FollowMe! Mobile Team Coordination in Wireless Sensor and Actuator Networks

Stephan Bosch, Mihai Marin-Perianu, Raluca Marin-Perianu, Hans Scholten, Paul Havinga
University of Twente, The Netherlands
Email: {s.bosch, m.marinperianu, r.s.marinperianu, j.scholten, p.j.m.havinga}@utwente.nl

***Abstract*—Autonomous vehicles are used in areas hazardous to humans, with significantly greater utility than the equivalent, manned vehicles. This paper explores the idea of a coordinated team of autonomous vehicles, with applications in cooperative surveillance, mapping unknown areas, disaster management or space exploration. Each vehicle is augmented with a wireless sensor node with movement sensing capabilities. One of the vehicles is the *leader* and is manually controlled by a remote controller. The rest of the vehicles are autonomous *followers* controlled by wireless actuator nodes. Speed and orientation are computed by the sensor nodes in real time using inertial navigation techniques. The leader periodically transmits these measures to the followers, which implement a lightweight fuzzy logic controller for imitating the leader's movement pattern. The solution is not restricted to vehicles on wheels, but supports any moving entities capable of determining their velocity and heading, thus opening promising perspectives for machine-to-machine and human-to-machine spontaneous interactions in the field. Visit [1] to see a video demonstration of the system.**

I. Introduction

Personal computers, mobile telephony and the Internet substantiated the vision of *networking everyone*, at a level of quality and speed that people could barely imagine fifty years ago. Nowadays, Wireless Sensor and Actuator Networks (WSANs) [2] lay down the technological foundations for the next step in a future pervasive world: *networking everything*. Significant progress has been made in this regard and WSAN platforms are no longer prototypes but functional products. However, the transition from typical environmental monitoring and static sensor arrays to dynamic, mobile applications is still slow. *WSANs in action* represent more a vision rather than a reality. The situation is likely to change in the near future, as WSAN-based solutions are getting traction on high potential markets, such as transport and logistics, automotive, process safety, industrial automation and robotics.

This paper makes a step forward in proving that *WSANs can sense, reason and react as a group with distributed intelligence, without any intervention from the back-end and despite the hardware limitations of sensor nodes*. More specifically, we address the problem of distributed movement coordination of autonomous vehicles equipped with wireless sensor nodes. The final goal is to have a self-organizing team (or swarm) of nodes that maintain a formation by periodically exchanging their sensed movement information. We aim to provide a *fully localized* solution, without any external PC-based control, and based solely on low-cost, low-power inertial sensors (no cameras or GPS, total cost of the hardware platform below 150 $). The applications are broad, from low-level manoeuvre learning [3] to entire missions, such as cooperative surveillance, mapping unknown areas [4], disaster management [5], space exploration [6].

Distributed team coordination in WSANs faces a number of challenges. Firstly, executing all the tasks on the node – sensor sampling, processing, communication and control – may easily exceed the computational and memory resources available. Consequently, we must find the right *scheduling* that trades-off between accuracy and responsiveness, on the one hand, and sampling frequency and wireless communication duty cycle, on the other hand. Secondly, actuator nodes must run a navigation control loop for regulating the movement of vehicles. Designing and implementing a suitable *controller* for this purpose is far from trivial on limited hardware. Thirdly, using inexpensive, low-power inertial sensors also means a relatively low accuracy and robustness to noise. Therefore, *calibration*, *filtering* and *dynamic error compensation* are strongly required for improving the quality of measurements.

In view of these challenges, the key contributions of this paper are as follows. Firstly, we devise a miniaturized, low-cost navigation system using low-power wireless sensor nodes equipped with three-axial accelerometers and magnetic compasses. Secondly, we explore fuzzy logic as a lightweight and robust control solution for coordinating the group movement in a leader-follower fashion. Thirdly, we report on all development phases, covering simulation, controller tuning, sensor calibration and software implementation. Finally, we evaluate the performance of our prototype system through field experiments with two toy cars (the leader and one follower) and we discuss the most important results. A video demonstration of the working system can be found at [1].

II. Related Work

The field of robotics accounts for extensive related work on swarms (or flocks) of robots [7]. Various technologies are used to determine inter-robot distances and orientations for the purpose of maintaining a formation: modulated infrared light [8], acoustic signals among submersibles [9], omnidirectional cameras and physical attachments [10]. Such robots typically feature powerful computing boards with clock frequencies ranging from 40 to 400 MHz. In comparison, our solution targets a more loosely-coupled coordination among resource-constrained devices, using only inertial sensors and low-power wireless communication.

978-1-4244-3304-9/09/$25.00 ©2009 IEEE

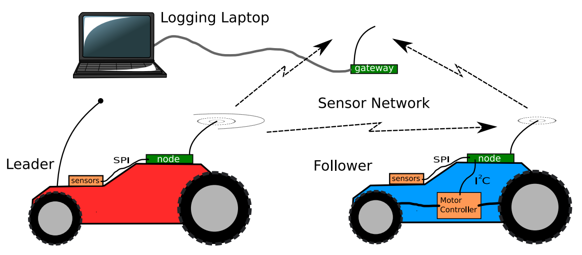


Fig. 1. Overview of the leader-follower scenario.

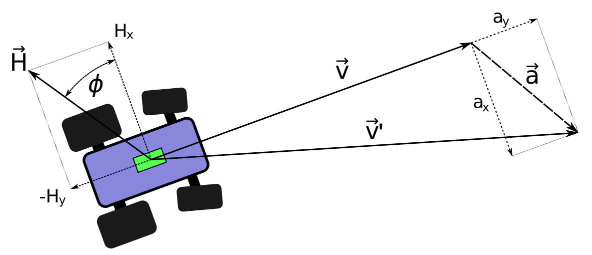


Fig. 2. Magnetic, velocity and acceleration vectors defined for the vehicle.

In recent work, Allred *et al.* [11] describe SensorFlock, which is composed of small bird-sized nodes called micro-air vehicles (MAVs). The flight control system fuses information from the GPS and gyroscope sensors on board. SensorFlock focuses on keeping the nodes autonomously in the air and provides an in-depth study of the RF characteristics and networking connectivity. In comparison, we focus on inertial sensing and explore fuzzy logic as a lightweight yet robust control method for coordinating the movements of mobile nodes in the field.

Wang *et al.* [12] overview several mobile WSAN platforms based on MICA motes and running TinyOS. The MAS-net project also considers the usage of MASmotes for formation control [13], including a leader-follower strategy. The MASmotes rely on a pseudo-GPS location system (based on cameras) and odometry to measure node displacement. In contrast, our solution is fully localized and does not require any infrastructure.

Fuzzy control for autonomous vehicles has also been considered by previous research. Kodagoda *et al.* [14] present an autonomous golf car controlled by a 450 Mhz PC. Simulations of similar systems are provided in [15] and [16]. Compared to these approaches, our system is much more resource constrained, exploits wireless communication and assumes a simpler physical vehicle model.

III. Navigation

As depicted in Figure 1, we consider the scenario of a *leader* vehicle, whose trajectory has to be copied by *follower* vehicles. The result is a moving ensemble that can be controlled from a single point or can follow a unique mission plan deployed only on the leader. Theoretically, synchronous movement is achieved when all the vehicles maintain the same *velocity* and *heading* with respect to a reference system. When moving, the leader computes its velocity and heading from sensor measurements, and broadcasts the data periodically to the followers. Each follower determines its own speed and heading, compares them with the information received from the leader and takes the necessary action to correct its trajectory if necessary. Note that no effort is made to keep track of the distance between the vehicles, which will be investigated in future work (refer to Section IX).

A. Velocity Integration

Compared to typical strapdown inertial navigation systems [17], we try to estimate the velocity v only from the accelerometer and compass data, and we aim for the smallest possible computational effort. This is possible, if we assume that the accelerometer is mounted firmly to the vehicle rigid frame and its Y axis points always to the direction of driving. Figure 2 illustrates our acceleration integration algorithm. The current velocity vector $\vec{v'}$ is computed by adding the instantaneous acceleration $\vec{a}$ to the vector $\vec{v}$ from the previous step (the size of $\vec{a}$ is intentionally exaggerated in the figure). The magnitude v' becomes:

$$v' = \sqrt{a_x^2 + (v + a_y)^2} \tag{1}$$

where a_x and a_y are the components of the acceleration vector. The time step for integration is considered equal to unity.

Unfortunately, accelerometers also measure the gravitational acceleration along with the true acceleration. Commonly, a gyroscopic sensor is used to keep track of the relative direction of the gravity vector to properly subtract it from the measurements. We avoid this additional complexity for now by assuming that our vehicles drive on a relatively smooth and level surface. This keeps the measured gravitational acceleration approximately constant and easy to subtract from the measurements. Extending our system to account for sloping surfaces is a topic for future work.

A known problem of integrating acceleration is the accumulation of errors in time. If no external reference is available, the error can potentially increase to infinity. Our assumption is that the vehicles do not move continuously, but also have stationary periods during which the velocity estimate can be reset to zero. This is also a good time to measure the gravitational acceleration since no other acceleration should be present. For determining the "standing still" situation, the sensor nodes continuously analyze the variance of the acceleration over a sliding time window with respect to a threshold determined experimentally. The downside of this approach is that it introduces a certain delay between the actual moment of stopping and the detection of standing still.

B. Heading Computation

The heading (or azimuth) ϕ is computed from the components of the magnetic field intensity H measured by the magnetic compass (see Figure 2):

$$\phi = \arctan(H_y/H_x) \tag{2}$$

The heading indicates the vehicle orientation with respect to the magnetic North Pole. To compensate for static inclination (roll and pitch tilt angles), we rely on the calibration procedure described in Section VII-D. Our experimental results show that this method is robust to small tilt effects whilst driving, as long as the vehicles remain on a relatively level surface.

IV. Wireless Communication

The communication protocol required by our approach is straightforward: the leader broadcasts its movement parameters periodically, while the followers just listen for the incoming data packets. This scheme ensures the scalability of the solution by allowing, theoretically, an indefinite number of followers. Nevertheless, a method of logging the behavior of both leader and follower is needed for testing and evaluation purposes. To solve this practical problem, our single follower always waits for the leader to transmit its movement parameters at the end of each sampling period, and then sends its own data. This method has the advantage of avoiding data packet collisions implicitly. The drawback is that scheduling the sampling and communication tasks on the follower becomes problematic (see Section VII-F). A gateway node listens to all incoming packets and logs them to a PC. Packet losses are identified based on sequence numbers. In addition to the movement data, the messages sent by the follower include also the controller outputs and raw sensor data. This is a valuable feature because we can reproduce the experiments in a PC-based simulator and correct or tune the controller accordingly.

V. Fuzzy Controller

Currently, the field of fuzzy logic is largely overlooked by the WSAN community. However, fuzzy logic has several properties that qualifies it as an effective tool for WSAN problems. Firstly, it can be implemented on limited hardware and it is computationally fast [18], [19]. Secondly, it handles unreliable and imprecise information (as usually the case with sensor data), offering a robust solution to decision fusion under uncertainty [20]. Thirdly, fuzzy-based methodology substantially reduces the design and development time in control systems [21]. Finally, fuzzy controllers handle non-linear systems (most real-life physical systems are non-linear) better when compared to conventional approaches [21].

A fuzzy controller executes three basic steps: *fuzzification*, *inference* and *defuzzification*. During fuzzification, the numeric input values are mapped to fuzzy sets by applying the membership functions. Based on the fuzzified inputs, the controller infers through its IF-THEN rule set and produces an aggregated fuzzy output. The final control action is derived by defuzzifying this aggregated fuzzy output.

In our case, the control objective of the follower is to adapt the velocity and heading according to the leader movements. Since the vehicle has separate motors for accelerating and steering, it is reasonable to decompose the problem into two independent controllers, with the benefit of simplifying the design and tuning. The two controllers are structurally identical. The only differences lie in the range of the input values, the definition of the membership functions and the post-processing operations.

The two fuzzy controllers follow the design methodology proposed by Mamdani [22]. The building blocks are:

- *Inputs*. As inputs we use the error e and the change in error Δe between the actual values of the movement parameters and the desired values. This is a common approach for improving the controller stability when the output value is close to the optimal operating point.
- *Fuzzy sets*. For an increased control granularity, we define five fuzzy sets for each input, namely NM (Negative Medium), NS (Negative Small), ZE (Zero Equal), PS (Positive Small) and PM (Positive Medium).
- *Membership functions*. To leverage the computational effort, we use triangular membership functions, equally spaced. The width of the membership functions influences the *aggressiveness* of the controller, i.e. what margin of error it tries to achieve.
- *Rule-based inference*. We use the *max-min* fuzzy inference method.
- *Output*. The controller output is decided by defuzzifying the aggregated inference result according to the *center of gravity* (COG) method.

Before actuating the car motors, the outputs of the fuzzy controllers are subjected to a post-processing step. This inverts the steering control when the vehicle is driving backwards, it applies the follower's inductive brake (see also Section VII-E) when the leader has stopped and it compensates for the asymmetry between acceleration and deceleration as caused by friction. As shown in the simulation model of Section VI-A, there is also increased friction during steering, which reduces the maximum acceleration during steering. This is currently not accounted for in the controller for simplicity.

VI. Simulation

The simulation framework we developed plays an important role in designing and tuning the fuzzy logic controller.

A. Simulation Model

To model the velocity dynamics of the vehicle realistically, we must take both the friction and the throttle capacity of the motor into account. The effective acceleration a_e, i.e. the result of the net force applied to the vehicle, is given by:

$$a_e = ta_t \; - \; a_f \tag{3}$$

where $t \in [-1; 1]$ is the current throttle control value, a_t is the maximum acceleration generated by the throttle (and depends on the actual level of the battery powering the motor) and a_f represents the acceleration induced by friction. The effective acceleration a_e is further integrated to estimate velocity, as explained in Section III-A.

Deriving an accurate model for the frictional acceleration a_f is more complicated due to the multitude of physical factors involved in the process. In our simulations, we use the following simplified model:

$$a_f = a_{fk} \; + \; c_d|v| \; + \; c_s|\delta| \; + \; ba_{fb} \tag{4}$$

where a_{fk} is the usual kinetic friction, $c_d|v|$ represents the drag friction (proportional to velocity v for small objects moving at low speeds, according to Stokes law), $c_s|\delta|$ yields the sliding friction when the vehicle is steering with angle δ and a_{fb} is the friction induced by braking ($b \in \{0;1\}$ being the brake control signal).

The final component of the vehicle model is the heading. We assume that the front wheels of the vehicle can steer with an arbitrary angle between $-\delta$ and δ. The equation for updating the current heading ϕ' is:

$$\phi' = \phi \; - \; s\delta \tag{5}$$

where ϕ is the heading at the previous time step and $s \in [-1;1]$ is the steering control value.

Unlike the throttle control, the steering control value is not directly applied to this simulation model. In stead, it is first passed through a low-pass filter to account for the mechanical responsiveness of the vehicle's steering. To simulate sensor and control inaccuracies, uniformly distributed noise is added to the simulated sensor measurements and to the controller outputs.

As an example, Figure 3 shows the behavior of the heading controller. The simulation is instructed to change the desired heading 90° to the right every 100 iterations, starting with a heading of 30° . This causes the simulated car to drive an approximately square pattern. At the current instance of time, we see the measured heading, the error e, the change in error Δe and the steering command. Our simulations show that both the velocity and heading controllers work adequately when subjected to the model of dynamics previously explained. The follower manages to closely keep to the leader trajectory and shows realistic changes in speed when taking turns. For detailed results on how field test results match the simulations, see Section VIII-B4.

B. Tuning

For tuning the fuzzy controller membership functions, we run a series of simulations with different controller configurations. For example, Figure 4 shows the average absolute error in the heading of a simulated vehicle driving a straight line, plotted against the width of the fuzzy input membership functions. Each point in the plot is the average result of 20 simulation runs. As we explain in Section V, the heading controller becomes more aggressive when the input membership functions get narrower. On the one hand, if the controller is too aggressive, it can produce undesired oscillations in steering. On the other hand, if it is not aggressive enough, the follower has low responsiveness and performs suboptimally. Figure 4 provides an indication of where the actual optimum can be found. We currently use a width of 30° for the membership functions.

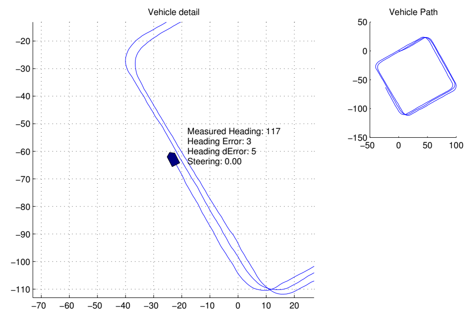


Fig. 3. Simulation view of the heading controller.

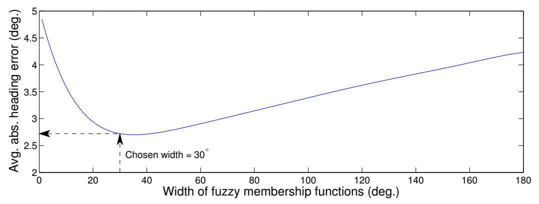


Fig. 4. Tuning the width of membership functions.

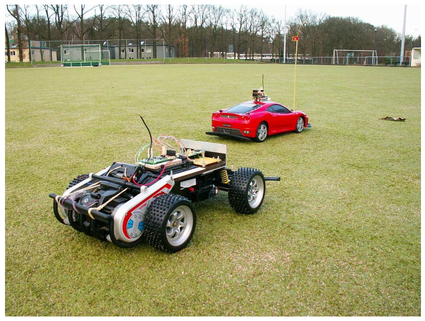


Fig. 5. Leader and follower toy cars in the field.

VII. Implementation

We implement our prototype of the leader-follower system on two toy cars. The follower is modified to allow the sensor node to control its actuators. The leader car remains unmodified apart from the sensor node attachment. Figure 5 shows the two vehicles at our test location. Visit [1] to see a video demonstration of the system.

It is evident from Figure 5 that the two toy cars are different. Apart from the visual differences, the leader car is more agile and has a higher top speed than the follower car. Also, the leader has a different and less sturdy supension for its wheels. As we show in Section VIII, these differences have an impact on the system performance, but dispite these differences the system still performs adequately.

In comparison to the sensor nodes, these toy cars have very large batteries and consume much more power. We could have used much smaller cars, but such large cars are easier equipped with external hardware, which is more important for this proof of concept. Also, the energy-efficiency is not the only concern of this research: it is also about simplicity, small size and low cost. Primarily, we want show that implementing a high-frequency motion sensing and feedback control loop is feasible on very limited WSAN nodes. This opens the possibility for many other dynamic applications, possibly having nothing to do with wheeled cars.

In the following, we provide a description of the hardware and software implementation.

A. Control

The follower's controller is executed at regular intervals using the latest sensor data communicated from the leader and the equivalent data collected on the follower itself. The frequency of controller execution determines how responsive the follower is to sudden changes in the leader's movement or sudden deviations the follower may incur. Increasing the control frequency will require using a matching sensor sampling frequency and leader-to-follower message frequency. As we explain in Section VII-F, this has a severe impact on the feasibily of scheduling all these tasks on the CPU. Increasing the communication frequency also has an impact on the used bandwidth, but the available bandwidth is large enough for this experiment (only 32 bytes are communicated each time). In Section VII-B2 we show that the lower bound of the required accelerometer sample frequency is not determined by the controller, but rather by the noise spectrum that needs to be accounted for. We chose the control frequency to be 16 Hz, which yields a theoretical response time of about 60ms while still being feasible in terms of scheduling and communication. This also matches well with the sample frequencies chosen for the sensors (refer to Section VII-B2).

B. Sensing Interface

1) Hardware: As the generic sensor node board, we use the Ambient μNode 2.0 platform [23], running on the low-power MSP430 microcontroller from Texas Instruments (48kB of FLASH memory and 10kB of RAM). The radio transceiver has a maximum data rate of 100kbps. The nodes run AmbientRT [24], a real-time multitasking operating system.

The selection of inertial sensors is driven by cost concerns, accuracy, form factor, power consumption and interfacing capabilities. As accelerometer, we choose the LIS3LV02DQ three-axial sensor from STMicroelectronics [25]. The price is around 15 $ and the typical power consumption is 2mW. The list of features include user selectable full scale of ± 2g and ± 6g, I^2C/SPI digital interface, programmable threshold for wake-up/free-fall and various sample rates up to 2.56kHz. Finding a low-power, three-axial magnetic compass operating at 3V supply voltage is more difficult. We choose the MicroMag 3 sensor manufactured by PNI Corporation [26]. The price range is 60 $ and the typical power consumption is 1.2mW. The MicroMag3 uses the Magneto-Inductive (MI) sensing technique [27] and provides a relatively large field measurement range (± 11 Gauss) on the digital SPI interface. The compass sensor and the accelerometer are connected to the node using a shared SPI bus.

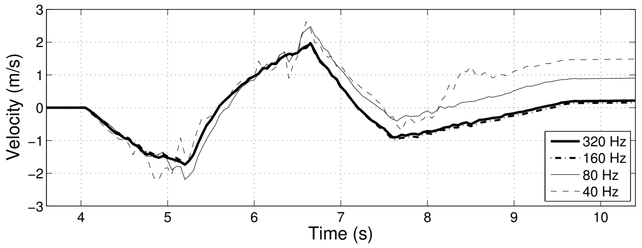


Fig. 6. Velocity integration at different frequencies.

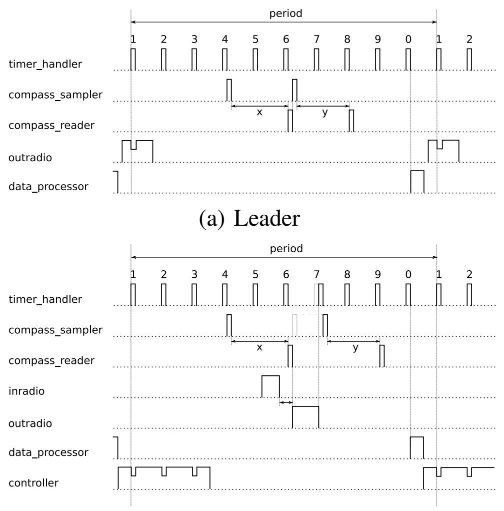
2) Sensor Drivers: Choosing the right sampling strategy for the two sensors creates an intricate trade-off among accuracy, power consumption and scheduling feasibility. For an accurate velocity integration, it is imperative to sample the acceleration at a rate higher than the Nyquist frequency of any significant noise in the input data. This noise is for the most part harmonic and caused by the vibrations that result from the vehicle rolling over the floor. We did not measure the noise directly, but we in stead evaluated the performance of the velocity integration at increasing sampling frequencies as shown in Figure 6. For this figure, the accelerometer is placed on a toy car that accelerates backward and forward. The lower 40 and 80 Hz frequencies exhibit large velocity fluctuations, whereas 160 and 320 Hz provide a velocity progression that correlates well with the experiment. Because the improvement from 160 to 320 Hz was not significant in multiple experiments, we choose the 160 Hz sample frequency for the accelerometer, which is conveniently ten times faster than the control frequency.

The sampling rate of the compass is also configurable, but special attention is required, because it relates inversely proportional to the achievable sensor resolution and subsequently to the angular sensitivity. We choose to use a sampling frequency of 16 Hz, which provides theoretically an angular sensitivity higher than 0.5°, and additionally matches the chosen control frequency.

Having the accelerometer sampled ten times faster than the compass can generate problems, as both sensors share the same SPI bus of the MSP430 microcontroller. Fortunately, the compass is able to complete its measurements in the background, meanwhile releasing the SPI bus for accelerometer sampling. The only complication is that the compass requires explicit read commands for each of the three axes. As a consequence, the sampling tasks of the two sensors have to be interleaved. The detailed scheduling strategy is explained in Section VII-F.

C. Sensor Placement

Sensor placement is an important aspect that can seriously affect the system performance. Both the accelerometer and the compass need to be mounted rigidly to the vehicle frame, in order to minimize the level of vibrations during movement. Because we use different vehicles for leader and follower, the placement of the sensors is not identical. This problem is easily solved through calibration (see Section VII-D).



(a) Leader

(b) Follower

Fig. 7. Task scheduling.

The placement of the compass sensor requires additional care. During our experiments we have identified the following factors influencing the compass: the electromagnetic field created by the car motors, the batteries of the sensor node and the WSN radio transceiver. To alleviate these problems, we use a 5mm shielding metal plate mounted below the sensor board to shield it from the steering motor, we place the battery pack as far from the compass as possible and we introduce an additional scheduling delay between the compass sampling and radio tasks to prevent compass and WSN radio from being active at the same time (see scheduling details from Section VII-F).

D. Calibration

The calibration procedure we utilize compensates for both magnetic field distortions (due to ferrous or magnetic nearby structures) and inclination relative to the reference horizontal plane (see also Section III-B). The idea is to drive the vehicles in a circular movement and collect the readings on the X and Y compass axis. Instead of the ideal circle centered in (0,0), we usually get an offset ellipse. From the minimum and maximum recorded values we determine the scale and offset calibration coefficients that project the ellipse back to the desired circle (for the details of this method see [28]). The calibrated compass values are subsequently computed as:

$$\begin{aligned} H_x' &= H_x X_{scale} + X_{offset} \\ H_y' &= H_y Y_{scale} + Y_{offset} \end{aligned} \quad (6)$$

E. Motor Controller

The follower toy car has two electric motors: one for driving and one for steering. To control them, we use a separate MSP430 microcontroller and dedicated circuitry. The two microcontrollers communicate on a separate software I^{2}C interface, so that the SPI-based dialog with the sensors is not affected. The following control commands are available: *Disable* - turns off the motor, the axle can turn freely, *Throttle*- controls the intensity of forward or backward acceleration, *Steering* - controls the angle of the front wheels, *Brake* - short-circuits the rear drive motor to induce inductive drag on the rear wheels.

F. Scheduling

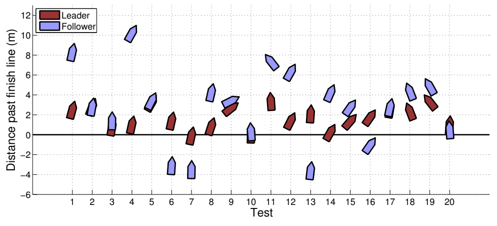
Getting the right scheduling on the limited sensor node is the most challenging part of the implementation. In this context, the real-time multitasking support of the AmbientRT operating system running on our sensor platform is highly useful. AmbientRT uses earliest deadline first with inheritance (EDFI) scheduling, meaning that task priorities are assigned depending on their maximum allowed completion time. As explained in Section VII-B2, the sampling tasks of the accelerometer and compass must be interleaved. In addition, the influence of WSN radio operation on the compass described in the previous section creates additional dependencies between the tasks associated with these two resources. It is essential therefore to devise the task deadlines in such a way that the execution sequence generated by the EDFI scheduler fulfills all these requirements.

On the leader, the process is simpler because we know when the radio transmission occurs (at the beginning of a new sampling period) and we do not have a control task (the leader car is driven remotely). Figure 7(a) sketches the execution sequence during one sampling period (task durations are not to scale). The heartbeat of the system, termed as the *timer_handler* task, samples the accelerometer at 160 Hz. Being slower, the compass sampling requires two trigger tasks, *compass_sampler* and *compass_reader*, for initiating the measurement and reading the data, respectively. In addition, the X and Y axis cannot be read simultaneously (see also Section VII-B2). At the end of the sampling period, the *data_processor* task updates the movement status based on the measured data and prepares the message to be transmitted by the *outradio* task.

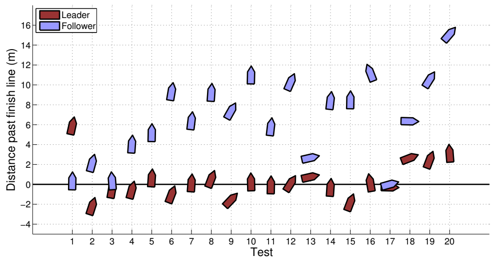
The scheduling on the follower is complicated by the asynchronous incoming messages from the leader. Moreover, for logging and debugging, we also need an *outradio* task on the follower. As shown in Figure 7(b), it is possible to receive data over the radio (the *inradio* task) while sampling the X axis of the compass sensor. In this situation we try to minimize the time overlap between the compass and radio tasks by introducing two delays: (1) the *outradio* task is initiated only after *compass_reader* finishes and (2) the sampling of the Y axis is postponed until the radio transmission is over. The *data_processor* task has a similar function as on the leader, but its completion triggers the *controller* inference and actions described in Section V.

VIII. Field Tests and Results

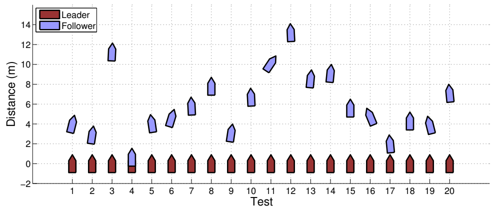
We experiment with the complete system at our university's hockey field, oriented 12° NE. This location has the advantage of being a large, relatively flat surface, without metallic



(a) Linear tests.



(b) Square tests.



(c) Random tests.

Fig. 8. Finish results of the field tests.

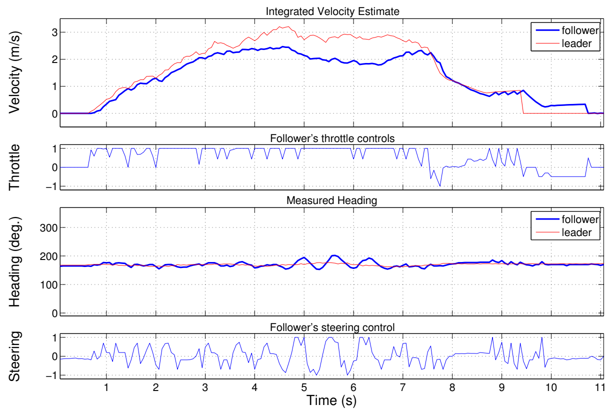


Fig. 9. Velocity and heading of the 10^{th} linear test.

structures close by. To evaluate the performance of the system in different situations, we perform tests with three different drive patterns:

- *Linear tests*. The leader drives a 20m straight line.
- *Square tests*. The leader drives a square of side length 15m. The finish line is perpendicular to the start line.
- *Random tests*: The leader drives along a random path for 30s with varying velocity.

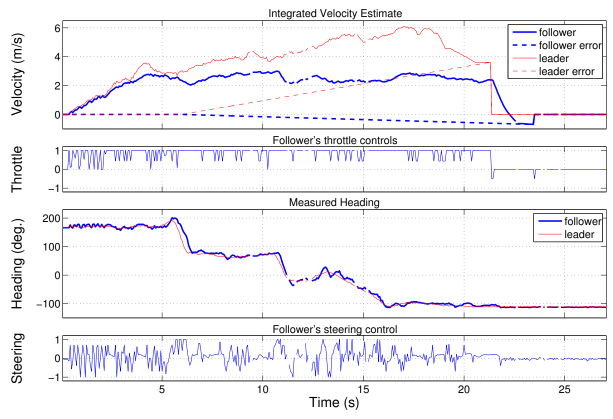
We perform 20 experiments for each type of test. We vary the driving speed, alternate between driving forward and backward (in random tests), and change the path orientation (in linear tests). In each experiment, the distance between the leader and the follower is 2m at the starting line. At the end of the experiment, we measure the *final heading* of the vehicles (using a regular compass), the *relative distance* between them and the *distance to the finish line* (for linear and square tests). In addition, we log the velocity and heading computed by the two sensor nodes, as well as the throttle and steering outputs of the fuzzy controllers running on the follower. Although not an absolute reference, these logs provide useful information about what is actually happening on the two nodes. Details are presented in the following sections.

1) Linear Tests: The linear tests are particularly valuable to assess whether the follower can maintain a constant heading while imitating the velocity of the leader. Figure 8(a) depicts the ending positions and headings of the two vehicles with respect to the finish line (thick horizontal line at 0m). We see that the follower succeeds to copy the leader movement in most of the experiments. The final relative distance is 3.7m on average, which means that the follower deviates with approximately 9cm per traveled meter (the initial relative distance is subtracted when computing this value). The difference in final headings is 8° on average and in 90% of the cases below 11° .

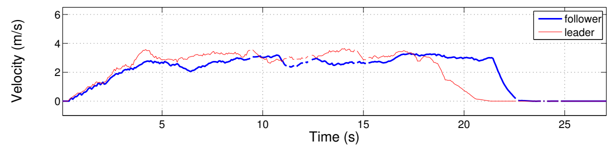
The results presented in Figure 8(a) show only the final situation of each test. Figure 9 gives further insight on how the performance of the follower may actually vary in a particular experiment (test 10). The top half shows the velocity as integrated by the two nodes and the corresponding throttle control of the follower, while the bottom half shows the changes in measured heading and the steering output of the heading controller. We make the following observations:

- In the interval 4-8s, the leader accelerates to a velocity higher than what the follower can manage. During the rest of the test, the follower succeeds to keep up. Eventually, the velocity is reset through the motion detection technique explained in Section III-A. The two cars end up very close to the finish line: 0.1m and 0.3m.
- Although the final difference in heading is only 6° , the follower sways at much greater angular deviations in the interval 4-8s. The excessive steering causes extra friction and reduces the velocity of the follower, which remains behind. The increased friction is evident from the fact that the follower's velocity drops while steering and the throttle is at maximum. The swaying behavior shows that the heading controller can become unstable when driving a straight line, although it corrects the orientation in the end.

2) Square Tests: Compared to linear driving, during square tests the leader successively steers by 90° and runs over a larger distance (60m in total). Figure 8(b) summarizes the results of the 20 experiments. As expected, the differences in position and orientation increase compared to linear tests. The average final distance between vehicles is 8.3m, meaning that the follower deviates approximately 11cm per traveled meter.



(a) Raw data.



(b) Corrected velocity.

Fig. 10. Velocity and heading of the 6th square test.

The difference in final heading is 12° on average and in 90% of the experiments below 20° .

Figure 10(a) plots the data transmitted by the sensor nodes during one particular square test (test 6). Gaps in the graphs correspond to occasional packet losses at the gateway. We make the following observations:

- Both the leader and the follower measure consistently the changes in heading. We can see clear turns corresponding to the rectangle corners at intervals of approximately 5s. A certain amount of swaying by the leader, especially around 3s, cannot be avoided because the leader is driven manually. The final difference in heading is 11° .
- The velocity increases correctly in the first 6s, until the first turn. From this moment, the leader accumulates errors and ends up with a high value at the moment of stopping. The velocity is reset through the motion detection technique explained in Section III-A. The follower does not have this problem. Its speed remains relatively constant from 6s until 21s, when it finds out that the leader has stopped and, consequently, applies the braking procedure. The follower velocity returns to a value much closer to zero compared to the leader.
- Looking back at Figure 8(b), we understand the effect of error accumulation in the leader velocity. The follower tries to keep up and accelerates to its maximum. The leader detects that it stopped with a certain delay and then informs the follower. The latter also needs some time to brake, so it eventually stops at 10.7m away from the leader.

3) Random Tests: The random tests involve steering in random directions and large variations in velocity. Each of the 20 experiments lasts 30 seconds (there is no finish line). Figure 8(c) shows the relative distances and headings of the two vehicles. We obtain an average of 6m final distance between vehicles and a final heading difference of 8° .

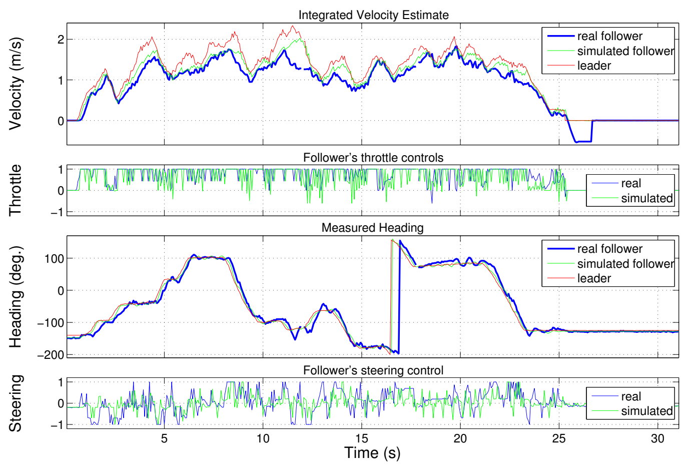


Fig. 11. Velocity and heading of the 17th random test (including simulation).

Figure 11 depicts one particular experiment (random test 17). In this experiment, the leader drives with high variations in velocity and heading. We make the following observations:

- Both the leader and the follower are responsive to the changes in heading. The abrupt transition just after 16s is in fact the trigonometric turning point from -180° to 180° . At this instance, we notice a delay in the response of the follower steering. The final difference in heading between the two vehicles is 6° .
- Both nodes record clearly the steep acceleration and deceleration. The leader velocity accumulates few errors and returns close to zero at the end of the experiment. The follower has a larger negative offset, which means that it drives faster than it thinks.
- The follower matches the leader velocity during the periods with less steep acceleration, for example around 1s and 16s, but falls behind during high peaks, such as 4s and 12s. In this test, the swaying behavior of the follower is much less prevalent, meaning that it incurs less friction, drives a straighter path and thus keeps up better with the leader. The main reason that the follower does not keep up at all times lies in the constructive limitations of the follower car, which cannot accelerate as fast as the leader. Overall, the follower moves slower, but, since it incurs a delay in braking, it drives further for a short time and it ends up at 2m distance to the leader, which is is approximately the same distance the cars started at.

Surprisingly, the random tests show better performance than the linear and square tests with respect to the finishing position, although the distance covered is larger. The explanation lies in the steering behaviour of the follower: since the follower does not sway that much during the random tests, it can attain a higher velocity due to the reduced friction and it can keep up better with the leader. Moreover, since the follower's steering quickly stabilizes to the desired heading, the heading deviation is smaller as well.

A. Summary of results

Figure 12 summarizes the results of final differences in distance and heading. The initial relative distance of 2m between cars is not subtracted from these results. The error bars are plotted with 5 and 95 percentiles.

In order to better characterize the continuous performance of the leader-follower system, Table I provides detailed statistics from all the logged tests. For both velocity and heading, we compute the mean and standard deviation of the absolute error between the two vehicles at any moment in time, as well as the correlation coefficient of the signals recorded in each test. We make the following observations:

- In contrast to the results of ending positions (from Figure 12), the values in Table I offer just the relative image of what the nodes "think" about their velocity and heading. More specifically, these values include the inherent sensor and integration errors.
- The velocity errors in the logs are very significant when compared to the finish distance results. This is caused by the fact that leader and follower accumulate different errors in the integration of acceleration, with the leader being worse in most cases. Because the follower cannot achieve the erroneously high leader velocity, the integration errors are not fully propagated in the finish results. Alternatively, in some cases the stopping delay between leader and follower can compensate for the distance the follower may have accumulated during the test.
- The heading correlation coefficient in linear tests is smaller than in square and random tests (0.64 compared to 0.99). The reason is that the heading varies much less in linear tests, as there are no explicit turns, and therefore any small difference in the compass readings of the two nodes and the follower's swaying behavior have a strong impact on the correlation.
- The performance in random tests is slightly worse than indicated by the results of ending positions. A detailed analysis of the logs reveals two main reasons. Firstly, during random tests, the leader is subjected to steep acceleration and steering at times. Due to constructive limitations, the follower can not always match this behavior, but later on it recovers the difference. Secondly, the vehicles drive over greater distances in the random tests than in other tests and thus accumulate higher errors through acceleration integration. However, their velocity is physically limited, which explains why the actual ending results are better than what the sensor nodes compute.

B. Discussion

The results presented so far open several points of discussion.

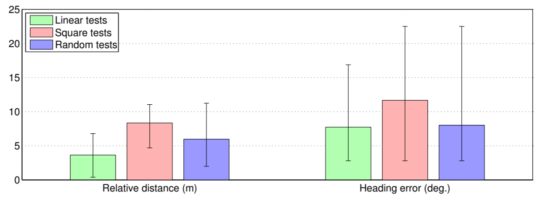


Fig. 12. Summary of results.

1) Vehicle Capability Differences: The test results are influenced by the fact that the vehicles have differing capabilities in terms of maximum velocity, acceleration and steering agility. It is very easy to drive the leader beyond the maximum attainable velocity and steering rate of the follower. In all experiments, care was taken to prevent this from happening, except for some of the random tests (for example the 17^{th} random test presented in Figure 11). However, on the straight paths, the swaying behavior of the follower further limits its attainable velocity, because this effect increases the incurred friction. The increased friction is evident from the drop in velocity during steering at constant throtlle. The swaying also increases the traveled distance, making the follower fall behind even when it matches the leader's velocity. The results show that the vehicle differences have an impact on the system performance, but the follower still manages to mimic the leader's movements within the limits of its capability.

2) Error Accumulation: Referring back to Figures 8(a) and 8(b), we notice that the follower generally ends up further relatively to the finish line than the leader (the random tests are irrelevant in this case, as there is no finish line). As mentioned before, this is caused by the errors accumulating in the acceleration integration process and the inherent delay of the motion detection technique (see Section III-A). The pertinent question that remains is what does the error accumulation look like in practice and what are the sources of the accumulating errors?

To examine the nature of the error accumulation, we go back to the square test depicted in Figure 10(a). The top plot shows an estimation of the accumulated error in velocity for both leader and follower (dotted lines). The error seems to start accumulating approximately linear when the cars make the first turn. This is also the moment where the difference in the integrated velocity of the two vehicles starts to be significant. Figure 10(b) shows the corrected velocities, where the linear error estimation is subtracted from the raw data. The matching between the two velocities is much better and corresponds to the actual behavior observed in the field.

The linear accumulation of the error in the integrated velocity suggests that a small but constant acceleration error emerges when the vehicles change direction. This is most likely due to small changes in tilt angles of the sensors, caused by the pull of centripetal force during steering. This in turn changes the direction of the gravitational acceleration. The gravity vector measured at the previous standstill is then not entirely accurate anymore. This situation is often maintained until a later steering operation changes it in another direction. We attribute the worse performance of the leader to the less sturdy suspension and the weaker mounting of the sensor on the hull of the vehicle. Whether a changing gravity vector can be identified and compensated for at runtime remains an open problem (see also future work in Section IX).

Other less significant sources for the integration errors are non-harmonic noise in the acceleration measurements, sampling errors and possibly temperature changes that affect the sensor.

3) Error Canceling: Despite the integration errors at both vehicles and the swaying behavior of the follower, the latter still manages to keep on the right trajectory, especially when the leader is driven with frequent changes in heading. This effect is shown both during the random tests and in an additional test that lasted for more than 10 minutes. This is partly explained by the fact that errors tend to compensate for one another. In addition, the limited velocity of the vehicles improves the overall performance. When the leader does not drive faster than what the follower can manage, the latter follows correctly, even when the estimate of the leader velocity rises beyond its constructive possibilities.

4) Simulation versus Practice: The main difference between simulation and practice is that the simulation does not consider the occurence of very large velocity integration errors. It only considers sensor noise and control inaccuracies. The linearly increasing velocity error is thus not observed in simulation. Also, the compass is assumed to work without incident in simulation. In reality, it is sensitive to many forms of interference, including motor actuation, passing large metal objects and possibly also the cars approaching one another. Therefore, our algorithms present adequate performance in simulation, while practical tests show more erratic behaviour.

For those experiments where these differences have less impact, the experimental results offer the possibility to validate and refine the simulation vehicle model presented in Section VI. For this purpose, we use the data recorded during the field tests as input to the simulation framework. Figure 11 shows the simulated velocity and heading of the follower, as compared with the perceived velocity and heading derived from sensor data. We notice that the field results validate the simulation output in terms of velocity. The most significant difference occurs between the simulated and the measured heading, with the real follower steering slower than in simulation. The reason is not the maximum steering angle the vehicle can achieve, but rather the delay between a change in the steering signal from the controller and the actual change in the heading of the vehicle. This may contribute to the fact that the follower's swaying behavior is much less prevalent in simulation, but we were unable to fully simulate the large deviations shown in the linear tests.

The second half of Table I gives a quantitative view on how well the simulation matches the real behavior. Similar to Section VIII-A, we provide three statistical indicators for each type of test: the mean and standard deviation of the absolute error, and the correlation coefficient of the signals recorded in each test. In general, the real and simulated behaviors match well. The heading correlation coefficient in linear tests is relatively small for similar reasons as explained in Section VIII-A. The velocity errors are high because the test logs include velocity integration errors and the simulations do not. Apart from the integration errors, a detailed analysis of individual tests shows that a more accurate friction model is needed to further enhance our simulations.

	Follower vs. Leader			**Simulated vs. Follower**		
	linear	square	random	linear	square	random
Velocity (m/s):						
mean abs. err.	0.53	1.97	2.75	0.20	0.48	0.58
std. dev.	0.55	1.42	1.93	0.24	0.39	0.34
corr. coeff.	0.93	0.63	0.54	0.96	0.84	0.84
Heading (°):						
mean abs. err.	7.38	10.23	14.99	7.35	9.11	11.38
std. dev.	9.79	14.23	21.06	9.78	11.96	15.45
corr. coeff.	0.64	0.99	0.99	0.64	0.99	0.99

TABLE I
TEST ERROR AND CORRELATION STATISTICS.

IX. CONCLUSIONS

As concluding remarks, we briefly iterate the main advantages and limitations of our system, name the practical lessons learned and outline the future work directions.

Advantages. Our system demonstrates that *WSANs can sense, reason and react as a group with distributed intelligence, without any intervention from the back-end and despite the hardware limitations of sensor nodes*. The solution is not restricted to vehicles on wheels, but supports *any moving entities capable of determining their velocity and heading*. Therefore, it creates a promising basis for both machine-to-machine and human-to-machine spontaneous interactions in the field. Another advantage is the constructive robustness to errors: from any initial or intermediate faulty orientation, the follower is able to return to the correct heading, due to the usage of a global reference system. Furthermore, making use of a fuzzy controller facilitates the implementation on resource constrained sensor nodes and handles robustly the noisy sensor data as well as the rough mechanical capabilities of the vehicles. With respect to wireless communication, the leader-follower model is inherently scalable and tolerates transitory packet losses.

Limitations. As in any inertial navigation solution, errors may accumulate through acceleration integration. Without stopping periods or external reference sources (e.g. GPS), the error can potentially grow to infinity. Additionally, our current implementation does not support dynamic tilt compensation (for example when moving on inclined terrain). The compass utilization is restricted to environments without strong magnetic influences, typically outdoors. With respect to convoy-like applications, the main limitation is the lack of any information and subsequently control of the relative distance between the vehicles (see also future work).

Lessons learned. The main lesson learned during the implementation on sensor nodes is that sampling inertial sensors requires a major share of the available CPU power and preferably a multitasking operating system. Scheduling all the tasks is difficult, especially due to the compass sensitivity to various interferences. The large amount of computation involved in inertial navigation algorithms is another potential source of hard-to-identify bugs. Introducing a feedback loop from the sensor node to the simulator proved to be an efficient debugging method for this problem. From field experiments, the most important lesson learned is that the constructive limitations of the vehicles can have both a negative impact (such as the lack of an actual braking system) and a positive influence (the limited engine power of the follower bounds the effect of error integration in velocity). Compass calibration remains a necessary step for producing accurate experimental results. Driving the toy cars proved to be challenging, as collisions at 30 km/h would destroy the sensors mounted on top. In general, experiments were more time-consuming than expected, required a dedicated, large open field and, last but not least, were dependent on the occasional good weather.

Future work. We plan to pursue multiple directions for future work:

- First of all, the work performed thus far considered only one follower, but it is important to extend this to a group of followers to verify our technique on large groups of nodes.
- Also, our vehicles are currently assumed to drive over a level surface, which is not practical in many cases. To solve this, we intend to investigate the use of cheap gyroscopic sensors to compensate for sensor inclination changes.
- To improve performance further, we would like to incorporate relative distance estimation into the fuzzy controller, making it possible to maintain a constant distance between the vehicles. This also makes our solution applicable to convoy-like applications. Coarsely, this can be implemented by integrating the perceived velocity diffence to keep track of the distance change. However, to prevent integration problems, a more direct means of measuring the distance is necessary, e.g. using RSSI distance estimation.
- Finally, our results show that differences between the vehicles, like for instance the maximum velocity, can have an impact on the system performance. More work is needed to resolve such issues, because we would like to explore pervasive group interactions among heterogeneous moving entities (not only vehicles on wheels) using the same general idea.

References

[1] "FollowMe presentation and live demonstration video," http://www.youtube.com/watch?v=ZzWYO5dbo1M.

[2] I. Akyildiz and I. Kasimoglu, "Wireless sensor and actor networks: Research challenges," *Ad Hoc Networks Journal*, vol. 2, no. 4, pp. 351–367, 2004.

[3] A. Coates, P. Abbeel, and A. Ng, "Learning for control from multiple demonstrations," in *Proceedings of the 25th Annual International Conference on Machine Learning (ICML 2008)*, A. McCallum and S. Roweis, Eds. Omnipress, 2008, pp. 144–151.

[4] C. S. F. S. W. Burgard, M. Moors, "Coordinated multi-robot exploration," *IEEE Transactions on Robotics*, vol. 21, no. 3, pp. 376–386, 2005.

[5] "AWARE Project," http://grvc.us.es/aware/.

[6] E. Gaura and R. Newman, "Wireless sensor networks: The quest for planetary field sensing," in *LCN*, 2006, pp. 596–603.

[7] G. Beni, "From Swarm Intelligence to Swarm Robotics," *Swarm Robotics: SAB International Workshop*, 2005.

[8] J. Pugh and A. Martinoli, "Small-Scale Robot Formation Movement Using a Simple On-Board Relative Positioning System," in *International Symposium on Experimental Robotics 2006*, 2006. [Online]. Available: http://www.grasp.upenn.edu/iser06/

[9] N. Kottege and U. Zimmer, "Relative localization for AUV swarms," in *International Symposium on Underwater Technology*, 2007.

[10] F. Mondada, L. Gambardella, D. Floreano, S. Nolfi, J. Deneuborg, and M. Dorigo, "The cooperation of swarm-bots: physical interactions in collective robotics," *IEEE Robotics & Automation*, vol. 12, no. 2, pp. 21–28, 2005.

[11] J. Allred, A. Hasan, S. Panichsakul, P. G. W. Pisano, R. H. J. Huang, D. Lawrence, and K. Mohseni, "Sensorflock: an airborne wireless sensor network of micro-air vehicles," in *SenSys*. ACM, 2007, pp. 117–129.

[12] Z. Wang, Z. Song, P.-Y. Chen, A. Arora, D. Stormont, and Y. Chen, "Masmote - a mobility node for mas-net (mobile actuator sensor networks)," *Robotics and Biomimetics*, pp. 816–821, 2004.

[13] Z. Wang, P. Chen, Z. Song, Y. Chen, and K. Moore, "Formation control in mobile actuator/sensor networks," vol. 5804, no. 1, 2005, pp. 706–717. [Online]. Available: http://link.aip.org/link/?PSI/5804/706/1

[14] K. Kodagoda, W. Wijesoma, and E. Teoh, "Fuzzy speed and steering control of an agv," *Control Systems Technology, IEEE Transactions on*, vol. 10, no. 1, pp. 112–120, Jan 2002.

[15] N. Hodge, L. Shi, and M. Trabia, "A distributed fuzzy logic controller for an autonomous vehicle," *J. Robot. Syst.*, vol. 21, no. 10, pp. 499–516, 2004.

[16] H.-H. Chiang, L.-S. Ma, J.-W. Perng, B.-F. Wu, and T.-T. Lee, "Longitudinal and lateral fuzzy control systems design for intelligent vehicles," in *ICNSC*, 2006, pp. 544–549.

[17] D. Titterton and J. Weston, *Strapdown Inertial Navigation Technology, 2nd Edition*. Institution onf Electrical Engineers, 2004.

[18] S. J. Henkind and M. Harrison, "An analysis of four uncertainty calculi," *IEEE Tran. on Systems, Man and Cybernetics*, vol. 18, no. 5, pp. 700–714, 1988.

[19] M. Marin-Perianu and P. J. M.Havinga, "D-FLER: A distributed fuzzy logic engine for rule-based wireless sensor networks," in *UCS*, 2007, pp. 86–101.

[20] V. N. S. Samarasooriya and P. K. Varshney, "A fuzzy modeling approach to decision fusion under uncertainty," *Fuzzy Sets and Systems*, vol. 114, no. 1, pp. 59–69, 2000.

[21] J. Bih, "Paradigm shift - an introduction to fuzzy logic," *IEEE Potentials*, vol. 25, no. 1, pp. 6–21, 2006.

[22] E. H. Mamdani and S. Assilian, "An experiment in linguistic synthesis with a fuzzy logic controller," *International Journal of Man-Machine Studies*, vol. 7, no. 1, pp. 1–13, 1975.

[23] "Ambient Systems," http://www.ambient-systems.net.

[24] T. Hofmeijer, S. Dulman, P. Jansen, and P. Havinga, "AmbientRT - real time system software support for data centric sensor networks," in *ISSNIP*, 2004, pp. 61–66.

[25] "STMicroelectronics," http://www.st.com.

[26] "PNI Corporation," http://www.pnicorp.com.

[27] M. Caruso, T. Bratland, C. Smith, and R. Schneider, "A New Perspective on Magnetic Field Sensing," *Sensors*, vol. 15, no. 12, pp. 34–46, 1998.

[28] M. J. Caruso, "Applications of magnetic sensors for low cost compass systems," 2000, pp. 177–184.