Dynamic Map Stitching in Swarm Robotics

Aditya Patil[1], Shreyas Chandolkar[2], Sohel Shaikh[3] & Pranit Kothawade[4]

[1]Student; SCTR's Pune Institute of Computer Technology, (Electronics and Telecommunication), Pune, Maharashtra, India, patiladitya1309@gmail.com
[2]Student; SCTR's Pune Institute of Computer Technology, (Electronics and Telecommunication), Pune, Maharashtra, India, sschandolkar@gmail.com
[3]Student; SCTR's Pune Institute of Computer Technology, (Electronics and Telecommunication), Pune, Maharashtra, India, sohel2002shaikh@gmail.com
[4]Student; SCTR's Pune Institute of Computer Technology, (Electronics and Telecommunication), Pune, Maharashtra, India, pranitk2002@gmail.com

Abstract:

This work aimed to develop and test a novel algorithm for autonomous swarm robotics mapping within a simulated and real-world environment. Initially, we created a custom Gazebo [1] simulation environment to facilitate swarm robotics testing, incorporating robot models and obstacles represented as walls. Our implementation consisted of three stages: two in the simulation environment and the third involving the real-world robot platform.

In the first stage, we executed the algorithm on static maps generated by the robots, saving a snapshot of each map locally and merging them to produce a global map. The algorithm converted .pgm files from gmapping to cv image format, performing feature extraction using the ORB (Oriented FAST and Rotated BRIEF) [11] algorithm. This enabled alignment of images through feature correlation, prioritizing one-to-one correspondences with the highest weights to ensure integration and fidelity to the underlying data.

In the second stage, we employed the ROS platform with Gazebo, where the algorithm continuously stitched maps generated by each robot in real-time. These maps, published as 3D occupancy grids [19] on a ROS topic, were merged at the same frequency, demonstrating the algorithm's capability to integrate multiple real-time data streams into a coherent global map.

The final stage involved testing the algorithm, validating its performance in a real-world scenario. Despite some discrepancies in alignment due to variations in correlation weights, the algorithm successfully produced visually coherent and accurate merged maps, highlighting its potential for real-world applications in swarm robotics.

Keywords: Swarm robotics, occupancy grid, Feature extraction, ROS, merged maps, vision

1 Introduction

Swarm robotics, intersecting robotics and collective behavior, involves the coordinated control of many simple robots working together in a decentralized, self-organized manner. Inspired by social insects like ants and bees, swarm robotics aims to mimic their collective intelligence for artificial robotic systems [2][3]. Applications span agriculture, search and rescue, environmental monitoring, and industrial automation, leveraging scalability, robustness, fault tolerance, and efficiency.

The work addresses the problem of enhancing mapping and navigation capabilities within swarm robotics using the Swarm Gradient Bug Algorithm (SGBA) [5]. Initially, the lack of a comprehensive 2D occupancy grid limited spatial mapping and storage, relying instead on exploratory navigation and Received Signal Strength Indication (RSSI) for positioning. The work aims to integrate a 2D occupancy grid [18], improving map accuracy and enabling dynamic task allocation based on unexplored areas.

To tackle this problem, extensive research was conducted on existing literature and the evolution of swarm robotics, drawing from the collective behaviors of social insects. The Research's homework involved developing a custom Gazebo simulation environment, implementing ORB-based [11] real-time map stitching, and validating the algorithm. Key technical components included inter-robot communication via ROS, sensor integration for environment perception, and dynamic task allocation using local mesh networks. The work's relevance to Electronics and Communication Engineering is underscored by its alignment with core subjects such as control systems, wireless communication, and embedded systems. By advancing traditional BUG algorithms and incorporating spatial deconfliction, SGBA ensures robustness and adaptability in complex environments.

This research highlights the theoretical foundations, practical applications, and technical advancements in swarm robotics, offering a comprehensive understanding of its scope, relevance, and future potential.

2 Proposed Methods

2.1 Communications

Effective communication is crucial when working with ROS-based robots, as it enables seamless data transfer and coordination among multiple systems. The Robot Operating System (ROS) utilizes a variety of communication protocols to facilitate this exchange of information. A prominent protocol employed in ROS is Transmission Control Protocol/Internet Protocol (TCP/IP), which provides a reliable and widely-adopted method for establishing connections and transmitting data. TCP/IP offers several advantages for ROS-based robot communication. Firstly, it ensures a stable and error-corrected data stream, minimizing the risk of packet loss or corruption during transmission. This is particularly important for time-sensitive applications, where the integrity of the data is paramount. Additionally, TCP/IP supports both point-to-point and multi-node communication, allowing for the creation of complex networks of interconnected robots and systems.

Furthermore, the ROS framework seamlessly integrates with TCP/IP, providing a set of predefined libraries and tools that simplify the process of setting up and managing these communication channels. This integration enables developers to focus on the core functionality of their ROS applications, rather than getting bogged down in the intricacies of the underlying network protocols. In addition to the TCP/IP-based communication, ROS also supports a master-slave architecture, where a central ROS master node coordinates and manages the communication between different ROS nodes. This master-slave model ensures that all ROS nodes are registered with the master and can discover each other's services and topics, enabling seamless collaboration and data exchange. The master node acts as a centralized hub, facilitating the routing of messages and the coordination of the overall ROS system. The combination of TCP/IP-based data transfer and the ROS master-slave architecture provides a robust and scalable communication framework for ROS-based robots. This multi-layered approach allows for the development of complex distributed systems, where individual robots or subsystems can communicate efficiently and reliably, while the ROS master ensures the overall coordination and integration of the entire robotics ecosystem.

2.3 System Design

Core Algorithm

1. Prompt the user to input a 1D array representing data points collected from sensors.
2. Convert the 1D array into a 2D occupancy grid matrix, where each element in the grid represents the occupancy status of a specific location.
3. Generate a local map for individual robots by extracting relevant portions of the occupancy grid corresponding to each robot's area of operation.

4. Collect all the local maps and iterate through them to extract common features such as pixel intensity, time stamp, and global pose associated with each occupied grid cell.
5. Plot a one-to-one correlation between the extracted features from all the local maps, visually representing their relationships.
6. Generate a 2D grid map superimposing based on the one-to-one correlation, possibly highlighting areas of high feature consistency or similarity across the local maps.
7. Pass the superimposed map to a global error correction module to decrease errors in terms of features and pose, adjusting the map to better align with the collective data.
8. Utilize the final global error correction data as feedback to refine the feature extraction process, potentially improving the accuracy of future map generations.
9. Generate the final map incorporating the refined data, providing a comprehensive representation of the environment based on the input sensor data and the collaborative mapping process.

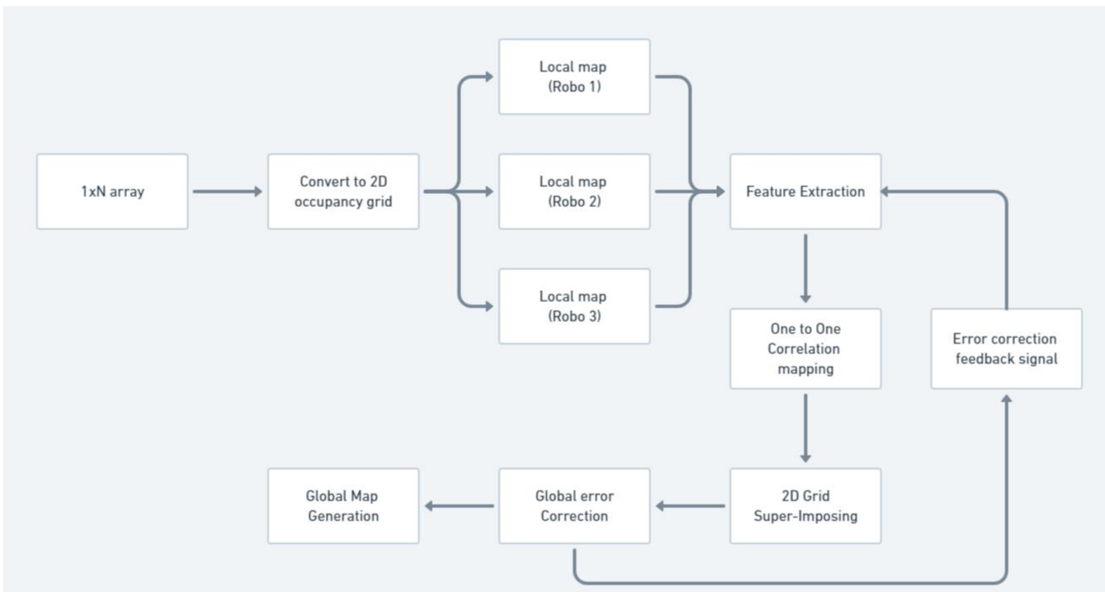

Fig. 1 Block Diagram

In the initial phase of our work, we undertook an extensive simulation of the Swarm and merge algorithm, a fundamental component of our system. This simulation encompassed the entire system, including the intricate aspects of mapping and navigation. To evaluate the system's robustness and effectiveness, we developed multiple map scenarios, each deliberately designed with diverse features. These maps were intentionally configured to include challenging obstacles, intricate loops, and precisely defined entry points. Our objective was to thoroughly test and validate the system's capability to navigate and map such complex environments.

Rviz[16] (abbreviated for 'ROS visualization') is a 3D visualization software tool designed for robots, sensors, and algorithms. It provides the capability to visualize a robot's perception of its environment, whether it's operating in a real-world or simulated setting. The primary objective of Rviz is to facilitate the visualization of a robot's state by leveraging sensor data to create a faithful representation of the robot's surroundings.

In the logical layer of our system, we harnessed the power of the ROS2 framework to orchestrate seamless integration. This integration was achieved through the creation of a launch file, which plays a pivotal role in initializing the simulation environment within the Gazebo[17] simulator. In this environment, we meticulously configured all the obstacles and deployed multiple robots at a designated starting point, each with a predefined desired direction. The real-time sensor data from these Raspberry Pi-powered robots was effectively channeled through Rviz, allowing it to capture and visualize the intricacies of the environment set up by Gazebo.

3. Results & Discussion

In the results section of our study, we present the outcomes of our algorithm that explores multi robot mapping, shedding light on their effectiveness and applicability in different contexts.

In this work's initial phase, we constructed a custom Gazebo simulation environment, a virtual space for swarm robotics testing. This digital environment included robot models, where obstacles were represented as walls. We utilized mapping agent gmapping by openslam organization. Each robot produces its own map using LidarScan data generated from YDLiDar Sensor and since the initial test being in the simulation environment, the odometer data is produced by Gazebo itself.

We divided our implementation in three stages, in the first two stages we utilized a simulation environment and in the third stage we tested the algorithm on a real world robot.

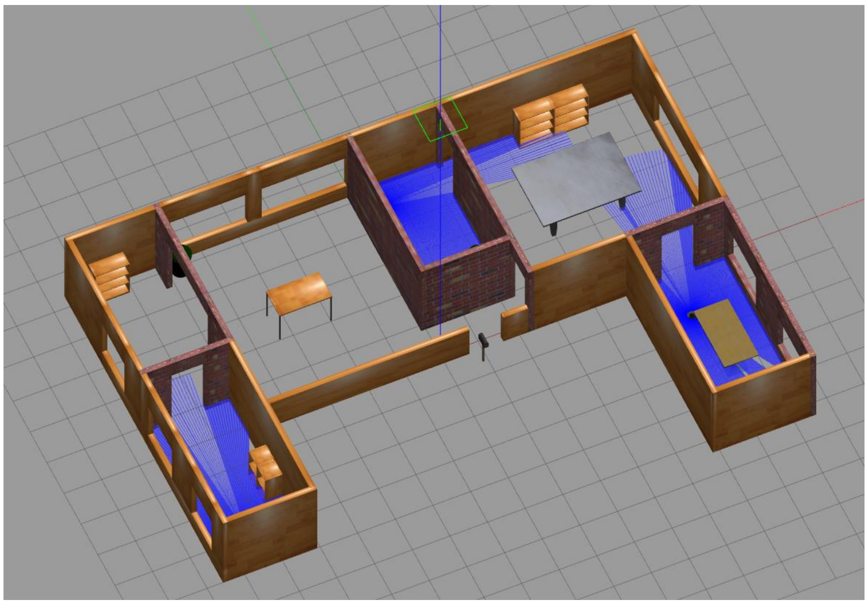

Fig. 2 Gazebo Environment to test the algorithm

In the first stage we executed the algorithm on a static map generated by the robots. A single snapshot of the map was saved on the local system and then the algorithm tries to merge the maps to produce a global map of the environment.

At the core level the algorithm initially converts the .pgm file generated by the gmapping package to cv image format. .pgm is a file format used to represent a occupancy grid in a single vector format. The file also contains a metadata tag which represents the height, width and other properties of the map such as occupancy threshold, origin and negate. Once the .pgm file is converted to cv image object the algorithm performs a feature extraction using orb detect and compute. orb detect and compute is an algorithm which extracts features from a given image object. Each feature is presented by using a key value pair, where the key parameter being the location of the feature with respect to origin of the image. The value being the description of the feature which describes the features like shading, contrast, hue, rapid color change.

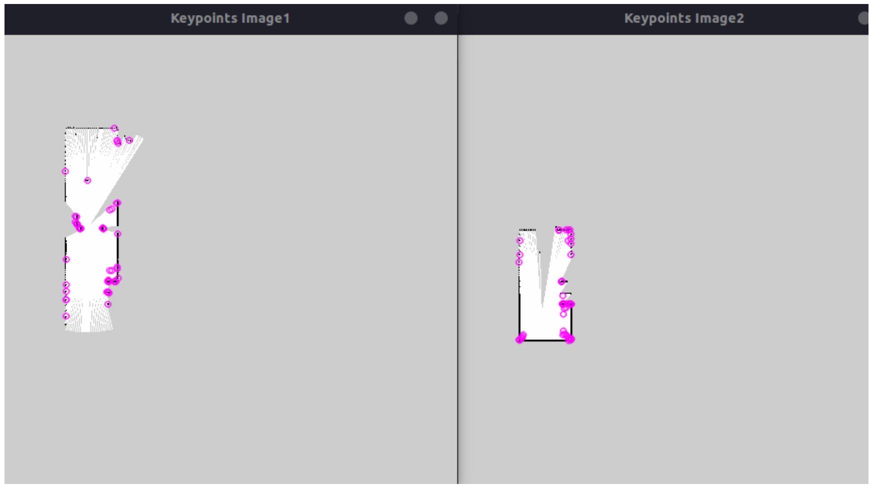

Fig. 3 Feature extraction within the opencv objects.

Employing the extracted features, the algorithm establishes a direct correlation between them, striving to align the images precisely according to their feature positions. When instances of overlap occur, precedence is granted to the one-to-one correspondence possessing the highest weight, thus resolving any conflicts that may arise. This meticulous approach ensures a seamless integration of the images, maintaining fidelity to the underlying data while mitigating discrepancies during the merging process.

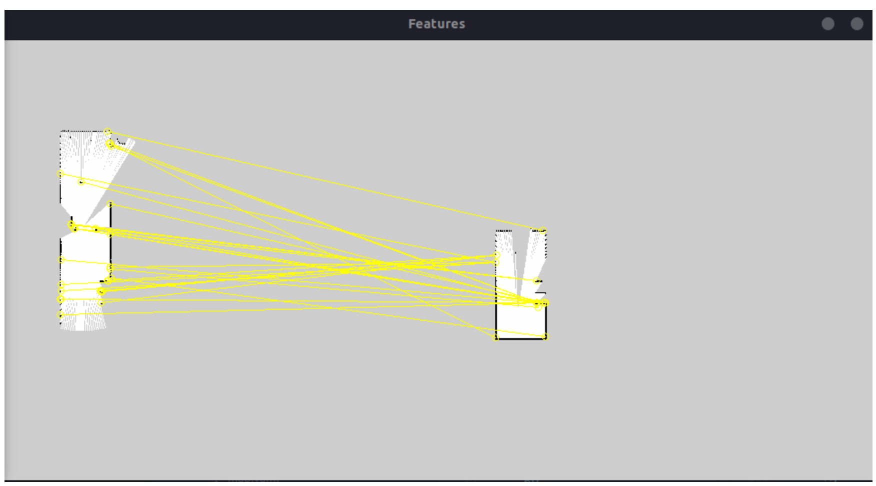

Fig. 4 One to one correlation of the features

The culmination of the stitched image arises from the intricate integration of all the individual maps provided to the algorithm. Despite meticulous efforts, subtle offsets may manifest between the merged map and its constituent parts. This discrepancy stems from variations in correlation weights, where not all feature alignments hold equal significance. Consequently, the merged maps may exhibit superficial alignment, necessitating a nuanced compromise in pose selection for overlap. This delicate balance ensures a harmonious and visually coherent final composition, wherein each element contributes to the overall narrative with precision and finesse.

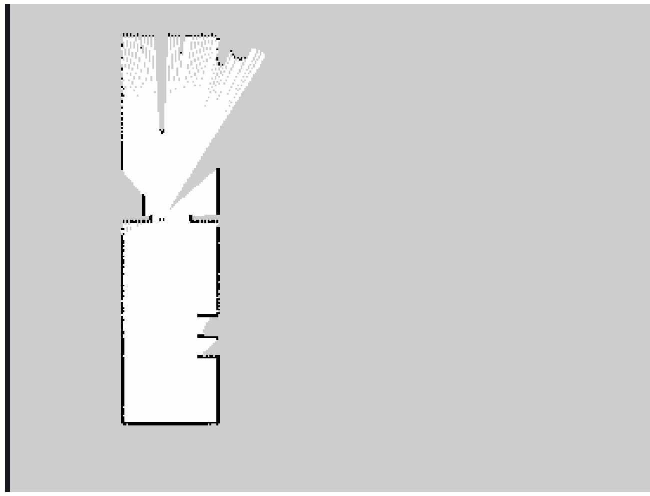

Fig. 5 Final stitched images

For the second part of the implementation we used the ROS platform along with the Gazebo where the algorithm continuously stitches the maps. Every robot creates its own version of the map. Again this map created is in the form of 3D occupancy grid published at a rate of about 20 Hz on a ROS topic with underlayered as a one to one tcp port connection. The algorithm executed the merging of all the maps with the same frequency as of the map generation.

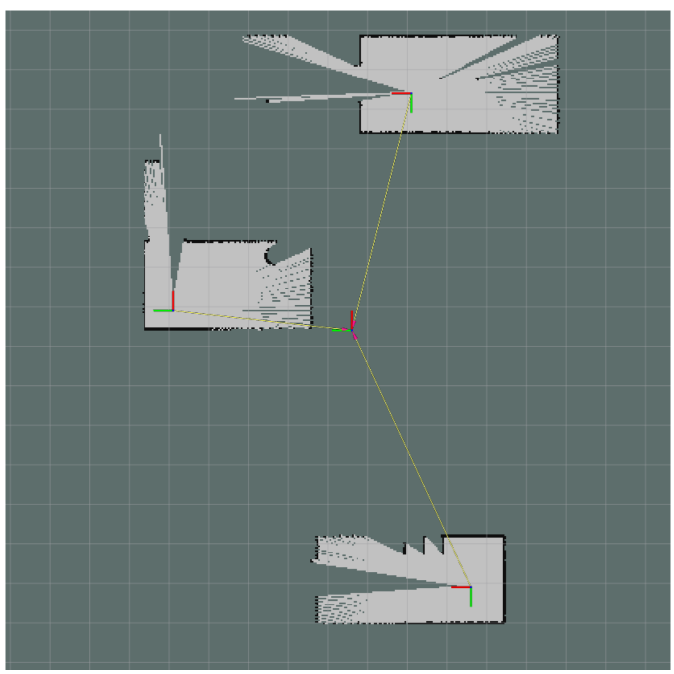

Fig. 6 Fused map in the simulation environment at a rate of 20 Hz

Having achieved success in the simulated environment through flawless execution of our algorithm, we advanced to the work's pivotal phase: implementing the algorithm on physical robots. This transition from simulation to physical implementation marks a significant milestone in our work journey, underscoring our commitment to practicality and real-world applicability.

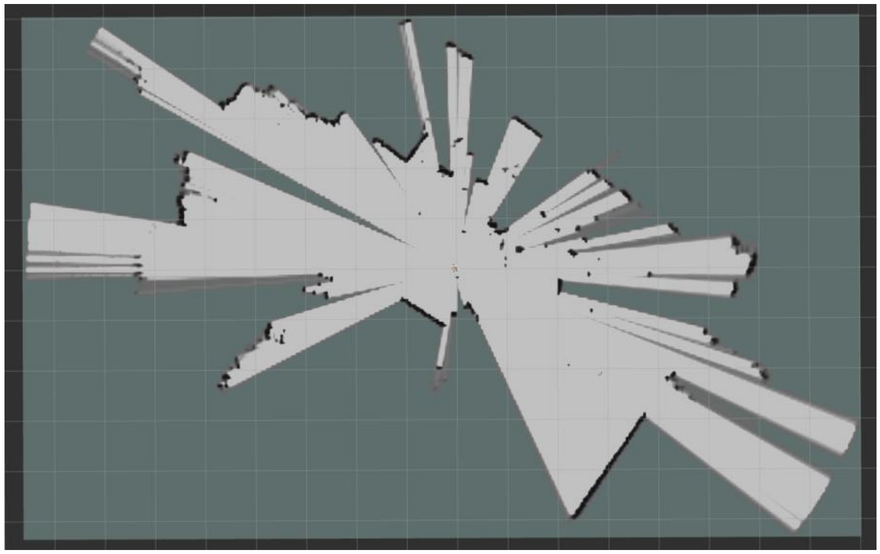

Fig. 7 Fused map of robots

We delved into a thorough time analysis to validate the outcomes of our algorithm. Beginning with a square room spanning 10 meters by 10 meters, we meticulously mapped its terrain. Then, we embarked on a comparative journey, employing two robots followed by three, and eventually four robots to traverse the same space. What emerged was a fascinating trend: the time investment for mapping the environment diminished quadratically as we augmented the number of robots in a linear fashion. This observation underscores the efficiency gains achieved through collaborative robotic efforts, illustrating a promising avenue for future exploration in automation and optimization.

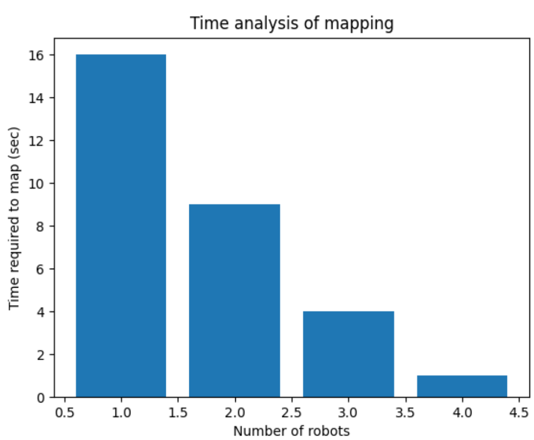

Fig. 8 Time analysis of mapping

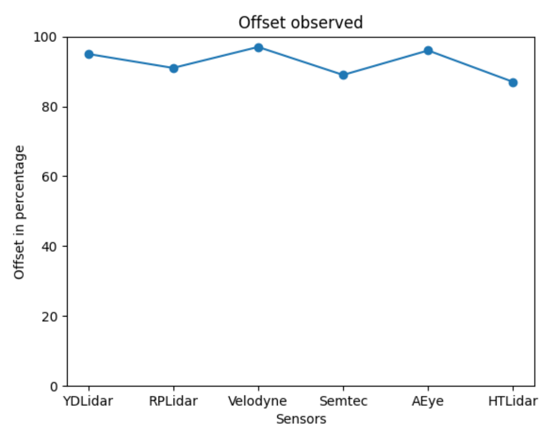

Fig. 9 Offset observed by different sensors

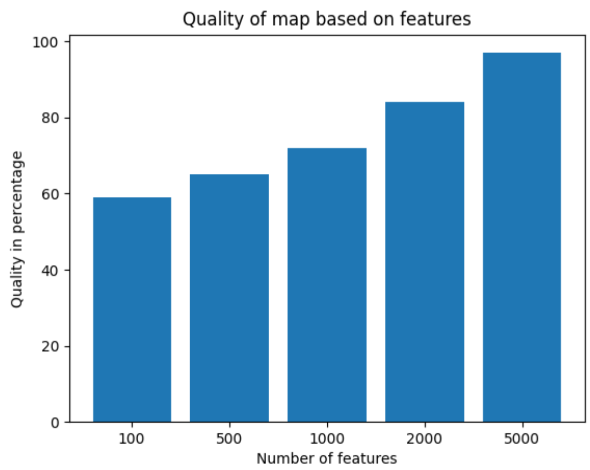

Fig. 10 Quality of map based on feature

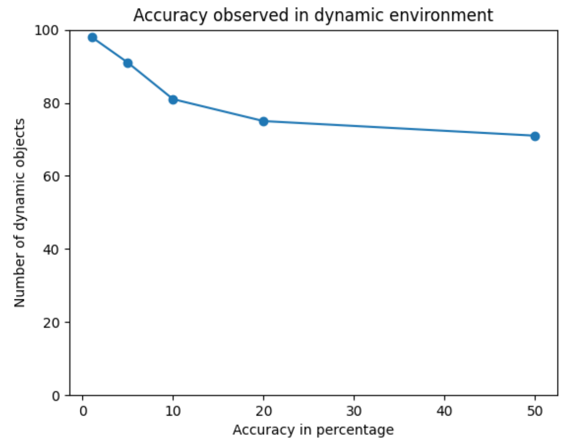

Fig. 11 Accuracy observed based on dynamic object

3.3 Tables and Figures

Table 1 Comparative analysis of Feature detection and description algorithms

Feature	SIFT [15]	SURF [12]	FAST [13]	BRIEF [14]	ORB [11]
Type	Feature Detection and Description	Feature Detection and Description	Feature Detection	Feature Description	Feature Detection and Description
Scale Invariance	Yes	Yes	No	No	Yes
Rotation Invariance	Yes	Yes	No	No	Yes
Speed	Moderate	Fast	Very Fast	Very Fast	Fast
Memory Usage	Moderate	Moderate	Low	Low	Low
Descriptor Size	Variable	64 or 128 dimensions	N/A (Descriptor Not Generated)	Variable (e.g., 128, 256, 512 bits)	N/A (Descriptor Not Generated)
Descriptor Calculation	Gradient Histograms	Wavelet Responses	N/A (Descriptor Not Generated)	Pixel Intensity Comparisons	Pixel Intensity Comparisons
Feature Matching	Nearest Neighbor with	Nearest Neighbor with Distance	N/A (No Descriptors)	Hamming Distance (Binary	Hamming Distance (Binary

	Ratio Test	Threshold		Comparison)	Comparison)
Keypoint Localization	Gradient-Based Refinement	Wavelet-Based Refinement	Circle-Based Test with Threshold	N/A (Descriptor Not Generated)	Harris Corner Measure and Refinement
Orientation Assignment	Gradient Direction	Wavelet Responses	N/A (No Orientation)	N/A (No Orientation)	Orientation from Intensity-Weighted Centroids
Feature Detection Method	Scale-Space Extrema Detection	Scale-Space Approximation	Accelerated Segment Test	N/A (Descriptor Not Generated)	Combination of FAST and Harris Measure
Feature Description	Histogram of Oriented Gradients (HOG)	Sum of Wavelet Responses	N/A (Descriptor Not Generated)	Binary Strings (BRIEF)	Binary Strings (BRIEF)
Strengths	Robustness to Scaling and Rotations	Speed and Efficiency	Speed and Efficiency	Memory Efficiency and Speed	Speed and Efficiency

4 Conclusion

Merge algorithms are sensitive to sensor measurement variations in unpredictable environments. To address this, improving communication and dynamically allocating tasks among other stitch algorithms optimizes environmental mapping. Recent SWARM algorithms demonstrate that six mobile robots can achieve a 92% success rate in exploration. Loop detection's efficacy relies on sensor and odometry data, with 40% of algorithms using it and 33% not requiring it. The remaining 27%, lacking specification or loop detection, risk repetitive behavior. Just 51% of studied merge algorithms are swarm-compatible, significantly enhancing exploration efficiency. A future avenue is developing swarm-compatible stitch algorithms.

Our work provided insights into swarm robotics, emphasizing decentralized control and self-organization's practical applications. Robust communication and task adaptability are essential in robotics.

Applications span various domains: agriculture (efficient crop monitoring, precision agriculture, autonomous harvesting), search and rescue (enhanced coordination and exploration in disaster zones), and industrial automation (optimizing logistics, surveillance, and material transport).

Acknowledgements

We wish to extend our sincere appreciation to the individuals whose unwavering support and guidance were instrumental in the successful completion of this work. In particular, We would like to express our heartfelt gratitude to our faculty mentor, Dr. R. C. Jaiswal, whose expertise, direction, and continuous support were the cornerstone of this research work.

We also want to acknowledge the invaluable contribution of Aditya Patwardhan, whose insights and industry knowledge brought a practical perspective to our work, enriching the work in numerous ways. Their involvement was truly invaluable.

Furthermore, We would like to recognize the significant influence of Professor K. C. Mcguire, whose research paper formed the foundation of our research work. Their work has been duly credited and referenced, and without it, our work would not have been possible.

We are deeply thankful for the collective efforts of these remarkable individuals, which were vital to the realization of this work.

References

[1] Bresson, G., Alsayed, Z., Yu, L. and Glaser, S., 2017. Simultaneous localization and mapping: A survey of current trends in autonomous driving. IEEE Transactions on Intelligent Vehicles, 2(3), pp.194-220.

[2] Cartwright, B.A.and Collett,T.S., 1983. Landmark learning in bees: experiments and models. Journal of comparative physiology, 151, pp.521-543.

[3] Lambrinos, D., Möller, R., Labhart, T., Pfeifer, R. and Wehner, R., 2000. A mobile robot employing insect strategies for navigation. Robotics and Autonomous systems, 30(1-2), pp.39-64.

[4] Borenstein, J. and Feng, L., 1996. Measurement and correction of systematic odometry errors in mobile robots. IEEE Transactions on robotics and automation, 12(6), pp.869-880.

[5] McGuire, K.N., De Wagter, C., Tuyls, K., Kappen, H.J. and de Croon, G.C., 2019. Minimal navigation solution for a swarm of tiny flying robots to explore an unknown environment. Science Robotics, 4(35), p.eaaw9710.

[6] Johansson, A. and Markdahl, J., 2023. Swarm Bug Algorithms for Path Generation in Unknown Environments. arXiv preprint arXiv:2308.07736. https://arxiv.org/pdf/2308.07736.pdf

[7] Ab Wahab, M.N., Nefti-Meziani, S. and Atyabi, A., 2015. A comprehensive review of swarm optimization algorithms. PloS one, 10(5), p.e0122827.

[8] Ng, J., Bräunl, T. Performance Comparison of Bug Navigation Algorithms. J Intell Robot Syst 50, 73–84 (2007). https://doi.org/10.1007/s10846-007-9157-6

[9] J. Borenstein, L. Feng, Measurement and correction of systematic odometry errors in mobile robots, IEEE Trans. Robot. Autom. 12 (6) (1996) 869–880, URL https://doi.org/10.1109/70.544770.

[10] Y. Zhu, T. Zhang, J. Song, X. Li, A new bug-type navigation algorithm considering practical implementation issues for mobile robots, in: Proc. IEEE Int. Conf. Robotics and Biomimetics, 2010, pp. 531–536, URL https://doi.org/10.1109/ROBIO.2010.5723382.

[11] Lowe, D.G., 2004. Distinctive image features from scale-invariant keypoints. International journal of computer vision, 60, pp.91-110.

[12] Bay, H., Tuytelaars, T. and Van Gool, L., 2006. Surf: Speeded up robust features. In Computer Vision–ECCV 2006: 9th European Conference on Computer Vision, Graz, Austria, May 7-13, 2006. Proceedings, Part I 9 (pp. 404-417). Springer Berlin Heidelberg.

[13] Rosten, E. and Drummond, T., 2006. Machine learning for high-speed corner detection. In Computer Vision–ECCV 2006: 9th European Conference on Computer Vision, Graz, Austria, May 7-13, 2006. Proceedings, Part I 9 (pp. 430-443). Springer Berlin Heidelberg.

[14] Calonder, M., Lepetit, V., Ozuysal, M., Trzcinski, T., Strecha, C. and Fua, P., 2011. BRIEF: Computing a local binary descriptor very fast. IEEE transactions on pattern analysis and machine intelligence, 34(7), pp.1281-1298.

[15] Rublee, E., Rabaud, V., Konolige, K. and Bradski, G., 2011, November. ORB: An efficient alternative to SIFT or SURF. In 2011 International conference on computer vision (pp. 2564-2571). Ieee.

[16] Kam, H.R., Lee, S.H., Park, T. and Kim, C.H., 2015. Rviz: a toolkit for real domain data visualization. Telecommunication Systems, 60, pp.337-345.

[17] Koenig, N. and Howard, A., 2004, September. Design and use paradigms for gazebo, an open-source multi-robot simulator. In 2004 IEEE/RSJ international conference on intelligent robots and systems (IROS)(IEEE Cat. No. 04CH37566) (Vol. 3, pp. 2149-2154). Ieee.
[18] Liu, Z., Chen, D. and von Wichert, G., 2012, May. 2d semantic mapping on occupancy grids. In ROBOTIK 2012; 7th German Conference on Robotics (pp. 1-6). VDE.
[19] Marín-Plaza, P., Beltrán, J., Hussein, A., Musleh, B., Martín, D., de la Escalera, A. and Armingol, J.M., 2016, February. Stereo vision-based local occupancy grid map for autonomous navigation in ros. In International Conference on Computer Vision Theory and Applications (Vol. 4, pp. 701-706). SciTePress.