabu Search):** Each robot keeps a recent list (tabu list) of visited cells to avoid revisiting. This mimics **Tabu Search** used in combinatorial optimization to escape local optima.
        - **Ant Foraging with Adaptive Brownian Levy Flight Transitions**: Enhances search efficiency by dynamically switching between **Brownian motion** and **Levy flight** in the stochastic component.
            - **Brownian Motion:** Represents **short, frequent, random steps**. Suitable for **local exploration**.

- **Lévy Flight:** Represents **long jumps interspersed with shorter moves**, suitable for **wide-area exploration**.
- **Adaptive Switching:** The robot dynamically **switches between Brownian and Lévy** based on the search context or success rate.

- **Genetic Shared Tabu Inverted Ant Cellular Automata (GSTIACA)**: A hybrid method combining **genetic algorithms, inverted ant pheromones, and Tabu Search** for **surveillance tasks**. Inverted pheromones cause robots to drift apart, increasing coverage.
  - **Genetic Algorithms (GA):** Used for **optimizing** search paths or parameters (e.g., best route, switching thresholds).
  - **Inverted Ant Pheromones:** Robots **repel** each other by leaving negative pheromones, spreading out over the environment.
  - **Tabu Search:** Each robot remembers recently visited cells and avoids them.
  - **Shared Memory:** Robots may share information (e.g., visited cells) to improve coordination.
  - **Cellular Automata:** Environment discretization supports localized interaction rules and fast computation.

- **Particle Swarm Optimization (PSO)**:
  - Developed by Kennedy and Eberhart, simulating **birds' flocking behavior**.
  - A stochastic technique where functionality depends solely on the **objective function**, not differential objectives or gradients.
  - Each robot navigates terrain semi-autonomously while functioning as part of the swarm, benefiting from collective experience.
  - **Velocity and Position Update:**

    For particle *i :*

$$v_i(t+1) = w \cdot v_i(t) + c_1 \cdot r_1 \cdot (pBest_i - x_i(t)) + c_2 \cdot r_2 \cdot (gBest - x_i(t))$$

$$x_i(t+1) = x_i(t) + v_i(t+1)$$

  - **Applications**: Solving optimization problems in both **local and global space**.
  - **Variants**:
    - **PSO with Inertia Weight (exploration vs. exploitation), PSO with Constriction Factor (replaces inertia weight with a constriction**

coefficient χ to stabilize convergence), and Standard Particle Swarm Optimization (SPSO): Evaluated for **seeking electromagnetic sources** by adjusting to physical constraints and avoiding obstacles. Inertia Weight PSO often proves most convenient.

- **Robotic Darwinian Particle Swarm Optimization (RDPSO)**: Combined with **Probabilistic Finite State Machine (PFSM)** and **Depth First Search (DFS)** to decrease exploration time and enhance robot motion. Also applied for **search and rescue operations** for multiple targets.
- **PSO-Based Algorithm with Flexible Fitness Functions**: Assesses performance on maps with obstacles, using standard, random target (Euclidean distance), and distance from base functions.
- **Exploration-enhanced RPSO (E2RPSO)**: For **multi-target searching** via extra exploration and extensive area coverage, avoiding obstacles by adjusting **dynamic Inertia Weight** and incorporating top-down/bottom-up diversity.
- **PSO with Inverse Perspective Map (IPM) Transformation**: An effective target function method for **positioning soccer robots** with high precision without GPS, reducing computations.
- **Adaptive Exploration Robotic PSO (AERPSO)**: Detects multiple static or dynamic targets quickly through cooperation, avoiding local minima and exploring unexplored regions. Uses evolutionary speed, aggregation degree, and adaptive Inertia Weight for obstacle avoidance and improved search time.

- **Bacterial Foraging Optimization (BFO) Algorithm**:
  - Mimics **E. coli bacteria's foraging behaviors**, known for high ability to escape local optimum and faster convergence.
  - Position Update:

$$\vec{x}_i(t+1) = \vec{x}_i(t) + C(i) \cdot \frac{\Delta(i)}{\|\Delta(i)\|}$$

- $C(i)$: step size
- $\Delta(i)$: random direction vector

  - **Variants**:
    - **Bacterial Chemotaxis Optimization (B.C.)**: A decentralized algorithm for **target search and trapping**, where each robot acts as a bacterium,

using **Voronoi methods** to divide areas and compute chemotaxis direction. Effective for detecting and surrounding targets while avoiding obstacles.

$$\text{direction} = -\nabla f(x) + \text{repulsion from neighbors}$$

- **Bacterial Chemotaxis-Inspired Coordination Strategy (BCCS)**: For **coverage and aggregation**, using **chaotic preprocessing** for initial robot positions and a random factor for rendezvous. Shows high exploration ability.

- **Bee Algorithm**:
    - Inspired by honey bees' breeding, mating, and foraging behaviors.
    - Includes algorithms like
        1. **Honey Bee Optimization (MBO)**:
            a. Initialize positions (food sources).
            b. Evaluate fitness.
            c. Employed bees search around their source.
            d. Onlooker bees probabilistically select sources based on fitness.
            e. Scouts randomly explore new sources.
            f. Memorize best food source (solution).

        2. **Fast Marriage in Honey Bee Optimization (FMHBO)**:
            a. The Queen selects drones based on fitness. (Eg: In a robot path planning problem, a solution (path) with shorter distance and lower energy use will have **higher fitness**.)
            b. Probabilistic Acceptance
            c. Priority based ranking of drones
        3. **Honey Bees Mating Optimization (HBMO)**:
            a. **Mating flight:** Queen flies, mates with drones using:

$$P(mate) = e^{-\frac{\Delta f}{T}}$$

            b. **Spermatheca:** Stores multiple drone genomes (solutions).
            c. **Brood Generation:** New offspring are created via **crossover and mutation** between queen and drone solutions.

    d. **Worker bees:** Improve broods using local search (e.g., hill climbing).

  Queens evolve → mate with fit drones → generate broods → broods are improved → best solutions selected for next gen.

  - Operates on a **forward pass concept**, with a "queen" (single robot) guiding others to a goal with the highest solution.
  - **Applications**:
    - **Surveillance robotics** integrating an **autonomous telepresence robot** (human-in-the-loop) for foraging tasks, which improves efficiency and convergence, especially in dynamic, complex scenarios.
    - **Complex area coverage problems** with forbidden/threat regions, using **labor division phenomena** (threshold and activation–inhibition models) to respond effectively to sudden threats.
    - **Wet-cleaning rooms** in large buildings using cyber-physical robotic cleaners, employing a **global bee search algorithm** and a **local formation-building approach**. Also applicable to harvesting, deactivating radioactive substances, and disinfecting areas.

- **A Bio-Inspired Neural Network Approach**:
  - Uses **neural dynamics** to guide robot groups for **complete area coverage (CAC)** in dynamic environments.
  - Each robot treats others as moving obstacles. Paths are generated from the neural network's landscape and previous robot positions, avoiding global searches or cost functions.

- **Cohort Intelligence (CI) Methodology**:
  - Models candidates' behavior based on their interaction to achieve a common goal, with each candidate's behavior improved by observing others in the cohort.
  - Used for **search and rescue** by swarm robots, employing roulette wheel selection, median method, and **perturbation technique** to avoid getting stuck in non-convex obstacles. Applicable in various obstacle scenarios (No Obstacle, Stationary, Single/Multiple Dynamic Obstacles).

3.2. Classical Algorithms

These methods provide a solid basis for area coverage, often using **deterministic approaches** and **set parameters**, and may involve centralized control.

- **Dynamic Voronoi-Based Algorithm**:
  - A mathematical model for **area division** based on initial "seeds" (robots). Each robot covers its own Voronoi cell, ensuring points are closer to their corresponding robot.
  - **Dynamic Voronoi-based algorithm**: Divides the target area dynamically as robots move. Used with **modified Bacterial Foraging Optimization (MBFO)** to coordinate robot positions and target areas, enabling efficient searching by following concentration gradients.

- **The Decentralised Space-Based Potential Field (D-SBPF) Algorithm**:
  - A simple **decentralized method** to disperse robot teams for quick exploration and area coverage.
  - Represents space using an **extended occupancy grid** where cells are attractive (undiscovered) or repulsive (discovered). A non-monotonic field scale factor improves corner searching and avoids local minima.
  - Allows robots to **leave/join/rejoin** the team at any stage.

- **The Force Vector Algorithm**:
  - Designed for most swarm robotics applications with minimal requirements for area coverage.
  - Inspired by **physics-based interactions**, FVA treats each robot as a particle experiencing **virtual forces**.

    Robots compute a **resultant force vector** based on:

    - **Repulsion from nearby robots** (to avoid clustering or collisions),
    - **Attraction to unvisited areas** or goals (to ensure dispersion or target approach),
    - Possibly **repulsion from obstacles** (for safe navigation).
  - Simple to implement, effective, and considered an alternative to more complex algorithms, requiring simple robots with more capabilities.

Robot Behavior

    - Each robot **moves according to the net force vector**, calculated locally from its surroundings.

    - This makes the approach **fully decentralized** — no need for global maps, central control, or complex planning.

    - The simplicity enables **small, cheap robots** to perform useful behaviors, like evenly spreading over an area or following formations.

- **Cooperative Distributed Asynchronous (CORDA) Model**:
    - A primary computational model for swarm robots, closely approximating real-life situations.
    - Robots follow sequential cycles: wait, observe, compute, move (non-overlapping).
    - Applicable for covering areas **with and without obstacles**. Robots divide space into blocks, exploring or "painting" blocks based on obstacle presence.

- **Deployment Entropy with Potential Fields Strategy**:
    - Used for **persistent area coverage** by sensing swarm robots, partitioning areas into regions.
    - Aims for **uniformity of agents per region**. Shows good spread and growing sensor coverage when potential fields are used.
    - Applies attractive and repulsive fields between robots to achieve greater spread. Scalable and suitable for decentralized control.

- **A Self-Organising Area Coverage Based on Gradient and Grouping (GGC)**:
    - A new method for area coverage using simple robots with minimal computing or storage.
    - Enables **self-organization** to cover unknown task areas. Grouping allows parallel coverage, improving speed.
    - Shows superior performance in coverage concepts, completion time, and robustness. Beneficial for submillimeter swarm robots in micro-medicine.

- **Frontier-Led Swarming Algorithm**:

    - For exploration and coverage of **unknown environments** while maintaining formation for short-range communication.
    - Includes **swarm rules** for a close-knit appearance and **frontier search** for exploration/coverage.
    - Effective in cluttered environments with unknown obstacles.

4. Algorithms Analysis

The algorithms used in swarm robotics for area coverage are either metaheuristic or classical, chosen based on the problem and environment (known/unknown, with/without obstacles). Modern algorithms tend to be less computationally intensive than earlier, mathematically elaborate ones.

General Observations and Analysis from Table 1:

- **Ant Colony Optimization (ACO)**:
    - **Strengths**: Achieves global and local concepts, noise improves performance, provides abilities to improve other algorithms.
    - **Limitations**: Uncertainty in convergence time, probability distribution changes by iteration.
    - **CAAM**: Avoids unnecessary explorations with short-term memory, better robot distribution, local control for obstacles. Limitations include sequences of random decisions and high computation time.
    - **Ant foraging + adaptive Brownian Levy flight**: Improves area coverage, lowers communication/sensing constraints, increases coverage to a threshold. Lacks detailed parameter analysis.
    - **GSTIACA**: Advanced surveillance techniques, local obstacle control, integrates AI with natural computing. Not yet applied to real robots.
- **Particle Swarm Optimization (PSO)**:
    - **Strengths**: Relatively simple to implement, few parameters, fast/inexpensive computations, robust, escapes local optima, works without centralized unit if positions reachable.
    - **Limitations**: Requires more powerful robots for obstacles, increased swarm size for each environment.
    - **RDPSO**: Good navigation in optimal time, higher success range, faster exploration, decreased time with increased swarm size. Saturation occurs when swarm size increases beyond a particular level, affecting optimal time and cost.
    - **PSO-Based Algorithm**: Increases area coverage and target detection by sharing information, flexible fitness functions. Does not focus on robot aggregation, requires external position information (like GPS).
    - **E2RPSO**: Avoids local optima, comprehensive search, vital for multi-target search by balancing exploration/exploitation. May not detect all targets.
    - **PSO + IPM**: Simple implementation, reduced computational/memory consumption, increased speed, decreased input size, eliminates perspective effects, high accuracy. Possibility of getting stuck in local optima.

    - **AERPSO**: Avoids local minima, detects all targets, explores unexplored regions, obstacle avoidance using evolutionary speed/aggregation degree, improves search time, balances exploration/exploitation. Not yet applied to real robots.
- **Bacterial Foraging Optimization (BFO)**:
    - **B.C. + Voronoi**: Less vulnerability to local optimum, robustness to unexpected failure, effectiveness. Time-consuming initialization, not verified on physical robots.
    - **BCCS**: Better coverage via preprocessing, better exploration, fewer iterations, higher success rate, distributed system. Uncertainty in irregular environments.
- **Honey Bee Algorithm**:
    - **Strengths**: Low computation, robustness, scalability, adaptability, simple, flexible, broad applications, ease of implementation, human telepresence speeds convergence.
    - **Limitations**: New algorithms require new fitness tests, slow sequential processing, large objective function evaluation.
    - **Labour division phenomenon**: High ability to solve dynamic environment/area coverage, effective response to sudden threats. Lack of global communication, may not apply to all situations.
    - **Bee search + local formation-building**: Global solutions, high ability to solve area coverage, uses for orientation. Requires several strategies for robot motion controllers.
- **Bio-Inspired Neural Network Approach**:
    - **Strengths**: Reduces completion time, robustness, fault-tolerant.
    - **Limitations**: Not perfectly accurate.
- **Cohort Intelligence (CI)**:
    - **Strengths**: Robots won't get stuck in non-convex regions using perturbation.
    - **Limitations**: May need many techniques to support it in some situations.
- **Dynamic Voronoi-Based Algorithm + MBFO**:
    - **Strengths**: Escapes local optimum, quick search, saves energy, robot motion follows gradient and sensor control reactions.
    - **Limitations**: Consumes computation time, not verified on physical robots.
- **Decentralised Space-Based Potential Field (D-SBPF)**:
    - **Strengths**: Simple, uniform, decentralized, quick search, enhances searching with attractive/repulsive grids, robots can leave/join/rejoin.
    - **Limitations**: Decreased efficiency/speed with few robots, lower exploration performance for complex geometric maps.
- **Force Vector Algorithm**:
    - **Strengths**: Applies well to robot swarms with few requirements, effective area coverage, general solution, simple to implement.
    - **Limitations**: Constraints on used robots, not as reliable as other algorithms, a secondary solution.
- **Cooperative Distributed Asynchronous (CORDA) Model**:
    - **Strengths**: Popular and suitable computational model, reduces system cost, fault-tolerant.

    - **Limitations**: Robot velocities affect it under limited visibility, requires more powerful robots for obstacles.
- **Deployment Entropy with Potential Fields Strategy**:
    - **Strengths**: Good spread of agents, growing sensor coverage, scalability, decentralized, more effective uniform distribution, low computational complexity.
    - **Limitations**: Lack of persistence results, robots know their position but not others'.
- **Self-Organising Area Coverage Method + Gradient and Grouping**:
    - **Strengths**: Less completion time, low computational cost, robustness, parallel coverage, useful for submillimeter robots.
    - **Limitations**: Number of teams and robot coverage distance must be manageable.
- **Frontier-Led Swarming Algorithm**:
    - **Strengths**: High performance, no re-tuning needed for new environments, covers cluttered areas with unknown obstacles.
    - **Limitations**: Not able to track dynamic obstacles, does not search for optimal parameters.

The reliability of swarm robot performance is highly linked to **sensing technologies** like GPS, infrared cameras, and LiDAR, which are crucial for understanding and navigating environments.

5. Mobile Robots Hardware Used for Swarm Robotics

The type of hardware varies based on coverage needs, including cameras, controllers, actuators, and sensors. Each component performs a specific task to assist the swarm's functionality.

- **Sensors**: Essential for providing information about the environment, enabling mapping by detecting features like land, roads, paths, and obstacles. The **I.R. Proximity Sensor** is standard for obstacle detection.
- **Controllers**: Crucial for dictating movement.
    - **Centralized control**: A lead robot dictates the movement of the entire swarm.
    - **Distributed (decentralized) control**: Each robot plans and dictates its own movement.
- **Communication Devices**: Fused with controllers for efficiency. Examples include **wireless devices** like Wi-Fi, Bluetooth, infrared, and LED lights.
- **Power Supply**: Essential, especially for small robots, usually **lithium batteries** (5–25 V DC) providing consistent, high-density voltage.

Specific Robot Platforms:

- **GRITSBot**: Inexpensive microrobot for swarm robotics, featuring automatic sensors, autonomous recharging, wireless reprogramming, and managing cooperative robots. Used for R-shaped area coverage.

- **mROBerTO (milli-robot-Toronto)**: An open-source modular millirobot (16x16 mm²) with proximity, IMU, compass, ambient light, and camera sensors. Capable of formation control using IR emitter/detector, communicates via Bluetooth/ANT+, and uses an ARM processor with 256 KB memory for complex tasks.
- **TurtleBot3 Burgers**: Development of earlier TurtleBot versions, managed by ROS. A low-cost, adaptable platform programmable in MATLAB and Python, featuring a **360-degree LiDAR sensor** and suitable for motion-planning. Used for covering cluttered areas with unknown obstacles.
- **e-puck 2**: Hardware extension of e-puck, compatible with Raspberry Pi, featuring powerful sensors (WiFi, USB, RGB LED, long-distance sensor). Used for area coverage as an entrapping approach in communication-free environments, relying on camera and laser sensors for indirect communication. Extensions like Pi-puck allow Raspberry Pi Zero attachment.
- **Colias IV**: A micro-ground mobile robot (4 cm diameter) with a strong ARM processor, camera, two digital microphones, and massive connectivity. Designed for studying intensive computational embedded models, good for visuals in swarm robotics algorithms.
- **WsBot**: A tiny, low-cost swarm robot for smart factories, featuring intelligent agents and industrial teams. Communicates via Wi-Fi and used for area coverage in smart factories, avoiding obstacles, and improving manufacturing productivity without a physical warehouse.
- **HeRo 2.0 (Heathkit Educational Robot)**: A low-cost, open-source robot with a 3D-printed body and off-the-shelf components. Uses an Espressif ESP8266 microcontroller for motor control and sensor data, with built-in Wi-Fi for robust communication via TCP/IP protocol with ROS. Features two wheels and eight IR sensors for obstacle detection. Used for area coverage, avoiding collisions.

6. Applications for Swarm Robotics to Perform Area Coverage Tasks

Swarm robots are increasingly used in various fields due to their reliability in large open spaces and terrains, reducing operational costs and time.

- **Geology**: Three-dimensional imaging of the earth's surface, significantly reducing mapping time from weeks to hours.
- **Military**: Intelligence collection, deploying UAV swarms for simultaneous reconnaissance, enhancing readiness and threat response.
- **Oil Sector**: Estimating oil spillage levels and depths in open seas, providing quick damage assessment and control methods.
- **Agriculture**:
    - **Sustainable broad-acre agriculture**: Low-cost robots controlled by a centralized laptop for tasks like **dealing with resistant weeds**. Use low-cost cameras and positioning sensors for large-scale coverage while avoiding obstacles.
    - **Spraying large fields**: Cooperative strategies use local information for decision-making, guaranteeing each area is visited once.

- **Source Seeking**: Using PSO algorithms on mobile robots with XBee modules to seek electromagnetic sources in a defined area, avoiding static and dynamic obstacles.
- **Aggregation and Formation**: GRITSBot robots forming letters (e.g., 'R') in a testbed.
- **Environmental Mapping**: E-puck robots using random walk algorithms (Brownian motion, Levy walk) to create collective maps of environments, particularly effective in closed environments.
- **Art and Creativity**: Robotic painting systems where swarm robots distribute paint on a canvas according to user-defined color densities.
- **Industrial Cleaning**: Multi-robotic dirt-cleaning algorithms (Grid Divide Algorithm, A* Path Planning Algorithm) using iRobots to clean industrial environments, proving more efficient than a single robot.
- **Non-Convex Area Coverage**: Distributing mobile robots with omnidirectional sensors in areas with obstacles, deriving better algorithms for coverage and homing by using local knowledge and visibility-based approaches.
- **Hospital Management**: Swarm robots assigned tasks like biomedical waste management and floor cleaning, reducing risk to medical staff (e.g., during COVID-19 pandemic). Robots use indoor positioning systems, automatic disinfecting boxes, and cleaning systems.
- **Target Exploration**: Swarm crawler robots using Lévy flight strategy to find targets (e.g., infrared-emitting balls) in large indoor environments, connecting to base stations via wireless communication.

7. Discussion and Conclusion

The survey highlights the **diversity and adaptability of algorithmic approaches** in swarm robotics for area coverage, exemplified by ACO, MBO, and PSO. Researchers show inventiveness in tackling various problems.

Key Aspects from Discussion:

- **Algorithmic Approaches**: Differ significantly in focusing on exploration, task allocation, or hybrid systems. The choice often depends on whether obstacles are static or dynamic.
- **Sensing Technology Influence**: Reliability heavily depends on sensors like GPS, infrared cameras, and LiDAR for environmental understanding and navigation. The choice of sensing technology greatly impacts swarm adaptability and coverage.
- **Application-Specific Challenges**: Diverse practical fields (healthcare, exploration, mapping, agriculture) require tailored algorithms and coordination strategies due to unique needs and difficulties.
- **Evolution Over Time**: Research over 2015-2022 shows advancements in algorithms, hardware integration, and practical applications, indicating maturity and acceptance of swarm robots for complex challenges.
- **Persistent Obstacles**: Recurring limitations include **scalability, adaptability to new situations, and creating robust coordination mechanisms**. Addressing these is crucial for future research.

- **Hardware's Role**: Improved capabilities are evident in robots like GRITSBot, TurtleBot3, and e-puck 2, with complex control systems, high-tech sensors, and strong networking, enhancing efficiency and coverage.

In conclusion, swarm robots are increasingly utilized in various fields due to technological advancements. Naturally inspired algorithms like **PSO, MBO, and ACO** are among the most suitable and widely used for robotics in this domain. These algorithms are chosen based on the problem type and environment (known/unknown, with/without obstacles). A key emerging trend is **individual coverage**, where each bot interacts with the swarm to achieve the goal. Modern algorithms are less dependent on complex calculations than older ones, and their combination with diverse hardware (sensors, cameras, communication, controllers) expands applications in areas like **defense, medicine, geology, and business**. While progress has been made, continuous challenges remain regarding scalability, adaptation, and coordination strategies.