Reconnaissance Mission: Development of an Algorithm for Indoor Localisation System with Collaborative Multi-Robot

Md. Masum Billah, Raisuddin Khan, Mohiuddin Ahmed and A.A. Shafie, *Member, IAENG*

***Abstract*— An algorithm for undertaking indoor localisation of multi-robot reconnaissance mission through collaboration has been presented in this paper. A set of communication techniques, an algorithm for sharing behavior for positioning of agent robots, and appropriate attitude which is useful for a group of robots to work together, are developed for collaborative reconnaissance. Two hexapod robots equipped with inertial sensors like accelerometer and rate gyro are chosen for multi-robot navigation in indoor environment for their advantages over the traditional wheeled robots. The sharing behavior and coordination of motion are verified experimentally through indoor localization. This localization system is applicable in such places where GPS signal is unavailable to identify the agent location.**

***Index Terms*—Reconnaissance, collaborative, localisation, multi-robot.**

I. Introduction

RECONNAISSANCE and surveillance are important activities for both military and civil organizations. Rescue missions for disaster survivors, illicit drug raids, and responses to chemical or toxic waste spills are just some of the examples that require reconnaissance. Automatic security and surveillance systems typically use sensors like cameras in fixed locations, either connected ad hoc or, increasingly through the shared communication lines of intelligent buildings [1]. These may be portable to allow for rapid deployment but still require human intervention to reposition when necessary. This shortcoming is exacerbated in cases in which the surveillance team does not have full control of the area to be investigated, as happens in many law-enforcement scenarios.

Typically, mobile robot behavior such as navigation, map building and estimation of own position is very important for autonomous reconnaissance mission [2]. For most mobile robot applications, GPS is used for positioning of the robots [1]. However, GPS might not be able to calculate position accurately when GPS signal or wave are disrupted, for example, when the mobile robot installed with GPS receiver is moving inside a tunnel, skyscraper, around tall buildings and in such other places. Inertial sensors like gyro and accelerometer are expected to help solve this problem [3]. The advantages of the inertial sensors are low cost, and simple to implement in most computer and mobile robot navigation systems. Also, the inertial systems can even be used on unpaved road without initial knowledge of the traveling environments (unknown environments) [4]. It is because of the fact that inertial systems use only internal sensors for its own position estimation.

A new method of localizing hexapod robots equipped with inertial sensors in indoor environment is presented in this paper. This new method can overcome the problems of the existing methods for undertaking reconnaissance.

Manuscript received March 18, 2012; revised April 13, 2012. This work was supported in part by the MOSTI, MOHI and RMC, IIUM.

Md. Masum Billah is with the Department of Mechatronics Engineering, Faculty of Engineering, International Islamic University Malaysia (phone: 603-61963384; e-mail: masum.uia@gmail.com).

Assoc. Prof. Raisuddin Khan is with the Department of Mechatronics Engineering, Faculty of Engineering, International Islamic University Malaysia (e-mail: raisuddin@iium.edu.my).

Assoc. Prof. Mohiuddin Ahmed is with the College of Computer Science and Information System, Zazan University (e-mail: ahmed255555@gmail.com).

Assoc. Prof. A.A. Shafie is with the Department of Mechatronics Engineering, Faculty of Engineering, International Islamic University Malaysia (e-mail: aashafie@iium.edu.my).

II. Algorithm's for Indoor Localization System

The proposed algorithm in this research is based on the fusion of inertial sensors for mobile robot dead-reckoning system. Fusion of the gyro sensor and accelerometer is implemented on a hexapod robot to recognize its own position. Method of positioning and localization used in this research are discussed in the following sections.

A. Positioning Method

The method of positioning used here is called error model method where each sensor measures the accumulated errors and compare it to the robots own position. The main features of this proposed method are:

1) Data received from the gyro and accelerometer is stored in a database.

2) A function then uses the data from the database for calculating the position (x_a, y_a) of the robot and the orientation angle, θ_g with respect to reference pose.

3) A function that monitors all error bounds to conform to data authentication.

Fig.1 shows the scheme for calculating position and orientation of a single robot.

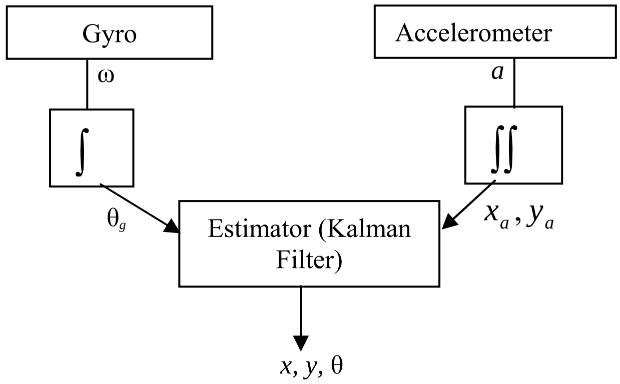

Fig. 1. Overview of Sensor Fusion System.

In Fig.1, ω, represents for angular velocity data from gyro sensor and *a,* acceleration data from accelerometer. When the robot moves to a new position, the new position data is updated in the database.

B. *Implementation of Dead-Reckoning System*

Dead-reckoning system should be designed to be able to minimize position and orientation errors. This can be accomplished by meticulously modeling sensor errors and by efficient design of a filter. Here we have implemented an indirect Kalman filter that combines all sensors information as error model.

1) *Error Model for Gyro Sensor*

Gyros and accelerometers measure rate of rotation and acceleration respectively. Measurements from gyro are integrated once to yield pose and that from accelerometer integrated twice to yield position. Inertial navigation systems have the advantage that they are self-contained, that is, they do not need external references. However, inertial sensor data drifts with time due to integration of data to yield position. The heading angle from a gyro sensor with bias drift is represented in Equation (1).

$$\begin{aligned} \theta_g(k+1) &= \theta_g(k) + \omega(k).\Delta k + Bb_g(k) \\ \hat{\theta}_g(k+1) &= \theta_g(k) + (\omega(k) + \delta\omega(k)).\Delta k + \hat{B}_g(k) \\ \hat{B}_g(k) &= Bb_g(k) + \delta B_g(k) \\ \hat{\theta}_g(k) &= \theta_g(k) + \delta\theta(k) \end{aligned} \tag{1}$$

Where, the hat symbol stands for estimated values and δ for each error. $\theta_g(k)$ is the true heading angle of the gyro sensor, $B_g(k)$ is the gyro bias drift and $Bb_g(k)$ is the theoretical gyro bias drift.

An equation for the heading angle error of the gyro sensor is obtained from Eq. (1), as shown in Equation (2).

$$\begin{aligned} \delta\theta_g(k+1) &= \delta\theta_g(k) + \delta\omega(k).\Delta(k) + \delta B_g(k) \\ \delta B_g(k+1) &= \delta B_g(k) + w(k) \\ \delta\omega(k+1) &= \delta\omega(k) + w(k) \end{aligned} \tag{2}$$

Where, w(k) is a system noise.

2) *Error Model for Accelerometer*

Same as gyro sensor, accelerometer also suffers from extensive drift with time due to the double integration of the acceleration data. The mobile robot position from an accelerometer with bias drift is represented by Equation (3).

$$\begin{aligned} v(k+1) &= v(k) + a(k).\Delta(k) + Bb_a(k) \\ x_a(k+1) &= x_a(k) + v(k)\cos\theta(k).\Delta k \\ y_a(k+1) &= y_a(k) + v(k)\sin\theta(k).\Delta k \end{aligned} \tag{3}$$

Where, $Bb_a(k)$ is a theoretical bias drift and *v*(k) is a true velocity of the mobile robot.

A position error equation for the accelerometer is obtained from Equation (2) and Equation (3), as shown in Equations (4) and (5).

$$\begin{aligned} \hat{v}(k) &= v(k) + \delta v(k) \\ \hat{x}_a(k) &= x_a(k) + \delta x_a(k) \\ \hat{y}_a(k) &= y_a(k) + \delta y_a(k) \\ \hat{B}_a(k) &= Bb_a(k) + \delta B_a(k) \end{aligned} \tag{4}$$

$$\begin{aligned} \delta v(k+1) &= \delta v(k) + \delta a(k)\Delta k + \delta B_a(k) \\ \delta x_a(k+1) &= \delta x_a(k) - v(k)\sin\theta(k)\Delta k.\delta\theta(k) + \cos\theta(k)\Delta k.\delta v(k) \\ \delta y_a(k+1) &= \delta y_a(k) + v(k)\cos\theta(k)\Delta k.\delta\theta(k) + \sin\theta(k)\Delta k.\delta v(k) \end{aligned} \tag{5}$$

Where, $B_a(k)$ is an accelerometer bias drift.

3) *Error minimization using Kalman Filtering*

An indirect feedback Kalman filter, [5] is used in this research as the state equations of the system. The state equation is shown in Equation (6) for the above error models.

$$x(k+1)=A(k)x(k)+w(k) \tag{6}$$

C. *Localization Algorithm*

In this research hexapod robots are employed for undertaking a reconnaissance mission. Each robot is equipped with one gyro sensor, one accelerometer, and four push button switches. The gyro and the accelerometer are placed on the body of the hexapod. One of the legs is then attached to the body at that junction. Out of the four push button sensors two are placed on this leg, one at the tip of the end link and the other at the knee as shown in Fig. 2. The remaining two push buttons are put on the adjacent leg in a similar way.

ISBN: 978-988-19252-9-9
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

Fig. 2. Push button attached to the leg.

Blue tooth module connected with a controller to transmit sensor data to a PC is placed on each robot. The entire system consists of a wireless network based on the blue tooth protocol. A coordinator node Bluetooth module connected to a personal computer is used to create and hold the network, acquire and process data. In this work, mobile nodes (hexapod robot) used for localisation, and coordinator node used to hold the network and collect data. Software is then developed and placed in the PC for managing the network and saving data in the data base.

Localisation of the hexapod robot is the first important step of the reconnaissance mission. For this purpose the position computation in the localisation algorithm is triggered by the coordinator node, which then sends acknowledgement to the mobile nodes. On receiving the message, each mobile node calculates the Received Signal Strength Indicator (RSSI) and Link Quality Indicator (LQI) values and sends them to the coordinator node. RSSI values from the mobile nodes are then processed to estimate the position of mobile node, and then the coordinator sends another acknowledgement for the next estimation. RSSI values from mobile nodes are converted from decibel meter (dBm) to centimeter according to the Hata model [6] using Equation (7):

$$P(d) = P_0 - 10.n_p.\log\tfrac{d}{d_0} \qquad (7)$$

Where, P(d) is the measured RSSI value in dBm, P_0 is the received power in dBm measured at reference distance d_0, n_p is the path-loss exponent, and d is the distance between the mobile nodes.

The flowchart shown in Fig. 3 shows the localisation algorithm.

III. Collaborative Behaviors

The hexapod robot shown in Fig. 4 was designed for cooperative operation. It was controlled by a microprocessor, a suite of sensors, and robots communication kit. In this experiment two hexapod robots were used.

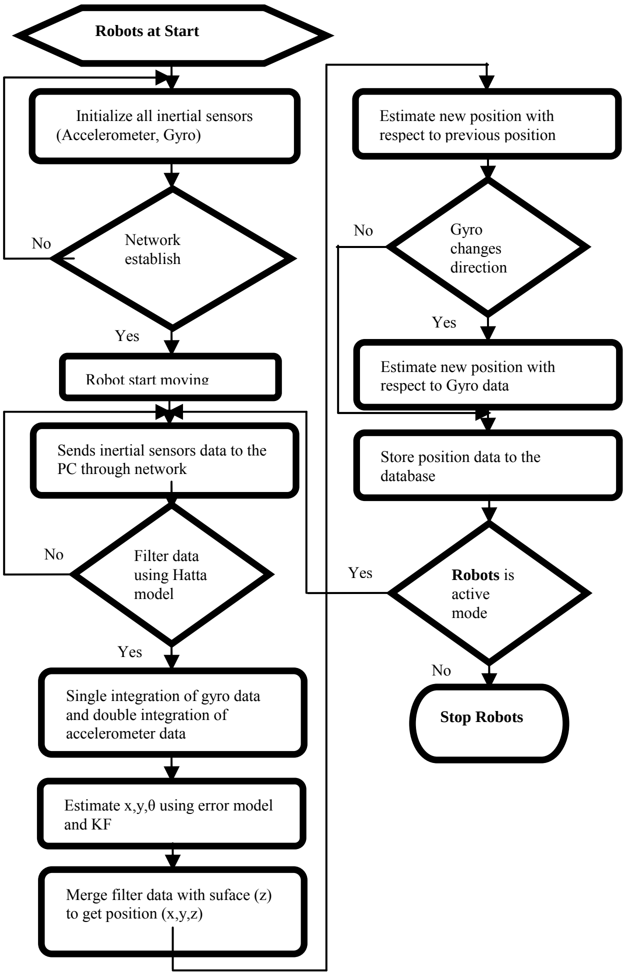

Fig. 3. Flowchart for robot localisation model.

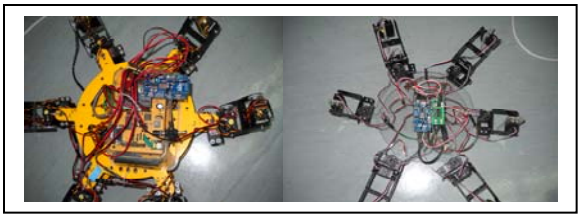

Fig. 4. Hexapod robots for co-operative operation.

A. *Sensors*

The hexapod has a large sensory suite, including bump sensors and the wireless communication system. All of the algorithms in this research use only the bump sensors, and the communication system.

The exterior shell of the hexapod legs are attached with bump sensor as shown in Fig. 5. It can detect ditch as well as hill type obstacles when moving on a plane.

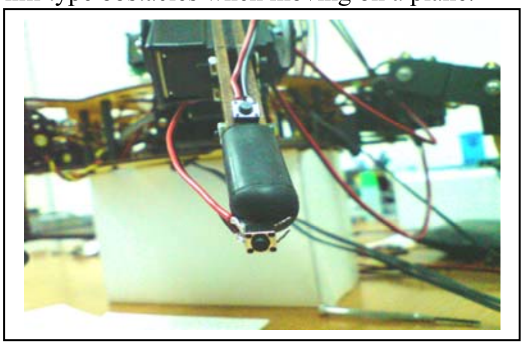

Fig. 5. Hexapod robot bump sensor configuration.

ISBN: 978-988-19252-9-9
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

B. *Wireless Communication System*

The wireless communication system allows each robot to communicate with its neighbors. The blue-tooth module for wireless communication system is used for communication between robots. One wireless transceiver module was mounted on each robot. Data can be transmitted from these transceivers independently. The transceivers allow the robots determine neighbor's position and behavior from the signal strengths of messages that are received within 2 cm to 100 meter separation distance.

The system has a maximum range of 100 meters, but is typically run at reduced power levels to limit the effective range to about 60 meters. The blue-tooth communications system runs at 115.2 kbps. A start-up board handles all the encoding, transmitting, receiving, and decoding. Data integrity is quite good over point-to-point communications in between the robots.

The communication process are shown in Fig. 6.

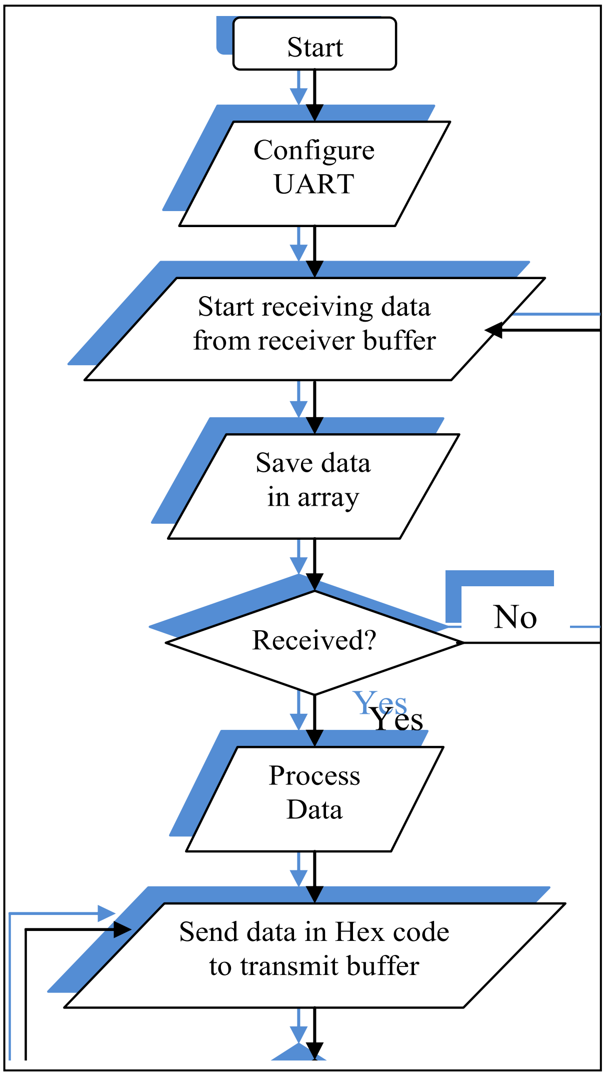

Fig. 6. Communication process.

IV. Results and Discussions

The reference and active robots were placed on the table with a separation distance of 50 cm. Following applications were done by communicating with wireless signal.

A. *matchHeadingToRobot*

The matchHeadingToRobot behavior uses both the bearing and orientation of the reference robot to direct the active robot to face in the same direction. It uses the orientToRobot behavior.

In Fig. 7, the front hexapod is the reference robot and the rear hexapod is matching its heading to it. This behavior is useful for making formations of robots. When a long-range wireless beacon is the heading reference this behavior can be used to move the robots along a global heading. The picture shows the robots are moving, and then they change their direction accordingly.

Fig. 7. Multi-Robot in matchHeadingToRobot behavior.

Step:

Rotate such that orientation is the same as the target robot

Minimize error and move to the goal orientation as fast as possible without overshoot.

matchHeadingToRobot(beh, nbr):

1. orientToRobot(beh, nbr.orientation + 180)

Line 1 uses the orientToRobot behavior to rotate the active robot, and is subject to the same dynamic constraints as of that behavior. So orientation is used for its increased precision.

B. *followRobot*

The followRobot behavior directs an active robot to follow a reference robot. This is a fundamental behavior and is used in many behaviors.

Step:

Always be within a distance d of the reference robot.

Always be facing the reference robot.

The active robot should move along minimum shortest path to its final position at constant velocity. The final position is any pose that satisfies the above two constraints.

followRobot(beh, nbr, rd):

1. orientToRobot(beh, nbr.bearing, 0)
2. if nbr.range > rd
3. beh.translationalVelocity = kf * (rd - nbr.range)
4. endif

C. *avoidRobot*

The avoidRobot behavior directs an active robot to move away from a reference robot. This behavior is used for dispersion.

Step:

The active robot should always be further away than distance d from the reference robot

The active robot should move along the shortest path to its final position at constant velocity. The final position is any one that satisfies the above constraint.

avoidRobot(beh, nbr, d):

1. beh1 <= EmptyBeh

ISBN: 978-988-19252-9-9
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

2. beh2 <= EmptyBeh
3. if nbr.range < d
4. orientToRobot(beh1, nbr.bearing, 180)
5. moveForward(beh2)
6. endif

D. *orientForOrbit*

This behavior orients an active robot with respect to a reference robot such that if the active robot were moving forward, it would move in a circular path around the reference robot.

Step:

Orient the active robot to move away from the reference robot if they are too close, to move towards the reference robot if they are too far, and transition smoothly between these two directions for an intermediate region.

orientForOrbit(beh, nbr, rd, orbitDir):
1. if nbr.range > rd + c
2. dir = 0
3. else if nbr.range < rd - c
4. dir = 180
5. else
6. if orbitDir = ClockWise
7. dir = 180 – nbr.range * (90 / c)
8. else
9. dir = 180 + nbr.range * (90 / c)
10. endif
11. endif
12. orientToRobot(beh, nbr.bearing, dir)

E. *Homing*

Homing is where one agent finds another agent in an attempt to decrease the distance between the agent and the goal location. Expressed formally, where P is position:

$$\forall R \; (Hexapod(R) \rightarrow home) \qquad (8)$$

Where home is:

$$\left(\frac{dP_R}{dt} \left(P_R - P_{goal} \right) > 0 \right) \qquad (9)$$

From these Equations, 8 and 9; we can derive the following algorithm:

```
Home()
if not (obstacle <= min_distance) | (goal is reached)
{
if (signal on left) turn left;
else if (signal on right) turn right;
else explore();
}
```

Fig. 8 and 9 show the implementation of the homing algorithm. In this example the second agent is the goal and the first agent's objective is to find and home in on the goal. The area outlined by the circle is the range of the localization system. Once the agent enters the range of this system it sees the other agent and homes in on it, otherwise the agent will just explore.

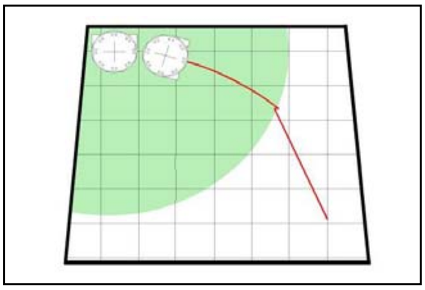

Fig. 8. Implementation of homing algorithm.

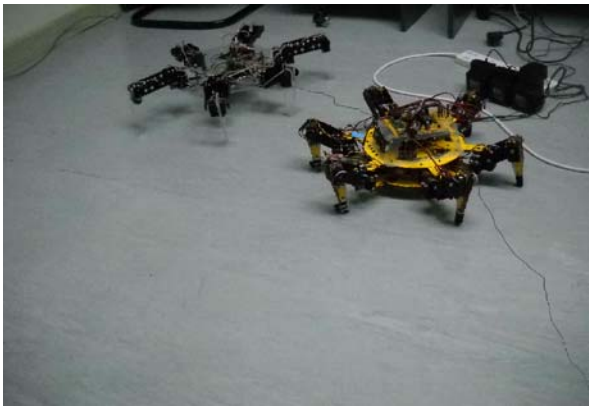

Fig. 9. Photograph of the homing of two robots.

This algorithm also produced results which were both repeatable and reliable, proving that not only was algorithm correct but also the localization system works effectively and quickly.

F. *Following (leader follower method)*

Following is the most complicated among the basic set of behaviors. The goal of Following is to maintain a fixed distance between the follower, f, and the leader, l. This can be expressed formally by Equation (10):

$$\exists l \left(\begin{array}{l} leader(l) \\ \forall \left(Follower(f) \rightarrow \frac{dp_f}{dt}(p_l - p_f) < m \right) \end{array} \right) \qquad (10)$$

The minimum distance between the leader and the follower is constrained by the bump sensors, whereas the direction of the leader is determined by the network.

The following algorithm can then be derived for this method:

```
Home();
if (bump) stop
```

Fig. 10 shows the simulation result of Homing algorithm in the agents. The dotted line denotes the leader path whereas the solid line denotes the Follower path.

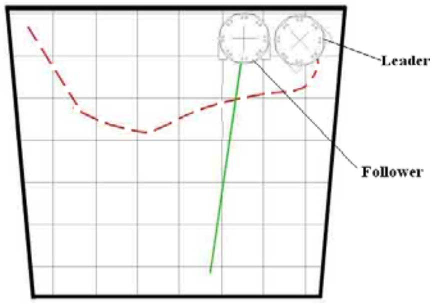

Fig. 10. Simulated Leader-Follower navigation path

To validate the Leader-Follower navigation algorithm, two robots were attached with marker pen at the bottom so that during implementation of the algorithm the robots will trace their navigation paths on the floor. The paths traced during the experiment are shown in Fig. 11.

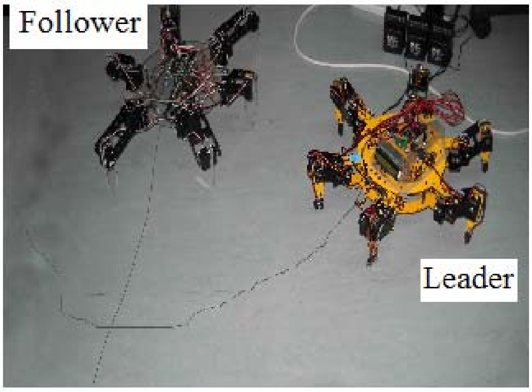

Fig. 11. Photograph of Leader-Follower paths traced on the floor.

V. Conclusions

A novel algorithm for multi-robot collaborative behavior and localization of agents has been developed. Some preliminary experiments have been presented for the collaborative multi-agent algorithm with the collective behavior in indoor localization. The experimental results show a team of two hexapod robots can localize themselves using inertial sensors as well as navigate like an organized group using their collaborative behavior. The developed algorithm can be implemented in reconnaissance mission to explore unknown terrain.

References

[1] Bhatte, S.; Majlesein, H.; Zhengmao Ye; Mohamadian, H. (2009). Development of novel wireless sensor networks and potential optimization via Kalman filtering Industrial Electronics and Applications, 2009. ICIEA 2009. 4th IEEE Conference on Digital Object Identifier: 10.1109/ICIEA.2009.5138955 Publication Year: 2009, Page(s): 3983 – 3988.

[2] Ferranti, E., N. Trigoni, M. Levene (2007). Brick& Mortar: an on-line multi-agent exploration algorithm, International Conference on Robotics and Automation, pp. 761 – 767.

[3] Alexander, B., J. Leonard and F. Fallon (2009). Cooperative Localization for Autonomous Underwater Vehicles, the International Journal of Robotics Research. 28; p. 714, DOI: 10.1177/0278364908100561

[4] Lamarche, F. and S. Donikian, “Crowd of Virtual Humans: a New Approach for Real Time Navigation in Complex and Structured Environments”, Computer Graphics Forum 23(3): pp 509-518, 2004.

[5] Chen X., Y. Yimin; Cai Shuting; Chen Jianping (2009) Modeling and Analysis of Multi-agent Coordination Using Nearest Neighbor Rules; Informatics in Control, Automation and Robotics, 2009. CAR '09. International Asia Conference on Digital Object Identifier: 10.1109/CAR.2009.18

[6] Wierstra, D., A. Forster, J. Peters, J. Schmidhuber (2009). Recurrent Policy Gradients, Elsevier Journal of Algorithms in Cognition, Informatics and Logic.