Design Experiences in Minimalistic Flying Sensor Node Platform through SensorFly

XINLEI CHEN, AVEEK PUROHIT, SHIJIA PAN, CARLOS RUIZ, JUN HAN, ZHENG SUN, FRANK MOKAYA, PATRICK TAGUE, and PEI ZHANG, Carnegie Mellon University

Indoor emergency response situations, such as urban fire, are characterized by dangerous constantly changing operating environments with little access to situational information for first responders. *In situ* information about the conditions, such as the extent and evolution of an indoor fire, can augment rescue efforts and reduce risk to emergency personnel. Static sensor networks that are pre-deployed or manually deployed have been proposed but are less practical due to need for large infrastructure, lack of adaptivity, and limited coverage. Controlled-mobility in sensor networks, that is, the capability of nodes to move as per network needs can provide the desired autonomy to overcome these limitations.

In this article, we present SensorFly, a controlled-mobile aerial sensor network platform for indoor emergency response application. The miniature, low-cost sensor platform has capabilities to self deploy, achieve three-dimensional sensing, and adapt to node and network disruptions in harsh environments. We describe hardware design trade-offs, the software architecture, and the implementation that enables limited-capability nodes to collectively achieve application goals. Through the indoor fire monitoring application scenario, we validate that the platform can achieve coverage and sensing accuracy that matches or exceeds static sensor networks and provide higher adaptability and autonomy.

CCS Concepts: • **Information systems → Sensor networks**; • **Computer systems organization → Embedded hardware**;

Additional Key Words and Phrases: Mobile sensor networks, aerial networks, hardware design

ACM Reference format:
Xinlei Chen, Aveek Purohit, Shijia Pan, Carlos Ruiz, Jun Han, Zheng Sun, Frank Mokaya, Patrick Tague, and Pei Zhang. 2017. Design Experiences in Minimalistic Flying Sensor Node Platform through SensorFly. *ACM Trans. Sen. Netw.* 13, 4, Article 33 (November 2017), 37 pages.
https://doi.org/10.1145/3131779

1 INTRODUCTION

Indoor emergency response scenarios, such as urban fire, earthquakes, gas leaks, or hostage situations, are characterized by dangerous and constantly changing operating environments for first responders. Rescue personnel have little prior information about the scene as well as adverse conditions, such as smoke or structural collapse, that may impede the planning and co-ordination efforts.

Authors' addresses: X. Chen, A. Purohit, S. Pan, C. Ruiz, J. Han, Z. Sun, F. Mokaya, P. Tague, and P. Zhang, CMU SV NASA AMES Research Park Building 23, Moffett Field, CA 94035; emails: xinlei.chen@sv.cmu.edu, {aveek.purohit, shijia.pan}@west.cmu.edu, {carlos.ruiz.dominguez, jun.han}@sv.cmu.edu, zhengs@ece.cmu.edu, Frank.Mokaya@sv.cmu.edu, patrick.tague@west.cmu.edu, peizhang@cmu.edu.

https://doi.org/10.1145/3131779

Deployments of wireless sensor networks have been proposed to improve the effectiveness of rescuers. Sensor networks can provide valuable fine-grained real-time environmental information [23, 35, 46]. For example, the temperature profile within a building structure on fire can be used to predict the fire's propagation through the building. Such information could potentially enable rescuers to anticipate the evolution of an emergency and adopt effective strategies to minimize injury, loss of life, and damage to property. The very low-cost of sensor nodes has been envisioned to enable practical large-scale deployments in such circumstances and provide enough redundancy to survive these harsh environments.

Previously proposed sensor networks have largely been static in nature [17, 20, 41]. The sensor nodes either must be pre-installed as part of an infrastructure or be deployed manually by first responders. As a result, these static node approaches require the following improvements for widespread adoption and utility:

- **Less infrastructure and maintenance.** In most approaches, a pre-installed sensing infrastructure is assumed to be in place. The cost of universally creating (placing of nodes) and maintaining (battery replacement) such infrastructure remains high.
- **Adaptability and robustness.** The harsh environment of emergency scenarios makes sensing nodes susceptible to damage and failure. A pre-installed static network infrastructure cannot effectively adapt to the destruction of parts of the network and requires high redundancy.
- **Adaptive spatial coverage.** With only a limited number of sensors, spatial coverage of sensed data is also restricted at deployment time. Repositioning or re-tasking nodes as per situation is not possible.

Recent literature has proposed the vision for how mobile sensors or *mobiscopes* can be used to monitor human spaces [4]. One envisioned idea is that of actuated or controlled mobility. Controlled mobility enables a network to mobilize its nodes to suit its demands. Such needs could involve tasks such as data gathering or maintaining network connectivity. Since the network can deploy autonomously, it can replace faulty nodes or reorganize as per its application requirements, addressing many limitations of static sensor networks. Moreover, considering that no universal sensing infrastructure must be installed and maintained, a much larger number of devices can be deployed economically. Sensing nodes like these that combine mobility with low-cost large-scale deployments can provide effective solutions for information gathering in emergency response scenarios.

While robotic platforms [6, 13] exist, they are unsuitable for use as *mobile carriers* for sensor nodes in indoor emergency response scenarios. This is because robotic platforms suffer from the following limitations:

- Single or monolithic robot platforms do not provide the robustness and coverage of highly distributed sensor networks. Furthermore, most robot platforms require sensors such as laser range finders or GPS for navigation, making them considerably more expensive than traditional sensor nodes and uneconomical for large or expendable deployments.
- Second, most existing swarm robot platforms are ground based. Ground-based robots do not allow for three-dimensional (3D) sensing as well as have limited reach. Existing micro-aerial vehicles [30] or networked unmanned aerial vehicles [5], apart from their high cost, have large form factors that is ill-suited to indoor operation and hazardous to human occupants.

An aerial controlled-mobile sensing platform, therefore, provides a better alternative for indoor emergency response. Designing such a platform, however, raises the unique challenge of realizing a low-cost, small form-factor device capable of autonomous flight and collaborative sensing.

The low-cost and small-form factor requirements limit the capabilities of individual nodes, making traditional robotics approaches less applicable. We contend that such a lightweight platform must take a network-centric view to collectively realize complex application goals with its simple sensing, navigation, and processing abilities.

1.1 Contributions

In this article, we present SensorFly, a controlled-mobile aerial sensor network for monitoring applications in indoor emergency scenarios. The design, modeling, and physical simulation of SensorFly are proposed. The main contributions of the work are:

(1) We discuss the experiences on design consideration through evolution of SensorFly: (1) how to obtain more physical separation and shielding between sensitive components; (2) how to increase the lift of the drone; (3) how to increase stability of the drone when in flight. The details will be discussed in Section 3.
(2) We propose the modeling of the navigational sensors used in SensorFly under real operation conditions. Their noise profile due to the miniature form-factor and flying motion is experimentally analyzed, and on-board computationally inexpensive noise filtering mechanisms are presented. Specifically,
   - We analyze the effect of the motors on the and design a filter to minimize the effect of the magnetic field distortions.
   - The nature of the noise due to motion and environmental effects is analyzed and a filter is designed to discard erroneous readings.
   - We present an accelerometer-based obstacle sensor, which eliminates the need for special infrared or ultrasonic obstacle detectors that do not meet the weight constraints of the node.

   The details will be discussed in Section 5.
(3) We present a physical simulation platform for drone algorithm design. The operating environment of a SensorFly-like cyber-physical system has many disparate computational and physical components that are not adequately modeled by existing robotics simulation packages. We present a simulation environment that combines a physical disaster model (indoor fire growth model), a radio path loss model, a wireless network model, and a node mobility model to comprehensively evaluate such cyber-physical systems. The details will be discussed in Section 6.

The rest of the article is organized as follows. Section 2 introduces our target application and its requirements that motivate platform design. Section 3 examines our constraints and hardware design tradeoffs. Section 4 describes our software architecture. Section 5 provides the implementation and characterization of the platforms capabilities. Section 6 compares our work with static sensor networks. Section 7 discusses related work. Section 8 presents a discussion of aspects for further study. Section 9 summarizes our work.

2 APPLICATION

The SensorFly system can be deployed in several sensing and monitoring applications, such as survivor search after earthquakes, reconnaissance in urban combat, or indoor toxic plume sensing [8, 33]. We present on the indoor fire emergency monitoring application to evaluate the effectiveness of our platform in providing autonomous, timely, and high-fidelity information to aid fire-fighter operations.

Fire response and fire rescue remain extremely challenging operations that annually claim the lives of over 100 firefighters in the United States [3]. Lack of situational information has been

identified as a critical limitation for firefighters [17]. On average, firefighters reach 61% of urban fires in 6min. However, the fire may spread extensively within that period. Firefighters have little or no knowledge of the location, extent, or advance of the fire in these situations.

Requirements. Sensors for real-time sensing and prediction of fire propagation are an active area of research. Researchers have proposed fire models to predict the advance of fire from *in situ* sensor readings.

This information can be valuable for several fire-fighting tasks:

- Firefighters can determine the extent of fire and predict its progression.
- Firefighters become aware of hazardous areas to avoid.
- Firefighters can effectively plan evacuation routes.

The fire model determines the type and spatial location of sensed data. A popular model is *Consolidated Model of Fire and Smoke Transport* (CFAST) [32]. CFAST is a zone model where each space is split into the top zone (consists of the high-temperature gases and smoke) and the bottom zone (consists of lower-temperature gases). The height of the interface between the two zones is called the smoke layer height and this layer descends as smoke builds up in the room. Structural information such as the presence of shafts or open stairways is also of interest to firefighters.

Fire monitoring sensor networks seek to provide a 3D profile of parameters, such as temperature, gas, pressure, and ceiling height in each room as inputs to such a model.

While many sensor networking systems have been proposed for fire monitoring, they are essentially composed of pre-deployed static sensor nodes [1, 17, 22, 41]. We present a detailed comparison with related work in Section 7. Such infrastructure is expensive to deploy and universal adoption remains far into the future. SensorFly nodes can be deployed at the time and location of fire. While, unlike static pre-deployed sensors, current version of SensorFly nodes cannot know their exact location and provide data from regions obstructed due to closed doorways or building structure collapse, the SensorFly system can provide valuable information where a static infrastructure is absent. Firefighters can introduce SensorFly nodes into connected spaces as they enter the building structure, without the need to physically explore every region.

Moreover, a larger number of SensorFly nodes can be deployed at the actual point of emergency as opposed to maintaining universal infrastructures. Finally, the aerial mobility allows better spatial resolution of sensing. The fire-monitoring scenario requires 3D sensing of vertical temperature profile and ceiling height in building structures. The SensorFly with its ability to hover vertically can provide higher fidelity data.

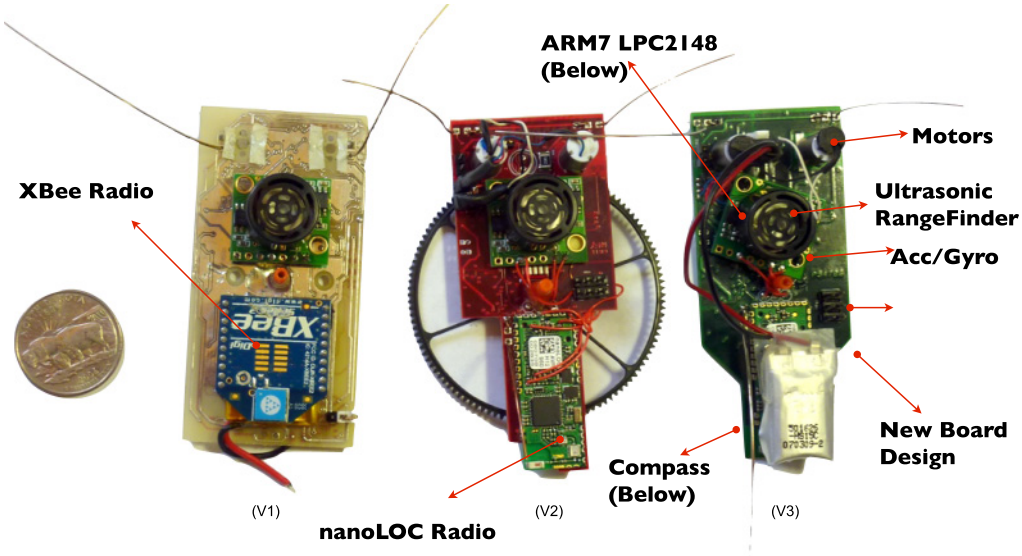
3 HARDWARE DESIGN

To demonstrate the feasibility of the SensorFly system, we have designed and built four generations of SensorFly nodes (Figure 1). This section focuses on the hardware design choices and the tradeoffs involved in building controlled-mobile aerial sensing platforms.

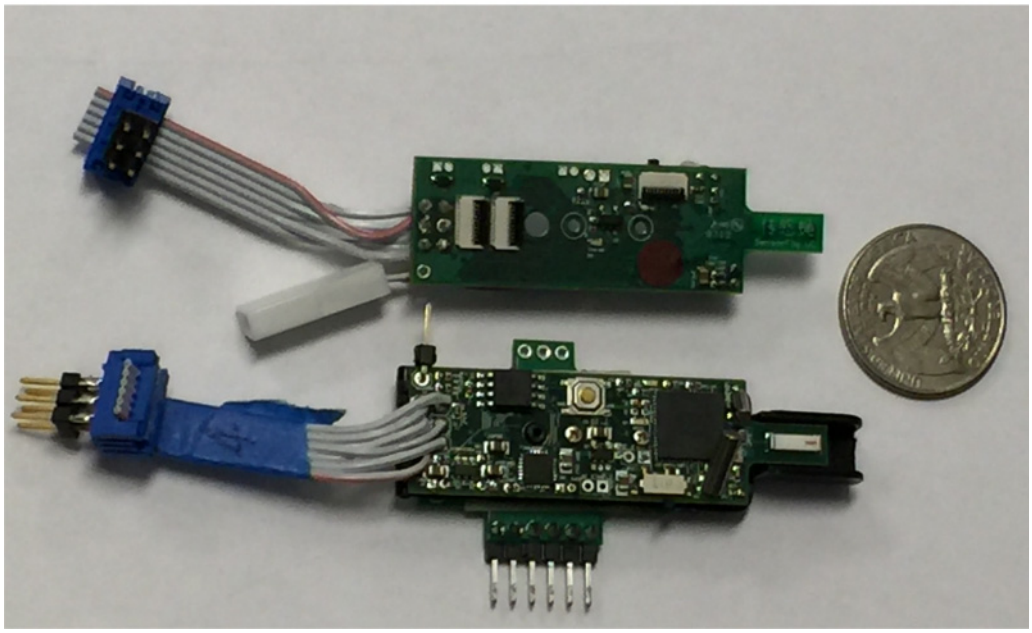
We discuss the experiences on design consideration through evolution of SensorFly: (1) how to obtain more physical separation and shielding between sensitive components; (2) how to increase the lift of the drone; (3) how to increase stability of the drone when in flight.

3.1 Key Constraints

The low-cost, low-weight aerial sensing platform presents several constraints and challenges, occupying a new and unique design space. The addition of mobility and control introduces new constraints like *weight*, *sensor interference*, and *higher noise*, to the traditional low-cost commercial off-the-shelf (COTS)-based sensor node hardware architectures. Careful consideration is required



(a) First three generations of SensorFly node design



(b) The fourth generation of SensorFly node design

Fig. 1. Four generations of SensorFly node design.

in new aspects of sensor network hardware design such as *component placement* and *weight balance*. To achieve the desired level of collective system capability given the minimum available resources of individual nodes, a delicate balance must be attained and trade-offs examined. The following factors affect our design approach:

Table 1. Weight of Several Possible Components of SensorFly Nodes

	Component	Weight
X	Drive Motors and Propeller Assembly	15g
X	130mAh Lithium Polymer Battery	4g
	200mAh Lithium Polymer Battery	7g
X	Controller Board	10g
	Camera Board Add-on	3g
	Audio Board Add-on	4g
	LED Board Add-on	3gs
	Ultrasonic Distance Sensor Add-on	4g
	Basic SensorFly Total Weight	29g
	Absolute Maximum Takeoff Weight	34g

Xs mark the components included in the base configuration of SensorFly nodes.

Cost. Low per-device cost allows us to scale up sensor node deployments. As a result, for the equivalent cost of an intelligent robot, many more sensor nodes can be used, enabling higher sensing coverage as well as discovery speed. Utilizing a low-cost flight mechanism, common to off-the-shelf RC helicopters, allows us to achieve a prototype cost of about $200. In mass production, similar RC helicopter assemblies are commercially available for about $20 [39]. While larger flying platforms such as the Parrot.AR Drone can provide better capability, they have higher production cost (~$300) and lower reach due to the bigger form-factor. The navigational capability of individual sensor nodes must be attained through low-cost COTS sensors. A trade-off must be made in forgoing accuracy for deployment scale to better realize our application objectives.

Weight. The miniature aerial platform adds a new metric, weight. The small weight enables longer flight times, greater reach, and better safety for indoor emergency response scenarios. The weight limit is decided by delicate trade-off point. Adding more weight requires bigger motors that in turn require a bigger battery. A larger craft eventually sacrifices the miniature form factor along with the mobility and scalability advantages that it provides. Table 1 shows the component-wise weight break-up of the 29g SensorFly node.

This weight constraint limits the number of sensors that can be carried. In addition, the weight must be balanced to achieve stability of the node in flight requiring careful component layout and board design.

Energy. Similar to many battery-operated systems, the SensorFly platform is highly energy constrained. Unlike most other sensor systems, however, SensorFly has many different operating modes with vastly different energy characteristics. Table 2 shows the energy consumption characteristics for some of these modes. Of all these modes, the ones involving flight are the most expensive in terms of energy usage.

This further underscores the need for a lighter node, as it enables us to reduce the power consumption of motors. The current battery and weight profile provides about 5min of airborne flying time. However, by optimizing movements for deployment and reconfiguration, and managing energy consumption when landed and sensing, the overall life of the network can be extended. Utilizing physical characteristics such as the *ground effect* can further reduce movement energy. We evaluate the flight time of the SensorFly nodes in Section 5.3.

Interference and Noise. The small form-factor requires placing sensors and components very close to each other on a miniature circuit board. At the same time, use of low-cost brushed motors

Table 2. SensorFly Node's Operation Modes and Their Power Usage Breakdown

Operation	Typical Power Usage
Mobile Mode	6.2W
Stationary Mode	
Data transmission	310mW
Data receive	330mW
Sensing only	225mW
Processing Only Mode	150mW
Idle Mode	1mW

creates large electromagnetic noise that interferes with the proper operation of the sensors. The placement of sensors and components must be done keeping in mind the effect and nature of induced noise. Additionally, software-filtering approaches are also needed to obtain better sensor readings.

3.2 Design Trade-Offs

The resource constraints and capability requirements force our component selections for the SensorFly platform. In this section, we examine our design choices and trade-offs. We also discuss the evolution of the current third-generation hardware platform, which incorporates learning from previous iterations.

Processor. The first processor used in the SensorFly node is the ARM7-based LPC2148. The micro-controller is capable of running at 60MHz and features 512KB of Flash memory as well as 42KB of RAM. It is limited in comparison to the computers that are currently used for most robotics applications. Through various iterations, we have found the processor to be capable of running both the flight control algorithms and the sensor filtering algorithms. In later versions, SensorFly incorporated a secondary external processor, an AVR AtMega644, for radio functions and control. By moving the majority of the time critical processes off-board, concurrency is handled better and application integration is simplified. This significantly reduces conflict between time critical components such as the radio and the flight controller.

Navigation Sensors. The choice of sensors requires careful consideration, due to the limit on their size and numbers. Navigational sensors have been explored extensively for robotic platforms [10]. Navigational sensors are needed to detect the motion of the node itself as well as sense the environment or other nodes. For motion estimation, miniature MEMS-based inertial sensors such as the accelerometer and gyroscope have become popular in consumer devices like mobile phones, and as a result are commercially available at low-cost. However, their susceptibility to noise is higher and a trade-off must be made against accuracy. For navigating the environment, a number of higher accuracy range-based options such as laser range finders, multiple-ultrasound sensors, camera, and lidar sensors are unsuitable for use in SensorFly. Table 3 summarizes the strengths and weaknesses of available sensors on the basis of cost, weight, and accuracy. Radio-based RF-ranging is an attractive technique, especially because the radio can also be used for communication. However, multipath effects limit the accuracy of radio ranging in indoor environments. This limits precise navigation and movement and calls for collective stochastic exploration approaches. We examine trade-offs further in Section 5. Furthermore, these sensors are re-used for multiple purposes as described in Section 3.3.

In the first version of SensorFly, a two-dimensional compass and a three-dimensional accelerometer were included as flight state sensors. Since the craft is passively stable, the

Table 3. Comparison of Navigational Sensors

Component	Cost	Weight	Accuracy
Accelerometer	Low cost COTS component	Low < 1g	Analog inertial sensor. Unreliable for distance estimation due to accumulating error. Used to detect collisions.
Gyroscope	Low cost COTS component	Low < 1g	Useful for angular velocity measurement. Unreliable for absolute angular position measurement but not affected by magnetic fields. Used for feedback to for yaw controller.
Compass	Low	Low < 1g	Low indoors due to sensitivity to magnetic fields. Error does not accumulate. Used to provide absolute heading.
Ultrasonic Ranger	Low	Medium ~4g	Fair. Depends on environmental factors such as interference and materials.
Nanotron nanoLoc RToF ranging	Low. Cost is amortized as radio is also used for communication.	Medium. (~4g)	Better accuracy than RSSI-based radio-ranging. Less accurate than ultrasound and laser range finders.
Laser Ranger	Medium	High 50g+	High. Not included due to weight constraints.
Vision	High	High	Accuracy depends on operation scenarios. Less effective in presence of smoke. Needs high processing power.

five-degree-of-freedom measurement should be enough to capture the full possible motion of the self-stabilizing platform chosen. However, the magnetic interference from the motors is large due to the nodes small physical size. This interference during flight renders the readings from the compass inconsistent and unsuitable for measuring rotation. To improve the flight controls, the later versions include a 3D compass, a three-axis accelerometer, as well as a 2D gyro. In addition, the placement and physical design of the boards mitigate the noise characteristics as described later. These sensors provide a full eight degree-of-freedom measurement. During flight, the gyros provide a rotational sensor immune from magnetic noise, while the compass provides an absolute reading.

Radio. To aid navigational needs, the current version of SensorFly uses the nanoLOC TRX transceiver module [29]. Apart from offering better performance against indoor multi-path fading effects, the radio provides inter-node range estimates based on round-trip time of flight (RToF) computations.

A Digi XBee [9] was used for the V1 of the hardware. This radio is commonly used in sensor networks. Received signal strength indicator (RSSI) was used for range estimation in V1.

Several factors prevented the success of this approach. First, in the indoor environment, the radio characteristics were extremely unpredictable, making the multi-path problem pronounced. Second, electromagnetic noise from the motor significantly increased the unpredictability of RSSI measurements. Finally, due to the physical orientations of the helicopter, the antenna cannot be placed at an omni-directional location. This increased the effect of node orientation on the RSSI. These factors prevented the use of RSSI as a viable solution for use in mobile sensor nodes.

Motors. A unique feature of the SensorFly nodes is the mechanical helicopter drives. We use two 7-mill core-less motors. These motors drive two coaxial main rotors, and are a significant source of noise in the system, as we shall explore later. While brush-less motors would provide better noise and thrust performance, their size, cost, and circuitry requirements make them ill-suited for our target application.

Furthermore, the low-cost of these core-less motors implies substantial variations in their response to input voltages as well as degradation in performance with use. However, due to the coaxial helicopter design and its passive stability, simple control algorithms can be used to counter-effect these variations. A third motor can be added to the node to provide controlled forward flight. Currently, a weight difference, created by placement of components on the board, is used to provide a constant forward motion, while the craft rotates in small circles to hover in place.

3.3 Component Reuse

An important strategy for achieving the platform's stringent weight goals is component reuse. Several elements of the node's mechanical and electronic components are selected to be capable of perform more than one function.

On the *electronic hardware* side, the Nanotron nanoLoc radio module enables communication as well as Round-trip Time-of-Flight radio ranging. This ranging capability provides a primitive form of localization and is a substitute for having laser or ultrasound range finders. Although, a trade-off is made in the accuracy of range estimation, the weight and cost constraints are impossible to meet with the alternative ranging mechanisms mentioned.

Sensors such as the accelerometer are used as obstacle sensors, for their ability to detect contact. The lightweight and robust design of the SensorFly nodes enables them to tolerate bumping into obstacles without affecting their flight performance. Thus, dedicated obstacle sensors like infrared or ultrasound-based detectors are avoided.

Amongst the *mechanical components*, the blades on the helicopter design serve to protect the body of the node from bumps. As the body is designed to be smaller than the blades, it acts as a protective buffer for the on-board electronics.

The circuit board housing the SensorFly electronics acts as the fuselage for the node. This requires careful selection of the board to provide enough rigidity and, in turn, prevent stress on the connections and traces when the node lands. At the same time, a thick board adds more weight to the node. In V1 of the design, a 20-mill double-layer board is used. While the design is a 1.6×3.1 inches square, the size reduced airflow thereby degrading the lift of the nodes. A 20-mill, four-layer circuit board shaped as a "T" was subsequently used to allow more air to flow through from the blades. Due to the reduced per-layer thickness nodes experienced higher failure rate due to stress on the metal traces caused by takeoff and landings. In the third version of SensorFly, a 30-mill board was selected, stress relief added to the "T" shape and component placement staggered to further reduce single stress points. This redesign greatly improved the fuselage strength.

The legs of the SensorFly node are built with gold plated spring wire and are used as charging terminals as well as landing supports.

Each node can have different weight distributions due to modular design of the SensorFly nodes. For example, different sensors can be added to the extension port of the SensorFly. To counter this

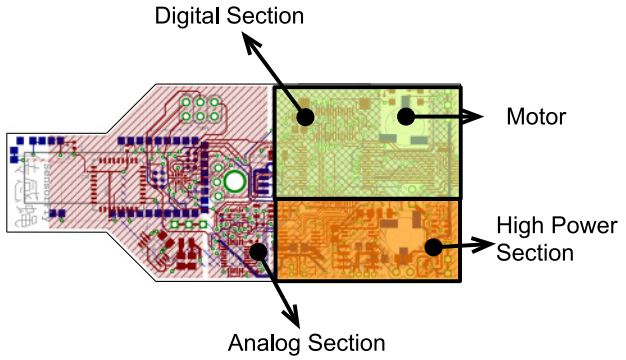


Fig. 2. The SensorFly node layout. Analog sensors are isolated and the compass placed away from the motors to reduce interference. The digital, analog, and high-power sections have separate power supply and the ground plane is interconnected through ferrite beads to filter noise.

Table 4. The Design and Performance Improvement on SensorFly Version 4

Design Improvement	Performance Improvement
Modular board design	• Easier replacement and update of the hardware. • Easier customization for equipping SensorFly nodes with different sensors.
Stacked Sensor platform	• More air to follow through to achieve similar flying time with higher payload • Increasing the lift of the drone
Sensor board hanged lower	• Lower center of mass for more stable flight (24g)

effect on the stability of the craft, the battery and the ultrasonic sensor act as adjustable counterweights and are positioned so as achieve the desired node balance.

3.4 Board Layout

The SensorFly board layout involves careful consideration of the noise characteristics of the components and their weight. The analog sensor components such as the accelerometer and gyroscope must be placed so they are isolated from the noise sources. The main source of noise in the platform is the high power source and the pair of brushed motors causing high electromagnetic interference. Similarly, the compass must be isolated from the motors as it is adversely affected by their rotation magnetic field.

Figure 2 shows the layout of the SensorFly board for versions 3 and 4. In version 3 the board is divided into three sections. The motors occupy the right (front) end, along with digital section, since the digital components are least affected by the EMI. The 3D compass is placed towards the tail of the node to minimize the magnetic effect of the motors. The left-most section is the analog section, which houses sensors, such as the gyro and accelerometer that are adversely affected by noise in the power-lines due to the motors back-emf. The high-power section that feeds the motors occupies the bottom-right corner of the board. Each section has separate power lines and a separate ground plane. These are interconnected through ferrite beads to filter out noise.

Compared to the previous versions, the major improvement design of SensorFly version 4 happens in hardware part, as shown in Table 4. (1) The sections were separated into multiple boards

with ground plans placed in between to further reduce noise. This allows more physical separation and shielding between sensitive components. The antenna is placed at the tip of the platform to minimize interference. The processing board is stacked closest to the motor, because it is least sensitive to electrical magnetic noise. The sensing board, on the other hand, is stacked at the furthest layer to avoid interference from these noise. The modularity also allows for updating part of the hardware and allows easier hardware customization for equipping SensorFly nodes with different sensors. (2) The sensor platform is stacked and has a lower vertical profile. This allows more air to flow through, thus increasing the lift of the drones. (3) Another consideration in the placement of sensors is their weight. The board weight must be balanced for stability of flight. The sensor board is hanged lower, which lowers the center of mass and increases stability of the drone when in flight. A slightly larger weight is maintained towards the front of the node, to substitute the tail-blade of the helicopter design, and enable forward movement. The 4g battery and 4g ultrasound height sensor are used as adjustable counterweights to balance the weight of the node. The battery is placed toward the tail of the node while the ultrasound sensor is placed in the middle to counteract the weight of the motors in the front.

3.5 Extensibility

Several types of sensors can be added to each node using the provided expansion ports, in accordance to the needs of different applications. The expansion port supports serial, SPI, 10-bit parallel, and I2C for sensor interconnection, as well as provides regulated and unregulated power pin through the node battery. Due to weight constraints, we plan to only include limited additional sensor module per craft to maintain weight characteristics of the craft. Several sensors have been designed, including the camera, speaker, microphone, and infrared detectors. We plan to explore the use of additional sensors as our research progresses.

4 SOFTWARE ARCHITECTURE

The SensorFly is designed to provide a platform for controlled-mobile and collaborative sensing for emergency response. The SensorFly system architecture, shown in Figure 3, consists of the firmware, the node-level system software, customizable network level services, and user application layers. The node system software consists of three major modules corresponding to the capabilities of the platform:

- **The sensor controller** provides access to on-board sensors and expansion ports. Includes filtering modules to mitigate noise caused by motors and motion.
- **The network controller** provides peer-to-peer aggregation and broadcast communication, with support for inter-node range estimation through RToF.
- **The flight controller** provides a high-level navigation API for hover, turn and single-direction flight. A biased random-walk dispersion and exploration algorithm is implemented utilizing the node ranging capability. The algorithm enables nodes to navigate and deploy in unknown environments without need for localization.

4.1 Sensor Controller

The sensor controller provides access to the on-board sensors that include the ultrasonic altitude sensor, three-axis accelerometer, two-axis gyroscope, and a 3D electronic compass, as well as to the sensor expansion port. The module provides an API for querying sensors, setting repetitive sample rates, as well as provides in-built filters for noise reduction.

The motors and miniature form factor of the SensorFly nodes affect the performance of sensors such as the compass, as described in Section 3. Moreover, some sensors have inherent noise

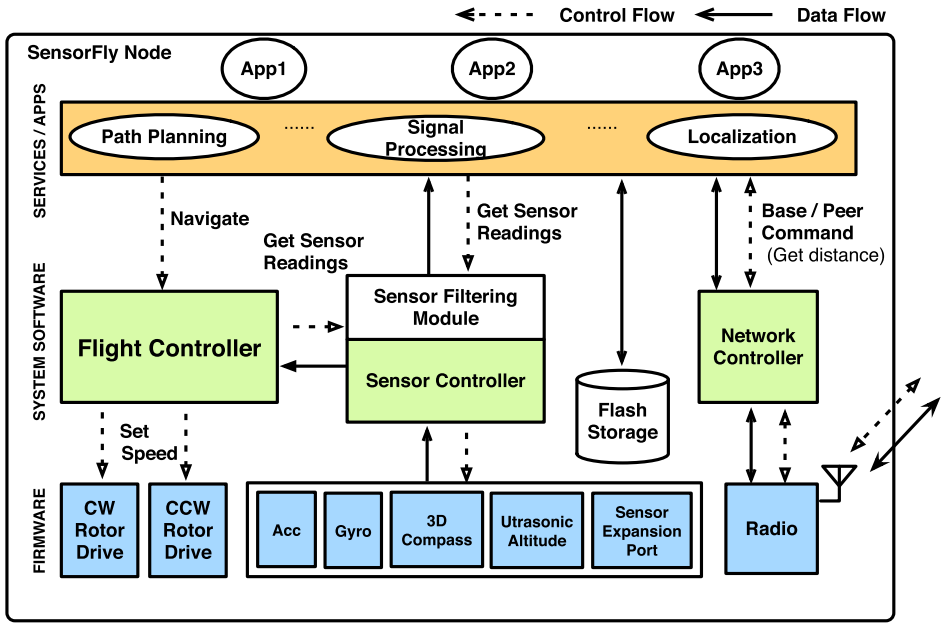


Fig. 3. SensorFly Node Architecture.

characteristics that can be filtered with knowledge about the dynamics and mobility of the node. Thus, sensor filtering must be performed for achieving useful capabilities such as altitude control and pose estimation.

The ultrasonic range finder used to measure the altitude of the SensorFly node is affected by several environmental factors such as absorption characteristics of the ground and interference from other sources such as fluorescent lamps. The sensor controller utilizes the vertical motion dynamics of our platform to discard erroneous readings, using a first-order recursive digital filter, as described in Section 5.

Similarly, the SensorFly has a three-axis electronic compass [15] for direction sensing. This is useful in estimating the pose of the node. However, the small dimensions of the SensorFly node require the compass to be placed close permanent magnet DC motors that distort compass reading. Analysis of error induced by the motor's moving magnetic field points to a symmetric distribution that can be filtered out to a large extent through a moving average filter implemented within the sensor controller module. The window size and other filter parameters are tuned through empirical analysis of sensor data, as detailed in Section 5.

The sensor controller also provides a virtual sensor for detecting obstacles using an accelerometer. An algorithm based on thresholds is able to distinguish bump events from the acceleration signal vector magnitude from normal flight.

4.2 Network Controller

Our networking implementation supports two major capabilities namely peer-to-peer data communication and radio-based inter-node ranging. The SensorFly has a dedicated AVR AtMega644 microcontroller for radio control. This enables better handling of packet transmit-receive and ranging operations that require timely processing. Especially, since flight control is the highest priority task on the primary microprocessor. The radio module, that is, the nanoLOC transceiver

and the AVR micro-controller, are connected via UART to the primary ARM7 LPC2148 microprocessor. The network controller implements the UART communication protocol, message queues for inbound and outbound packets, and provides a high-level API for sending, receiving, and forwarding data.

4.2.1 Data Communication. The network protocol for the mobile SensorFly nodes essentially consists of an aggregate and broadcast communication model. The monitoring network seeks to route all sensed data to the base station. Additionally, due to the constant motion of nodes, establishing routes and running explicit node discovery service is impractical. Nodes therefore periodically broadcast messages containing their sensor data. Neighboring nodes, on hearing the broadcast message, aggregate the node's sensor data with their message.

Each node's sensor data consists of a sequence identifier and a time-to-live field. The time-to-live is decremented with the number of hops as well as on the expiration of a local time window, to control the time for which stale data propagates in the network. A node's data is propagated by other nodes only if the time-to-live is still not zero. Old sensor data from a node is replaced with fresh data, if it is received before the expiration of the time-to-live field. This scheme is akin to a controlled reverse-flood of data to the base station.

4.2.2 Ranging. The network controller also provides an API for node-to-node range estimation. The range estimates are used in the exploration algorithm currently employed by SensorFly nodes, which consists of biased-random walks [26] that disperse nodes away from each other based on their distance from each other. Inter-node ranging is a primitive used by topology estimation schemes.

The radio has the capability to compute distance using a round-trip time-of-flight (RToF)-based technique called Symmetric Double Sided Two Way Ranging (SDS-TWR) [28]. The round-trip time-of-flight method measures the elapsed time between the host node sending a data signal to the remote node and receiving an acknowledgment from it. Using the estimated speed of propagation of a typical signal through a medium and the signal turnaround time, that is, the time for the remote node to send out an acknowledgement packet, the host computes the distance from the remote node. Using physical layer timestamps and hardware-generated acknowledgments, the nanoLOC TRX radio achieves a predictable turnaround time.

Unlike other time-of-flight methods, this method does not require tight clock synchronization between nodes. The time elapsed is computed from timestamps of individual nodes themselves. This removes the need for extra hardware for global time synchronization, which is the source of complexity and higher cost in other systems. Likewise, no special antenna arrays are required such as angle-of-arrival ranging methods. We perform an experimental evaluation of the ranging performance in Section 5.1.

4.3 Flight Controller

A SensorFly's miniature helicopter flying mechanism has many advantages like the ability to take-off, land, and turn in confined indoor spaces, maximizing sensing coverage. Realizing and controlling a helicopter-based sensor-networking platform presents many interesting aspects.

On one hand, the helicopter has highly coupled dynamics. Prior autonomous helicopters [16] have required more accurate feedback sensors and computationally expensive algorithms for precise control. On the other hand, the sensor-networking platform has a single CPU with limited computation (60MHz) and memory (42Kb) resources that must perform sensing, control and network processing tasks. Besides, as described before, the performance of control strategies is limited by sensor noise, attributed to node form factor, and cost constraints.

The SensorFly overcomes these challenges by using a lightweight damage-resistant node design and by sacrificing precise control and navigation. The SensorFly system is designed to approach tasks as a networked group that achieves system-wide objectives while tolerating errors in individual node motion. The robust design ensures that nodes can collide with obstacles and still be able to fly, removing the need for precise obstacle avoidance. In fact, the nodes detect obstacles through contact. This allows the flight controller component to implement computationally inexpensive proportional-integral-derivative (PID) control loops that provide *good enough* stability and utilize a biased-random walk approach to navigation.

The flight controller provides a high-level navigation API with commands for hovering, turning, and moving forward. The following sections briefly describe the node dynamics, the navigation and exploration approach used, and the control algorithms.

4.3.1 Exploration and Navigation. The fire-monitoring scenario required a group of resource constrained SensorFly nodes to explore an unknown environment. No assumptions can be made regarding the availability of localization beacons or infrastructure in the indoor environments, for nodes to estimate their location. Detailed and updated maps of building structures may not always be available to the firefighters. Moreover, the harsh environments with smoke and fire make sophisticated vision-based navigation capabilities ineffective. Thus, SensorFly nodes utilize their low cost and large numbers to spread out and maximize coverage through a biased random walk-based exploration algorithm, similar to that presented and analyzed by Morlok et al. [26].

The SensorFly nodes hop, that is, takeoff, fly for certain duration, and land in the operating arena. The direction of hop is chosen randomly. For increasing coverage, the random walk is biased toward moving away from other SensorFly nodes while still maintaining connectivity. When a SensorFly node lands, it estimates its distance from other nodes in its vicinity. If a node exists within a *coverage range*, then the node resumes its random motion. If no node exists in the *coverage range*, then the node deploys and continues to sense data. In addition, if a node finds no other nodes within a *connectivity range*, then it resumes it's random motion. This is to prevent networks from being partitioned. Nodes can hover at desired heights as per application sensing needs. The evaluation of this approach for fire monitoring is provided in Section 6.

The previous exploration algorithm requires the capability of nodes to takeoff, control their altitude, follow a random path, and land safely. The SensorFly node design and design choices enable these capabilities to be attained in a computationally efficient fashion.

4.3.2 Altitude and Yaw Control. The SensorFly uses a kind of co-axial counter-rotating dual rotor design, which is passively stable for hover and forward flight [27]. This reduces the number of sensors and computing power required to stabilize it. In addition, this configuration and SensorFly's low weight allow the rotors to operate at relatively low RPM compared to conventional rotors, making them safer for indoor operation.

The control of the coaxial-helicopter-based platform is simple compared to other helicopter design designs, with altitude and yaw being the controllable entities. A constant weight bias towards the front of the node enables it to move in the forward, when the yaw is held constant. The main features of controlling the node are:

- Altitude control is attained by controlling the speed of the two main rotors of the node.
- Yaw control, that is, turning the helicopter from side to side is achieved by increasing the speed of one rotor and reducing the speed of the other rotor by the same amount.
- Forward flight of the helicopter is attained by placing the center of gravity towards the front of the aircraft. The main rotors follow the tilting of the node body and pull the helicopter forward. To hover, the helicopter rotates in small circles.

Fig. 4. The 29g SensorFly node hovering in a hallway.

This flight controller provides SensorFly nodes with the capability to takeoff and maintains altitude, through a PID controller designed from first principle dynamic models of the node and empirical tuning. Similarly, another PID control loop is implemented for controlling the spin of the node and maintaining pose to achieve forward flight.

5 HARDWARE CHARACTERIZATION

This section describes the implementation and characterizes the distinctive characteristics of the SensorFly platform, in context of the fire monitoring application. Specifically, the altitude sensing, pose estimation, ranging, and flight performance is detailed and evaluated. Figure 4 shows the SensorFly node hovering using the height sensor and algorithms described bellow.

We propose the modeling of the navigational sensors used in SensorFly under real operation conditions. Their noise profile due to the miniature form-factor and flying motion is experimentally analyzed, and on-board computationally inexpensive noise filtering mechanisms are presented. Specifically:

- We analyze the effect of the motors on the and design a filter to minimize the effect of the magnetic field distortions.
- The nature of the noise due to motion and environmental effects is analyzed and a filter is designed to discard erroneous readings.
- We present an accelerometer-based obstacle sensor, which eliminates the need for special infrared or ultrasonic obstacle detectors that do not meet the weight constraints of the node.

5.0.1 Ultrasound. It is essential for SensorFly nodes to stay passively stable and reduce the timeliness bounds for any network-dependent navigation or control inputs. Therefore, the SensorFly nodes require an autonomous and stable hover capability that is independent of network layer services such as localization. For example, relying on a radio location service to obtain feedback for altitude control would require extremely low latency (~1ms) ranging and communication. This is extremely difficult in a resource-constrained sensor network with low-power radio links. Therefore, we use a LV-MaxSonar-EZ1 [25] ultrasonic range finder mounted below the node fuselage, to measure the node's altitude from ground and provide feedback for our control algorithms.

However, the motion of the SensorFly node causes degradation in the performance of the sensor. Figure 5(a) shows the variation in measurements at different known altitudes from the ground. The measured values have noise in the form of short duration positive and negative pulses. Therefore, using only raw sensor data for control, the node shows large vertical oscillations during hover.

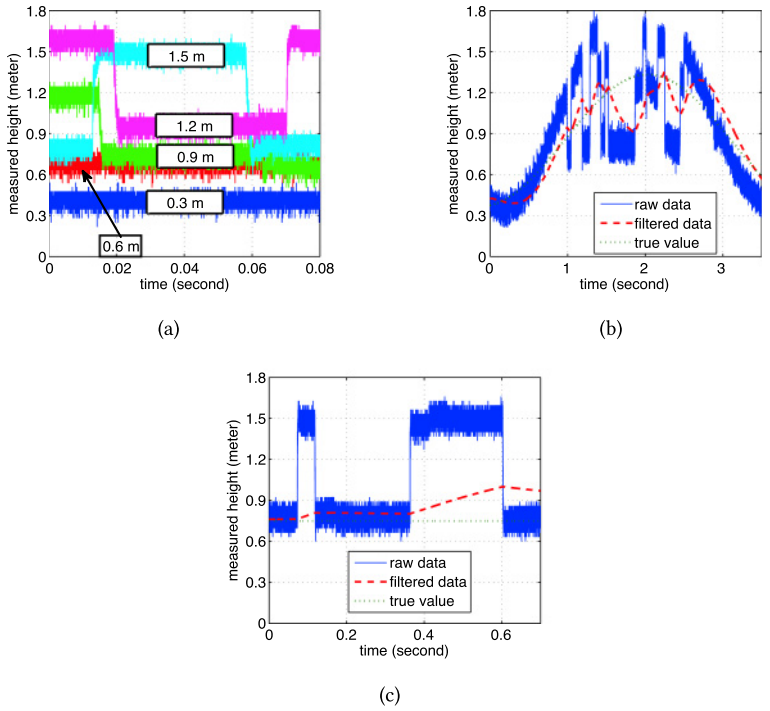


Fig. 5. The noise characteristics of the sonic altitude sensor, and our filtering techniques are illustrated, which provide feedback for autonomous hover control. (a) Shows the measured value at different known altitudes. (b) Shows measured value, when the helicopter moves up and down. (c) Shows the smoothing effect of the filter for a sample noise pulse. Our filtering technique reduces root mean squared error by 43%.

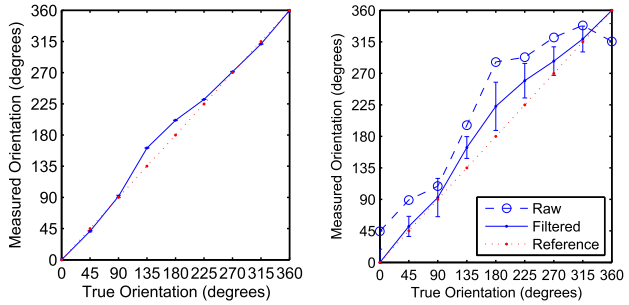
Apart from echoes, the noise can be attributed to many factors. First, the node motion causes short duration vibrations or *roll*, which changes the direction of the sensor, giving incorrect altitude readings. Second, the ultrasound-based sensor is affected by materials in the environment, for example, soft surfaces such as carpets absorb ultrasound waves and distort distance measurements. Other sources of ultrasound noise in the environment, such as computers and lamps, also have seemingly random cumulative effects.

As a result, it is difficult to use statistical techniques to mitigate its effect, since the noise is a characteristic of node movement in an unknown external environment. The sensor controller utilizes the vertical motion dynamics of our platform to discard erroneous readings. Our control algorithm limits the rate at which the platform can gain or lose altitude through maximum thrust limits on the motors. Utilizing this information, the sensor controller implements a first-order recursive digital filter to smooth out the abrupt changes in the sensor measurements. The filter is simple and computationally efficient enough to be employed at even 1KHz sampling rates. In our system, since the control module runs at a frequency of 1KHz, we adopt 1KHz sampling rate for the filtering.

In the general form, the filter can be expressed as

$$y_n = ax_n + by_{n-1}, \tag{1}$$

where y_n and x_n, are, respectively, the filter output and input raw sensor reading for the nth iteration, while a and b are the parameters of the filter.



(a) Motor 0% Duty Cycle (b) Motor 100% Duty Cycle

Fig. 6. The plot shows compass measurements versus true direction values at different motor duty cycles. (a) Shows compass measurements when motor is stopped. The average measurement error is under 5°. (b) Shows compass measurements at 100% motor duty cycle. The raw compass readings are the raw data from compass without filtering, which is shown in dashed line. Single instance raw compass readings with a maximum error of 92.1°. The continuous line shows filtered readings with a 20-reading (2s) moving window and the error bars represent the standard deviation of each measurement from the filtered results. Filtering reduces maximum error by almost 10× to 9.8°.

The parameters a and b must be computed such that the filter can distinguish between real altitude change and noise. We determine these values through an experimental setup, which involves moving the node up and down, between altitudes of 0.3m and 1.5m from the ground, at the typical maximum velocity. Logging the measured altitude values, we heuristically tune the filter to give minimum mean squared error with respect to true altitude, giving parameter values of $a = 0.513$ and $b = 0.487$. Figure 5(b) shows the true altitude, raw measured values and the filtered values for one experiment run.

Figure 5(c) shows the performance of our simple yet effective filter in smoothing out a erroneous pulses at the true altitude of 0.8m. With the filtering technique, average error reduces by 43%. The abrupt changes in altitude feedback reduce to the order of 0.2m, leading to significantly lower vertical oscillations during hover.

5.0.2 Compass. To offer direction information, a three-axis electronic compass [15] is used on SensorFly to provide an absolute measurement. The measurement is independent of both indoor radio multipath propagation effects, and inertial motion measurement errors. Direction sensing augments the SensorFly platform's radio-based range estimation capability for implementing higher accuracy localization protocols. For radio-based localization systems, the compass can thus be used for providing direction bounds on subsequent location of mobile nodes [21]. In inertial measurement systems, based on integrating accelerometer or gyroscope readings, the compass can be used as a reset to prevent error accumulation [45].

However, the motors produce a magnetic field and distort compass measurements. This is because the small dimensions of the SensorFly node require the compass to be placed in close proximity to the permanent magnet DC motors. In addition, the motion mechanism of the SensorFly nodes also affect the electronic compass.

Through experimentation, we observe two components of motor-induced measurement noise.

- When the motors are off, the permanent magnets produce a constant field distortion. This effect can be negated through a calibration routine, which is built into the compass module. Figure 6(a) compares the measured values to the true values for motor input voltage

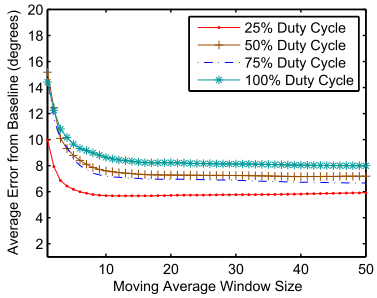


Fig. 7. This figure shows the compass average measurement error at different filter(moving average) widow sizes, for different motor input voltages. A 20 reading (2s) window provides a tradeoff between latency and accuracy.

duty cycle of 0% (motor stopped). The compass measurements are within 5° of the true value.

- When the motors are running, they induce a changing magnetic field. Moreover, the use of inexpensive brushed motors causes sparks and unpredictable spikes in compass readings.

The sensor controller provides a moving average filter to offset the effects of the motors on direction measurements.

Figure 6 compares the measured values to the true values, at various motor input duty cycles. Error bars represent the standard deviation from the mean for a set of 200 measurements obtained at each location. The error increases significantly at higher motor speeds. At 100% duty cycle, illustrated in Figure 6(b), raw readings have errors up to 92.1°. However, the mean of each set of readings provides a sufficiently accurate estimate of direction. Considering the mean, the maximum error is 24.7° at 100% motor input duty cycle, that is, full motor speed. A moving average filter therefore provides a significant reduction in compass errors.

The choice of the filter window size should be chose carefully. On the one hand, large N brings "smoother" results. This is because a moving average filter can be equivalent with a low pass filter and larger N means lower cut-off frequency. On the other hand, larger N also bearings on the latency of the control loop feedback. We evaluated the performance of filters at different window sizes to determine a suitable trade-off point. We obtained 200 readings each for eight different orientations at 0–100% motor duty cycles. Since the sampling rate of the compass is 10Hz, the total time length for 200 readings is 20s.

Figure 7 shows the average errors for filters with different window sizes and at different duty cycles. Four different line types are used to represent four different duty cycles. When the moving average window size increase from 0 to 10, the average errors decrease fast for all duty cycles. When the moving average window size is between 10 and 20, the errors of low duty cycles (25%) are very stable while the errors of the other three slightly decrease. When the moving average window size is larger than 20, all four lines are very stable. Therefore, a 20-readings or 2s moving average filter represents a tradeoff between accuracy and latency for compass measurements and provides an average error of 9.8° at maximum motor duty cycle. This is used as the default parameter for the filter.

Metallic objects or environmental conditions can distort the magnetic field at certain indoor locations. However, as all nodes obtain similar headings at any given location, the compass measurement is still useful as a location signature. In addition, the compass is very sensitive to the position of the battery and its connecting power lines. The battery is thus affixed at the same

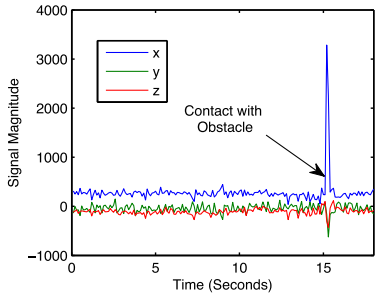


Fig. 8. We use physical collisions to detect obstacles using accelerometer data. The profile at 15s shows a contact with an obstacle in the x-y plane.

standard location for all nodes, and their compasses calibrated so they provide consistent readings. The compass is thus compensated for the magnetic field distortion produced by the platform itself.

5.0.3 Obstacle Sensor. Obstacle detectors are essential for SensorFly platform. However, infrared or ultrasonic obstacle detectors do not meet the weight constraints of the SensorFly node. To deal with this problem, the sensor controller provides a virtual sensor for detecting obstacles using an accelerometer. The low weight and robustness of the node make it immune to collision damage with most stationary obstacles.

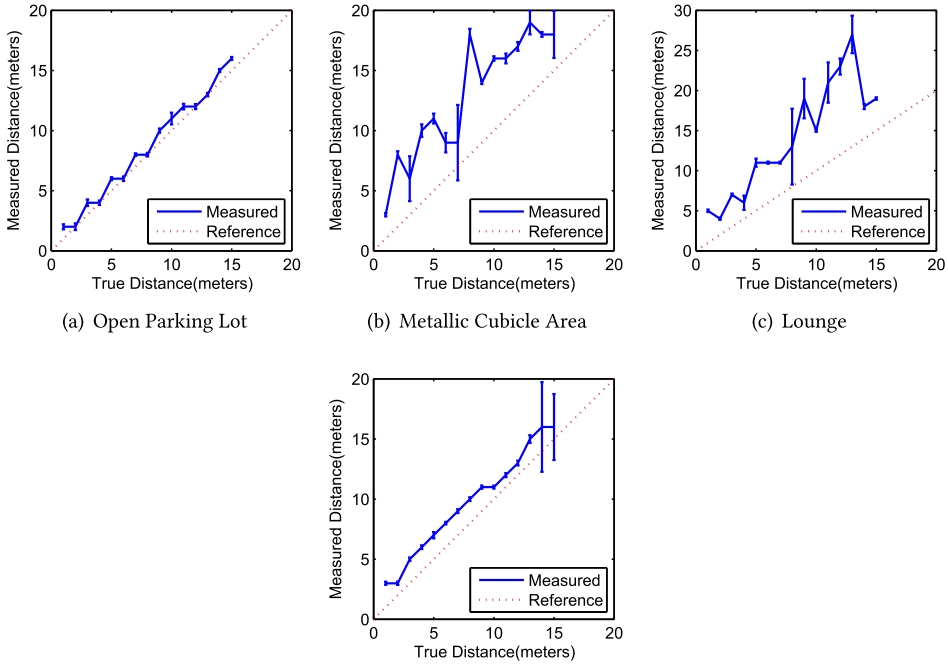
To detect contact with an obstacle, we use data from the accelerometer sensor sampled at 10Hz. An algorithm based on thresholds is able to detect bump events from the acceleration signal vector magnitude. Figure 8 shows a hovering node bumping into an obstacle, horizontally, in the accelerometer's x-y plane. The peaks in the plot show bump events and are easily discernible from accelerometer readings obtained during unobstructed flight.

5.1 Ranging

Radio ranging enables us to attain navigation capabilities, while at the same time, meet our weight and cost constraints. However, our indoor and mobile operating environments introduce multi-path and Non-Line-of-Sight (NLoS) errors. These errors are a feature of the specific space configuration near the node's location and a general model cannot be assumed. Thus, to characterize the accuracy of ranging obtainable, we evaluate the SensorFly node radio ranging in typical operating scenarios.

We performed range measurement tests at four different locations—three indoor and one outdoors. The indoor locations included a metallic cubicle area, a hallway, and a classroom with furnishings representative of multi-path rich RF propagation environment. An empty parking lot was selected for the outdoor test, to minimize the effect of reflections. At each location, measurements were made for inter-node distances of 1m to 15m, taking 100 readings for each distance.

Outdoor tests illustrated in Figure 9(a), show the baseline measurement to be consistently within 1m, with an average of 0.6m. Figure 9 also shows indoor measurements for three separate locations. Indoor tests exhibit a much larger error in measured range. Moreover, the errors are not consistent across locations. While a large university hallway, shown in Figure 9(d), presented a largely linear relationship between distance and RToF measurements, more constrained cubicle floors and corridors, illustrated in Figures 9(b) and 9(c), show higher variations. The average error for our test setup was around 4.2m from the true value.



(a) Open Parking Lot (b) Metallic Cubicle Area (c) Lounge

(d) Hallway

Fig. 9. Evaluation of RToF range measurements at different location. (a) Shows outdoor measurements, which characterize error sources other than multipath. (b, c, and d) Show indoor locations. While the average error is high (4.2m), the measurements have a high correlation (94%) with distance.

Overall, our experiments indicate that RToF measurements cannot be utilized directly to obtain accurate indoor locations. These errors are caused due to the specific configurations of the environment and a general model cannot be assumed. However, a higher correlation exists with distance (when compared to other distance metrics like RSSI, hop-count, etc.), enabling the SensorFly nodes to use the measurements for a biased-random walk exploration algorithm. This also provides a better metric that RSSI and hop-count for use in-network localization and topology estimation protocols [11, 31], while meeting the platforms relatively strict cost, weight, and accuracy constraints.

5.2 Motion

We use PID controllers to implement the desired height and yaw control for SensorFly nodes. PID control is simple, does not require detailed dynamic models, and can be implemented using minimal computing power. However, hand tuning PID gains is tedious without simulation models. Such models exist for larger helicopter designs like quad rotors, but not for the SensorFly's essentially hard-to-instrument miniature design with low-cost components. Therefore, we derived an approximate first-principles dynamic model for SensorFly nodes.

Figure 10 shows the forces acting on the SensorFly during stable vertical flight. The force F is the net force on the helicopter, where f_0 is equal to weight, and f is the upward thrust due to the combined effect of the two rotating blades. These forces can be expressed as

$$F = f_0 + f, \tag{2}$$

$$f_0 = -mg, \tag{3}$$

where m is the mass of the helicopter and g is the acceleration due to gravity.

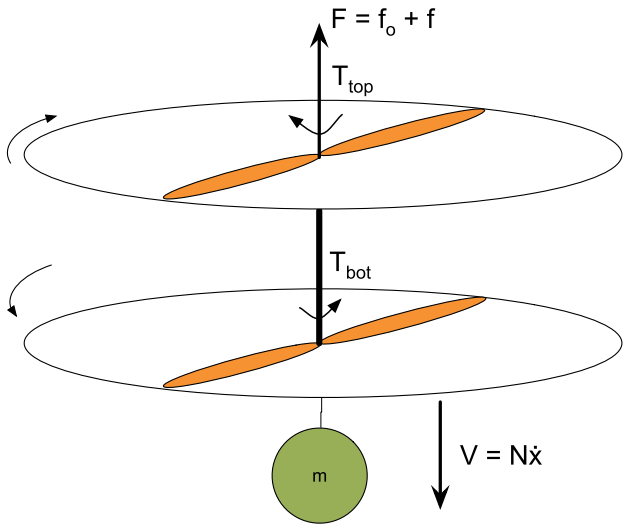


Fig. 10. This figure shows the forces acting on the SensorFly during stable vertical flight.

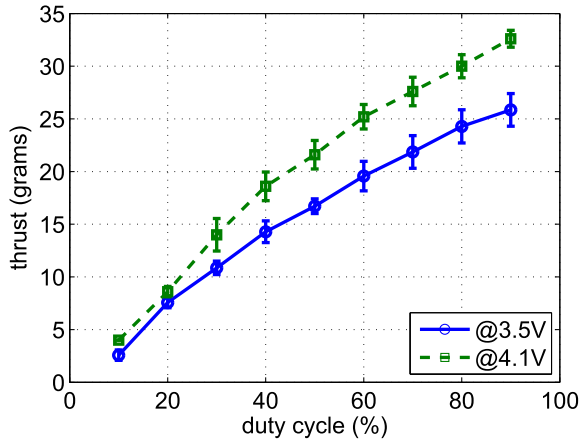


Fig. 11. In this figure, the empirically determined relationship between the upward thrust f generated by rotors and the input duty cycle is shown at different battery voltage levels.

The thrust f is a function of the speed of rotation of the helicopter blades and hence the duty cycle of the voltage input to the drive motors. However, due to non-linearity in the drive motors and varying battery capacity, we empirically determine the relation between upward thrust f and the input duty cycle, as shown in Figure 11.

The force V in Figure 10 is the viscous drag or air resistance, which opposes the relative motion of an object through air. Drag forces act in a direction opposite to the oncoming flow velocity and are proportional to the velocity of the craft. N, the coefficient of viscous drag, is the proportionality constant relating the two for a given medium and object surface, giving

$$V = N\dot{x}, \tag{4}$$

where x is the vertical displacement from the ground.

An unbalance of forces in the vertical direction results in a linear acceleration given by

$$m\ddot{x} = F - N\dot{x}. \tag{5}$$

In the Laplace domain,

$$ms^2X(s) = F(s) - sNX(s). \tag{6}$$

The PID controller for maintaining altitude is based on a first principles model of the SensorFly node given by

$$X(s) = \left(\frac{\frac{1}{m}}{s^2 + \frac{N}{m}s}\right)F(s), \tag{7}$$

where x is the altitude, F is the input, m is the mass of a SensorFly node, and N is the coefficient of viscous drag.

Blade rotation causes a torque to be applied to the helicopter body. In stable hovering condition, the top and bottom rotor torques, T_{top} and T_{bot}, should be balanced. The dynamic equation is given by

$$I \times \alpha = T_{top} - T_{bot}, \tag{8}$$

where α is the yaw rate and I is the moment of inertia of the node about the vertical axis through the center of gravity.

A second PID controller, which adjusts the individual rotor speeds while keeping the total thrust constant, enables the craft to hold a certain direction (trim) or turn at a desired rate.

T_{top} and T_{bot} depend on the rotational speed of the top and bottom rotors, respectively. However, due to the greater lift contributions of the bottom rotor, to change the individual torques while keeping the lift constant, the speed of the top rotor and bottom rotor is changed by a ratio of 0.7 in opposite directions. For example, if the speed of the top rotor is increased by 5%, the speed of the bottom rotor is decreased by $0.7 \times$ 5%. The same mechanism is used to turn the node in an open loop mode, by creating a net torque in the desired direction. The ratio 0.7 is empirically obtained for V3 nodes and is conditioned on the shape of the circuit board.

From Equations (7) and (8), we design two independent PID control loops for height control and yaw control. The feedback for the altitude PID loop is a sonic distance sensor, measuring the height x of the node from the ground, while a two-axis gyroscope measures w for the yaw control loop. We take a heuristic approach to obtain an approximation of the system dynamics for simulation. The simulation enables us to obtain an initial ballpark value for the controllers proportional, derivative and integral gains. We further tune the control-loop gains obtained with real experiments. In our current basic configuration, a proportional gain of $K_p = 0.2$, a derivative gain of $K_d = 0.2$, and an integral gain of $K_i = 2 \times 10^{-4}$ is used as default parameters for altitude control loop. The loop frequency is 1KHz. For yaw control, $K_p = 0.6$, $K_i = 2 \times 10^{-3}$, and $K_d = 0$. The control module, running at a frequency of 1KHz, consumes only 2% time of the microcontroller.

5.3 Flight Time

Table 5 shows the performance of the flight controller and sensor control software in achieving stable hover at a given height. The flight time is lower for hovering at higher target altitudes due to the higher overshoots and settling time. While the current system provides sufficient stability at very low computational cost, better control strategies remain the focus of our work in the future. The flight time of approximately 5min is attainable with the prototype. Optimization of the mechanical design can extend flight time to 15min as has been obtained by similar, although RF-controlled, flying crafts [12].

Table 5. Performance of Flight Controller and Sensor Controller Software Modules

Set Height	Maximum Overshoot	Settling Time (70%)	Avg. Steady State Height	Flight Time
1ft	4ft	25s	1.5ft	6:20min
2ft	6ft	40s	2.5ft	5:30min
3ft	6ft	50s	3.5ft	4:50min

6 EVALUATION

In this section, we evaluate the performance of our platform in a realistic fire scenario. CFAST [32] is a computer simulation environment that fire investigators, safety officials, engineers, architects, and builders use to simulate the impact of past or potential fires and smoke in a specific building environment. It is a two-zone fire model used to calculate the evolving distribution of smoke, fire gases, and temperature throughout compartments of a building during a fire. Considering a multi-compartment building scenario, we extend the realistic fire growth model to include mobile SensorFly nodes with parameters and capabilities derived from actual experimental characterization presented in Section 5.

We present a physical simulation platform for drone algorithm design. The operating environment of a SensorFly-like cyber-physical system has many disparate computational and physical components that are not adequately modeled by existing robotics simulation packages. We present a simulation environment that combines a physical disaster model (indoor fire growth model), a radio path loss model, a wireless network model, and a node mobility model to comprehensively evaluate such cyber-physical systems.

6.1 Methodology

We compare the performance of the autonomously deployed mobile SensorFly nodes with a statically pre-deployed network of sensors in a three-room building shown in Figure 12. The CFAST fire simulation runs for a total of $t = 1800$s, providing fire evolution data such as layer interface height, temperature, pressure, and gaseous composition along the height of each room. Fires are set in the Kitchen and Living room compartments at $t = 0$s. Each room has typical furniture with their combustible properties as provided in the simulation environment. The CFAST zone model assumes the conditions at a certain height of the room to be uniform. Therefore, only variations along the vertical dimension in the building compartments are of interest in sensor placement.

The static network is pre-deployed consisting of three nodes placed at heights of 0.5m, 1.2m, and 2.4m in each of the three rooms. The nodes measure temperature at 10s intervals and route the data back to the base node at the entrance of the building structure. The SensorFly nodes are introduced into the environment at $t = 0$s into the simulation arena. The nodes deploy autonomously and route back sensed temperature readings to the base station at 10s intervals, moving vertically to obtain data at different heights. Figure 12 also shows the placement of sensors in the simulation arena as well as the entry point for SensorFly nodes. The mobility models, network protocol, and radio link characteristics used for the simulation are described in the following section.

6.2 Evaluation Metrics

Both approaches provide discrete sensor readings in time and height. Sensor readings from both approaches are interpolated to give a continuous surface along the time and spatial dimensions. With the simulation data as ground truth, we define two metrics as a measure of the ability of the approach to provide accurate fire evolution predictions:

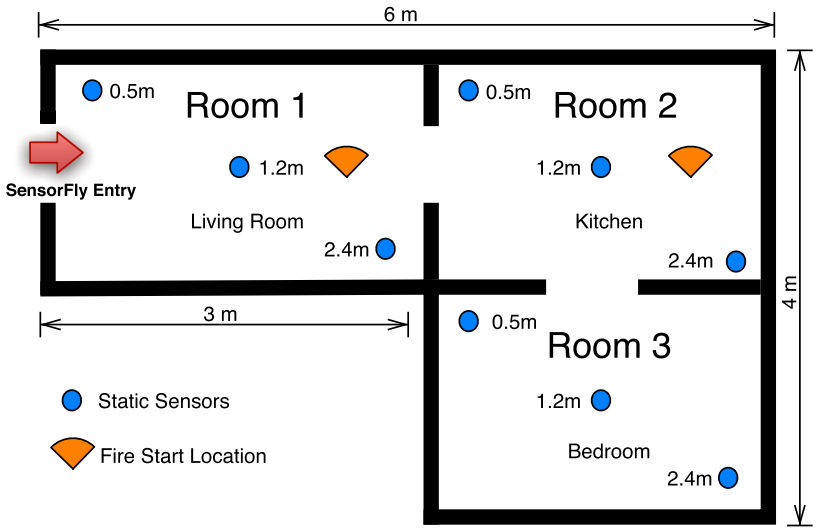


Fig. 12. Arena for SensorFly and CFAST fire simulations. The figure shows the building geometry, the placement of nodes for the static network, the entry point for mobile SensorFly nodes, and the point where fires are initiated at $t = 0$s.

Average Model Error. This is defined as the root-mean-square error of the predicted model obtained by interpolating the discrete sensor readings reported by the static nodes and mobile SensorFly nodes. The positions of the static nodes are predefined and the real positions of the flying nodes are tracking by the simulation platform. It is noticed that the simulation platform maintains two kinds of positions for flying nodes: the real positions of the flying nodes as ground-truth positions and the estimated location derived from localization module in Figure 13. The flying nodes interpolate the sensor reading from estimated positions to get a continuous surface. The readings are interpolated in both the height and time dimensions to the maximum resolution of the CFAST simulator, 0.1m in height and 10s in time. For static nodes, the error comes from (1) sensor reading and (2) interpolations. For the flying nodes, besides the previous two errors, the estimated position brings another error. This metric captures the performance of the system in terms of predicting an accurate model from sensed data.

Spatio-Temporal Percentage Coverage. The static nodes have limited resolution in the space dimension, since only a few nodes can be economically installed as part of a universal infrastructure. Conversely, the mobile SensorFly nodes by virtue of being introduced into the environment and relying on autonomous means to deploy are constrained in the temporal dimension. That is, the mobile nodes may not be available at the desired location. The spatio-temporal percentage coverage captures the effect of both these characteristics. It is defined as the ratio of the number of sensor readings in the height-time plane to the maximum possible resolution obtainable. To make this metric calculable, we split the height-time plane to 1s by 1 cubic meter. The maximum possible resolution obtainable is decided by the desired average model error. If we would like to achieve higher resolution, then there would be higher average model error. In the implementation, we need to find a trade-off between resolution and accuracy.

First, we compute these metrics from deployments without accounting for the effect of network disruptions. Second, to evaluate the adaptability of the solution, we introduce network disruptions by randomly failing a subset of nodes. The failure of nodes causes loss of sensed data from the

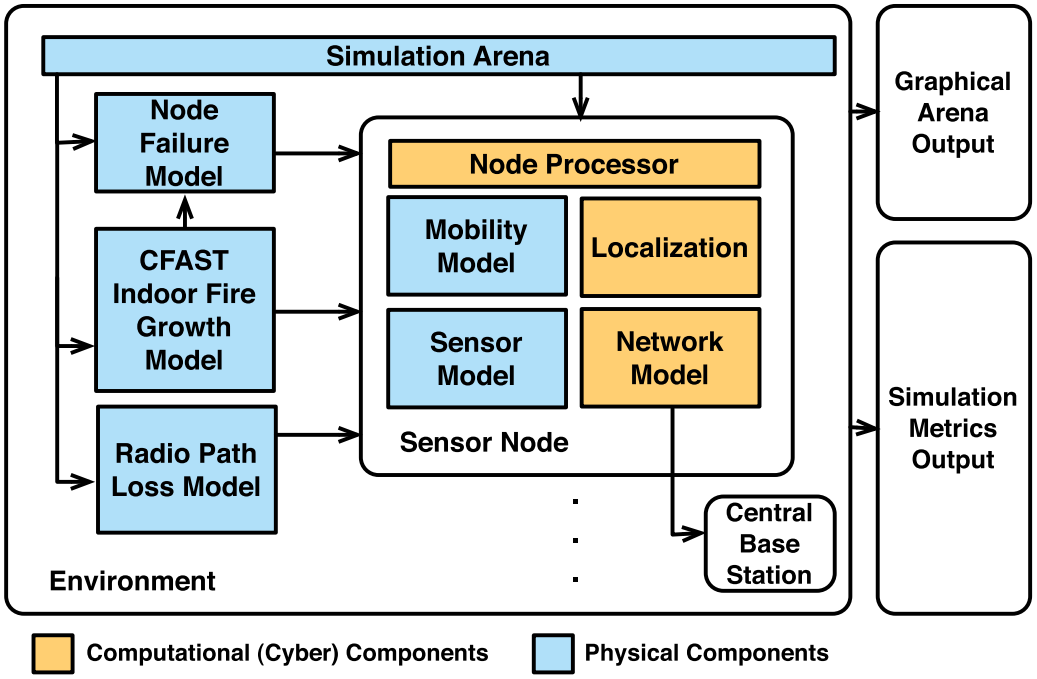


Fig. 13. Simulation framework.

nodes themselves, as well as from the nodes, which are partitioned from the base station. We also consider the effect of the number of deployed SensorFly nodes on sensing effectiveness.

6.3 Simulation Framework

To make our simulation result close to the real scenario, we incorporate the real and experimentally evaluated capabilities and characteristics of the SensorFly platform to the simulation framework. We describe the various components of the framework (shown in Figure 13) in the following paragraphs.

Two key parts make the simulation framework close to a real scenario: the environment modules and the sensor node module. The environment modules describe the physical aspects of the surroundings while the sensor node modules describes capabilities and characteristics of the SensorFly platform from experiments.

In environment modules, the fire growth model provides a realistic prediction of temperature and smoke in the specified simulation arena. The radio path loss model enables wireless link characterization for communication between nodes. The failure model incorporates the failure rate of nodes corresponding to environmental conditions. The simulation arena provides the configuration parameters for the system, including the building geometry and material information. The environment modules act as inputs to the mobile sensor node. The sensor node module consists of modeled components of the hardware such as sensors, radio, and processor. In addition, it incorporates some software modules such as localization algorithms and network protocols. Finally, the simulator outputs a real-time graphical representation of the simulation arena and movement of sensor nodes. Each module is described in more detail in the following sections.

The sensor node module consists of modeled components of the hardware, such as sensors, radio, and processor. The capabilities and characteristics of the SensorFly platform are obtained from experiments. In addition, it incorporates some software modules, such as localization algorithms and network protocols.

Finally, the simulator outputs a real-time graphical representation of the simulation arena and movement of sensor nodes. Each module is described in more detail in the following sections.

6.3.1 Indoor Fire Model. Sensors for real-time sensing and prediction of fire propagation is an active area of research. Researchers have proposed fire models to predict the advance of fire from in-situ sensor readings. Such a model is required to provide physical sensing layer for the simulation environment.

Our current implantation includes a popular model called CFAST [32]. CFAST is a zone model where each space is split into two zones with uniform conditions in each zone. The top zone consists of the high temperature gases and smoke, while the lower layer consists of lower temperature gases. The height of the interface between the two zones is called the smoke layer height and this layer descends as smoke builds up in the room. The layer height is used as an indication of the extent of the fire.

The CFAST model is described as an initial value problem for a system of ordinary differential equations. These equations arise from first-principle laws of conservation of mass, the conservation of energy, the ideal gas law and relations for density and internal energy. The model estimates the pressure, layer height, and temperatures given the accumulation of mass and enthalpy in the two layers as a function of time. The model takes as input the simulation arena, which is described in Section 6.3.2.

The outputs of CFAST are variables that are needed for estimating the conditions in a building subject to a fire. These include temperatures of the upper and lower gas layers within each compartment, the surface temperatures within each compartment, and the visible smoke and gas species concentrations within each layer.

6.3.2 Simulation Arena. The simulation arena is the stage for the simulation. This is where the fire originates and propagates as well as the mobile nodes sense temperature and smoke information. The framework requires creating a simulation arena that supplies the configuration parameters required by the CFAST fire growth model, sensor node mobility model, and the radio path loss model. The arena parameters include:

- Information about the building geometry such as compartment sizes, materials of construction, and material properties.
- Connections between compartments such as horizontal flow openings such as doors, windows, vertical flow openings in floors and ceilings, and mechanical ventilation connections.
- Fire properties including fire size and species production rates as a function of time.
- Common combustible material such as furniture, defined by their thermal conductivity, specific heat, density, thickness, and burning behavior.
- Placement of initial sensor node positions (entry points).

The parameters are defined through a configuration file that is read by simulation framework, which is implemented in MATLAB. CFAST compatible input files are generated by the MATLAB program.

6.3.3 Mobility Model. The mobility model specifies the algorithm for motion path planning and obstacle detection. The model can be configured to correspond to various platforms. The framework currently models the SensorFly system. The SensorFly platform has the ability to measure

height through the ultrasonic range finder, measure its pose through an integrated three-axis compass, and measure the distance from other nodes through the radio's time-of-flight capability. Assuming these capabilities, a motion model is implemented for the simulated SensorFly node using empirically collected data.

- When the SensorFly nodes are connected to the base station directly or through other nodes, they follow a biased random walk for exploring and deploying in the on-fire building. The nodes move at designated speed in random directions until they are within a minimum specified distance from only one other SensorFly node, to maintain connectivity as well as spread out through the building for exploration.
- The SensorFly nodes may move so far that no other node is within a maximum specified distance or when no data route exists to the base station. Since the nodes are lost and do not know where to go, the best way is to move in random directions until they find another SensorFly node that can connect them to the base station. This behavior is to guard against partitioning of the network.
- When either of the two motions aforementioned, nodes also move vertically at a specified speed in a periodic fashion to obtain readings at different heights. The vertical position corresponding to a given reading is obtained from the height sensor.

This motion model is realistic and easy to implement scheme for our system and requires few physical and sensing assumptions for accurate simulation.

6.3.4 Sensors. The framework allows for sensors that can sense the variables output by the fire model, as well as obstacles in the simulation arena. The virtual SensorFly node is equipped with sensors corresponding to the actual hardware, including an ultrasound height sensor, an electronic compass, a gyroscope, and an accelerometer. It includes an error model for sensors that is currently assumed to have a normal distribution based on empirical observations on our hardware.

6.3.5 Network Model. The framework implements an aggregate and broadcast communication model for communication. The monitoring nodes seek to route all sensed data to the base station. Accounting for constant motion of nodes, establishing routes, and running explicit node discovery service is considered impractical and avoided. Nodes are simulated to periodically broadcast messages containing their sensor data.

Neighboring nodes, on hearing the broadcast message, aggregate the node's sensor data with their message. Each node's sensor data consists of a sequence identifier and a time-to-live field. The time-to-live is decremented with the number of hops as well as on the expiration of a local time window, to control the time for which stale data propagates in the network. A node's data is propagated by other nodes only if the time-to-live is still not zero. Old sensor data from a node is replaced with fresh data, if it is received before the expiration of the time-to-live field. This scheme is akin to a controlled reverse-flood of data to the base station.

6.3.6 Radio Path Loss Model. A radio path loss model is required to provide a more realistic characterization of the wireless links for node-to-node communication. As complex models would be too computationally expensive and unfeasible, we use shadowing with a path loss exponent of 3 as the radio link model for simulations. This is similar to that employed by previous indoor fire monitoring work [41] and is an estimate for a single-floor multi-room scenario [36].

6.3.7 Node Failure Model. The framework provides a configurable stochastic model for node failures. It allows a failure rate to be specified corresponding to maximum threshold for environmental variables such as temperature and gas concentrations, as predicted by CFAST. This allows testing of redundancy schemes and sensing performance for a realistic deployment.

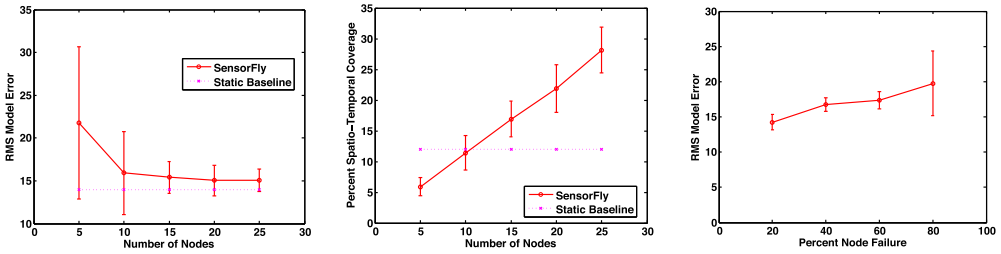


Fig. 14. (a) Average error in predicted model shown as a function of the size of SensorFly deployment. (b) The percentage coverage in time and height dimensions is shown as a function of deployment size. (c) Shows the error obtained in 25 node SensorFly deployments, when a percentage of nodes fail at t = 900s.

6.3.8 Localization. We do not require any accurate localization for navigation or mobility as nodes follow random paths through the building. This is because (1) the goal of the system is to achieve high spatial-temporal percentage coverage for sensing the building and (2) the SensorFly nodes are resource-constrained without vision and laser sensors. Therefore, the system obtains coarse-grained locations for tagging sensor readings with radio-based method every second. The framework models the capability of the SensorFly nodes to measure inter-node distances through time-of-flight ranging. All the inter-node distances are broadcasted to the base station for multi-lateration localization. Knowing estimates of inter-node distances, an iterative multi-lateration algorithm is simulated to obtain compartment-level location as in Reference [37].

6.4 Results

The simulation starts at $t = 0$ with a fully deployed static network with nodes in all three rooms, and all SensorFly nodes introduced into room 1. Figure 14(a) shows the average model error as the number of nodes introduced into the arena is increased. The error from the static deployment of nine nodes, three in each room, is shown for comparison. The error is large for a small number of nodes primarily because of the lower coverage. The error and the variance in error decreases as the number of nodes is increased. The error stabilizes at ten nodes, about the same as the size of the static node deployment. The difference in error between the autonomous SensorFly deployment and the base line static one is about 14% with similar-sized deployments. It decreases further as the number of nodes increases. The stabilization can be attributed to the fact that once nodes are present in all three rooms, additional nodes increase redundancy but improve the error only slightly given the uniform model. A scenario with higher resolution sensing needs will benefit more with a larger number of nodes.

Figure 14(b) shows the percentage coverage of SensorFly deployments of various sizes. The percentage coverage, as defined earlier, is the total points, in the height and time dimension, available from sensing to the maximum points provided by the CFAST simulator. In a real-world scenario, the resolution would be infinite. However, this baseline corresponds to an ideal case sensing resolution suitable for the phenomena being sensed and its spatial distribution. The coverage increases with the number of nodes as expected, with SensorFly coverage being about equal to that of static nodes at the deployment size of ten nodes. The coverage increases almost linearly thereafter. This is because the mobile SensorFly nodes can provide very high resolution in the height dimension. However, when the number of nodes is fewer, the nodes are slower to enter into a compartment

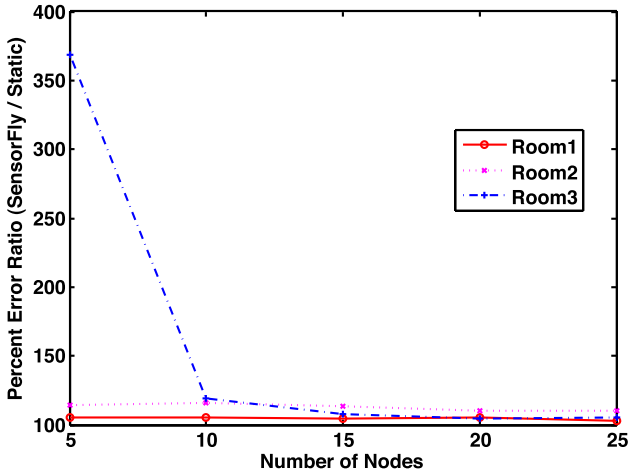


Fig. 15. The figure shows the ratio of SensorFly errors to static node error by compartment location. SensorFly nodes achieve similar performance closer to areas where they are introduced. For Room 3, which is farther away and obstructed, the error is higher for small sized deployments.

and therefore have less temporal coverage. Arguably, this is a result of the biased random exploration and deployment method employed.

Figure 14(c) shows the adaptability of the network to node failure and network disruptions. In a 25-node deployment, a percentage of nodes are failed at t = 900s. The actual nodes to fail are picked at random from the total nodes for each run. The failure of nodes causes loss of sensed data from the nodes themselves, as well as from the nodes that are partitioned from the base station. The mobile node deployment shows a sub-linear increase in error for node disruptions. Increasing the number of nodes failed from 20% to 80% causes a 5-degree increase in error, as nodes not within range of other functioning nodes resume random motion, until a connected network is re-established. This makes the network adaptive to node failure.

We evaluate the break up of errors in different building compartments to further illustrate the nature of mobile deployments. Figure 15 shows the ratio of the error of SensorFly deployments to the baseline static deployments on a room-by-room basis. The geometry of the deployment arena quite obviously has an effect on the accuracy of the predicted model in case of SensorFly. As the SensorFly nodes are introduced into Room 1 (Living Room), the error matches or is lesser than the static deployment case due to the higher resolution possible due to mobility. Room 3 is farthest from the point of entry and obstructed by two doorways. Therefore, smaller deployments of autonomously deployed SensorFly nodes have large errors as compared to static nodes in Room 3. However, as the number of nodes increase, the nodes spread out faster and achieve better coverage in time, eventually matching errors shown by static nodes.

Figure 16 shows the spatial coverage that is obtained by SensorFly nodes in the three-room simulation arena over a duration of 1,800s. The arena volume is divided into 1-cubic-meter volume regions. A region is covered if a SensorFly node visits it during the simulation run time. The coverage is computed as the percentage of regions covered by the node to the total number of regions in the simulation arena. We observe that for a given time a larger number of nodes, following our biased random-walk exploration scheme, can obtain higher coverage. However, diminishing returns are observed, for spatial coverage alone, once the number of nodes increases above 15 nodes. For the five-node scenario, the coverage remains flat with time due to the inability of nodes to explore

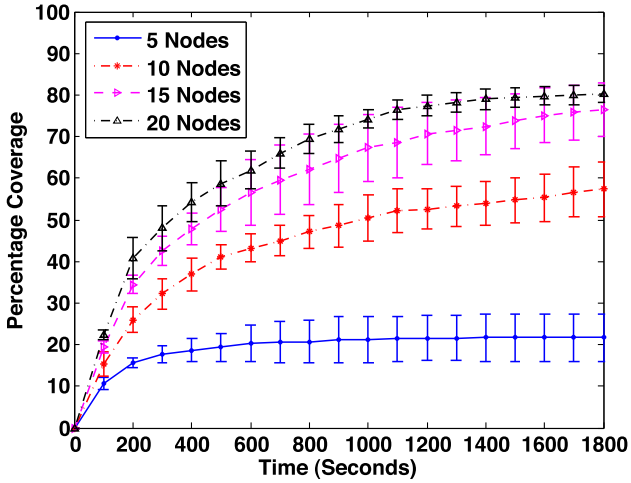


Fig. 16. The figure shows the percentage 3D coverage for SensorFly nodes in the three-room simulation arena over a duration of 1,800s. The error bars show the standard deviation of coverage obtained.

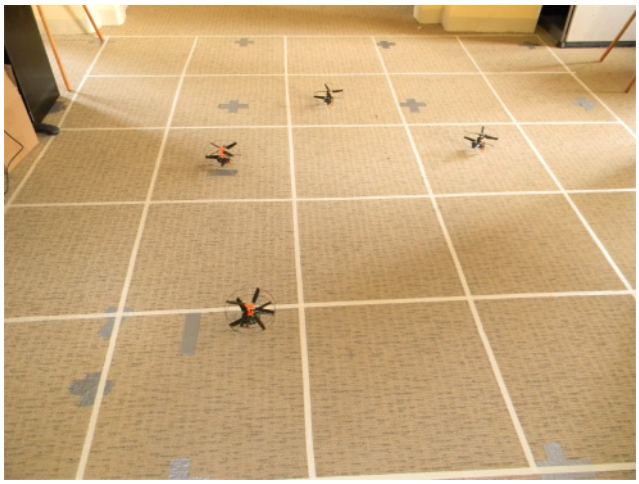


Fig. 17. Experimental setup for determining 2D space coverage in a 100-square-feet space with four SensorFly nodes.

further while maintaining a network route to the base station. SensorFly nodes achieve better performance closer to areas where they are introduced.

6.5 Experiment

We performed a small-scale controlled experiment with real SensorFly nodes to validate the coverage trends obtained from the simulation. SensorFly nodes programmed with the biased random-walk exploration scheme were introduced into an enclosed area of 100 square feet as shown in Figure 17. The area was divided into square regions of 2 square feet each. A region was designated as being covered if a SensorFly node visits it during a time of 2min. Coverage was determined visually as a percentage of visited regions to the total regions in the area. The number of nodes

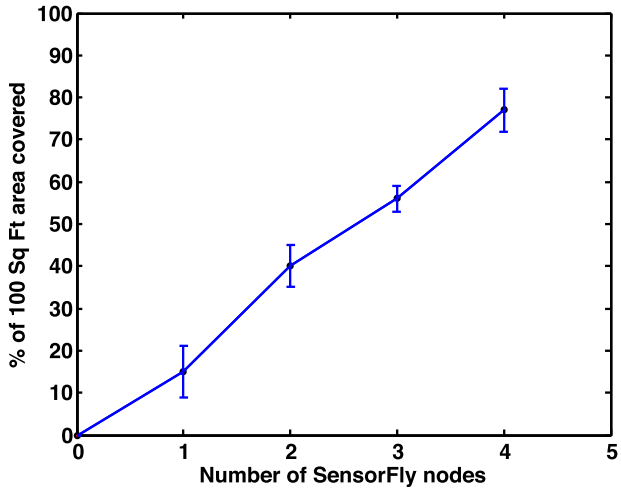


Fig. 18. The 2D space coverage in an area of 100 square feet over a duration of 2min using the biased random walk exploration scheme.

was varied from one to four. Each experiment was performed ten times and the average coverage with standard deviations is plotted in Figure 18. We observe a similar trend of increasing coverage with increasing size of deployment as seen in the simulation.

6.6 Physical Feature-Based Simulation for Version Comparison

To show the improvement of our system design on different SensorFly versions, we evaluate the performance of four versions of SensorFly platforms with physical feature-based simulation [34]. The simulation is conducted by 20 times with ten nodes to prevent bias.

Since the main differences of four SensorFly platforms lie on sensing noise and flight time, we adopt DrunkWalk [8] indoor localization and navigation algorithm to illustrate the design improvement. DrunkWalk is a collaborative and adaptive indoor localization and navigation algorithm for Micro-Aerial Vehicles (MAVs). In general, the localization accuracy is particularly affected by sensing noise while the navigation is affected by both sensing noise and flight time.

In the physical feature-based simulation, we collect the real sensing noise and flight time from experiments and evaluate the localization accuracy and navigation. All experiments were performed 25 times with ten SensorFly nodes. We run the simulation for a time period of 90s, 180s, 240s, and 300s, which correspond to the average flight time of SensorFly V1 to V4.

To illustrate the performance improvements caused by system design, specifically sensing noise, we plot the cumulative distribution function (CDF) of location estimation errors of SensorFly V1 to V4 in Figure 19.

More than 50% location estimation errors of SensorFly V4 are less than 1m, while the previous three versions have 24%, 16%, and 12%, respectively. This shows that SensorFly V4 enables higher resolution indoor localization with lower sensing noise, which is caused by system design improvement over previous versions (described in Sections 3–5).

In addition, 90% location estimations of SensorFly V4 have errors less than three meters. In contrast, the previous three versions have 56%, 38%, and 30%, respectively. This shows that low sensing noise from system design improvement of SensorFly V4 offers stronger ability to limit the location estimation error than previous three versions.

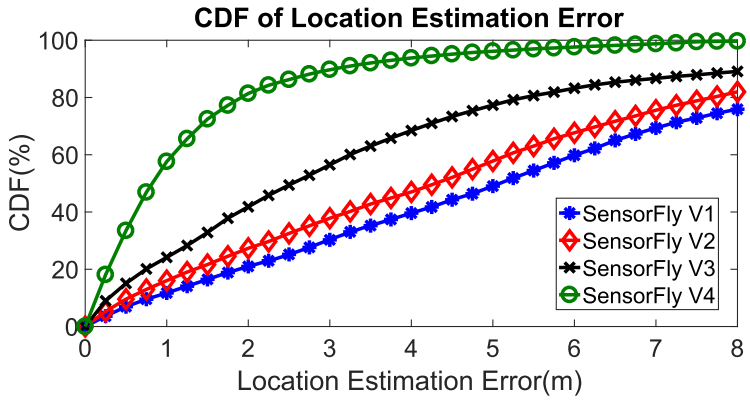


Fig. 19. The cumulative distribution function (CDF) of location estimation errors of four versions of SensorFly platforms.

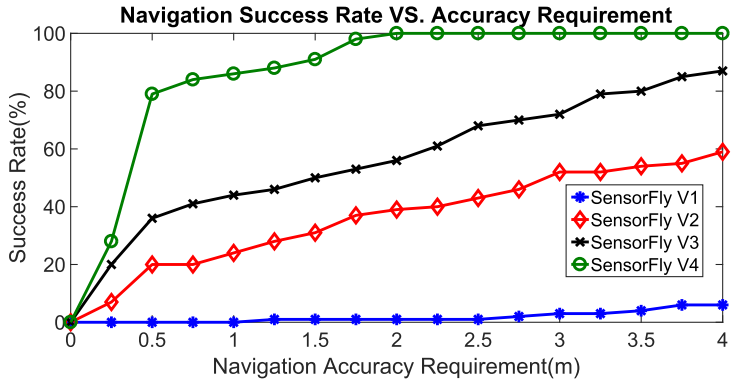


Fig. 20. The figure shows navigation success rate under different navigation accuracy constraints for four versions of SensorFly platforms. It is noticed that the flight time of four SensorFly versions varies from 90s to 300s.

To illustrate the performance improvements caused by system design (sensing noise + flight time), we compare the navigation success rate under different accuracy constraints. A successful navigation is achieved when the node can be navigated to the destination within the given navigation accuracy requirement before it runs out of battery. It is noticed that the flight time of four SensorFly versions are 90s, 180s, 240s, and 300s, respectively. For example, if the destination location is (4m, 5m) and the navigation accuracy requirement is 1m, a successful navigation for SensorFly V1 means that the node can arrive within the range of 1m from (4m, 5m) within its flight time (90s).

Figure 20 shows the navigation success rate as a function of navigation accuracy for four SensorFly versions. When the navigation accuracy is strict (0.5m), SensorFly V4 has ~80% success rate, while the other three versions have 36%, 20%, and 0%, respectively. The large improvement comes from (1) the lower sensing noise of V4 to ensure accurate localization and (2) the longer flight time to ensure the node have enough time to arrive within 0.5m range from the destination.

When the navigation accuracy requirement is less constrained (2m), SensorFly V4 achieves 100% success rate. In contrast, the other three previous versions have 56%, 37%, and 1% success

rate, respectively. This shows that the hardware improvements on both sensing noise and flight time of SensorFly V4 greatly improves successful MAV navigation.

7 RELATED WORK

Research in sensor networks has been actively conducted for the better part of ten years. By far, the most explored area has been fixed networks [24, 40, 42]. These sensing applications have some similar characteristics to SensorFly, in that they are mostly networks with multiple neighbors, and nodes are composed of microcontrollers, radios, and sensors. The main metrics of the system are energy usage, data flow, and data aggregation.

Mobility has been explored in sensor networks largely in context of sensor nodes being carried by human beings or animals [18]. However, unlike SensorFly, the work focuses mainly on adapting the system to user mobility.

Controlled-mobility has been envisioned by previous work in wireless sensor networks that proposes the idea of using controllably mobile elements in the network to alleviate resource limitations and improve system performance by adapting to deployment demands [19]. Somasundara et al. provide a theoretical analysis of the advantages of controlled-mobility in improving sensing fidelity and node lifetimes with prototype deployments using ground robot platforms [38]. The SensorFly, on the other hand, focuses on the hardware platform designed for enabling controlled-mobile indoor aerial sensing.

Mobility has often been explored as part of robotics research. A segment of research focuses on monolithic or a small team of robots, which may be remote-controlled or autonomous. Typically research on these devices focuses on individual stability and independent navigation, requiring sophisticated sensors [13]. Robotic platforms tend to have a higher per device cost and are economical for deployment in much smaller numbers compared to traditional sensor networks. Consequently, they are constrained in their ability to cover large areas simultaneously and rapidly.

The idea of miniature indoor flying platforms has been proposed before in literature. One work explores using a miniature electric helicopter to combine UAV flocking and wireless cluster computing [14]. Allred et al. present wireless link characterization for a network of semi-autonomous MAV's for atmospheric plume sensing [5]. SensorFly is the lightest functional system by at least a factor of 5 and targets a different design space. The SensorFly provides a platform for indoor sensing, accomplishing navigation and networking under strict resource constraints, with a high degree of collaboration.

Wood et al. have worked on flapping-wing micro-mechanical flying devices capable of autonomous flight [43], weighing under 200mg. This is the target hardware platform for the RoboBees [2] project. We believe these flying mechanisms represent exciting advances and underscore the need for research into controlled-mobile, highly resource constrained collaborative sensing and coverage algorithms. With a fully functional hardware platform, SensorFly allows us to validate assumptions and determine true tradeoff points in realistic deployments.

For fire monitoring, the FIRE [1] project proposes SmokeNet, a pre-deployed network of nodes with smoke and deferential temperature sensors. The system also consists of nodes with LED's to visually alert or guide firefighters. The SIREN [17] project focuses on improving the firefighter's access to information, using a WiFi-enabled PDA with peer-to-peer networking capabilities to communicate with an infrastructure of sensors. These sensors warn firefighters of hazards as well as help with navigation and localization. The FireGrid [41] project employs zone models and utilizes an array of static sensors positioned from ceiling to floor. Using the correlation between the sensors, the system proposes a communication protocol for emergency response that minimizes congestion. The assumption of a universal pre-established infrastructure, in all of the previously mentioned work, limits their adoption in the near-term.

Another proposed approach has been that of automatically deploying nodes as firefighters advance into the building on fire [22]. These sensor nodes are primarily concerned with relaying firefighter data back to the incident commander. Since this approach involves firefighters entering the building for sensor deployment to take place, such a system would have limited utility for providing situational *a priori* information without risking the rescuers' lives.

8 DISCUSSION

Having presented a description and evaluation of our controlled-mobile aerial platform, we note that several aspects warrant further discussion and could result in possible extensions to this work.

Interoperability with Static Networks. Pre-deployed static sensor networks have certain advantages, such as knowledge of exact locations and ability to provide data from regions that may be occluded due to closed doorways or structural collapse. On the other hand, the point-of-emergency deployment capability and mobility of SensorFly nodes allow larger deployments and higher resolution sensing of spaces where they can be introduced.

A hybrid approach with SensorFly nodes working in collaboration with static sensing infrastructure, where it exists, can combine the advantages of both approaches. Thus, research into making controlled-mobile and static networks interoperable, as well as work on leveraging the static network to augment the mobile-network's localization protocols, could be beneficial.

Communication Protocols. Networks of controlled-mobile nodes present new opportunities and challenges for design of communication protocols. In this article, we present a simple aggregate and broadcast scheme that is sufficient for relaying the low-bandwidth temperature data. Nevertheless, we envision more complex communication scenarios with heterogeneous sensor nodes with varying bandwidth requirements. The network must provide quality-of-service for both higher-bandwidth data, such as camera streams as well as low-bandwidth temperature readings. Another challenge is the co-existence of delay-tolerant communication with communication that has timeliness requirements, such as that required for localization of nodes. Furthermore, the constant but controlled mobility of nodes provides opportunities in designing routing and discovery schemes better suited to such networks.

Radio Propagation. Radio propagation may be affected adversely in emergency response environments, such as in the presence of smoke and fire. This may impact the speed and extent of coverage obtained for a deployment of specific size. Recent research has studied the effect of fire on wireless propagation in wildfire environments [7]. The work suggests that ionization in flames causes particular frequency bands to be attenuated. An empirical study of the characteristics of fire propagation in indoor environments for our chirp spread spectrum radio would be helpful in obtaining a more accurate estimation of system performance in real deployments.

Energy and Flight Time. Miniature aerial platforms remain limited in their flight time due to high-power consumption of motors. Improvements in mechanical design and battery technology can extend flight times to the order of 15–20min, which however may not be sufficient for many applications. We seek to explore collaborative techniques to distribute movement tasks and manage energy across the group. For example, in the fire monitoring scenario examined, nodes can collaborate with nodes in their close proximity and *duty cycle* their flying task. One node can land and act as relay, while another flies and senses the temperature. When the battery level of the flying node becomes low, the roles can be reversed.

Environment Influence on the Performance. The SensorFly is designed to deploy autonomously for different challenging operations such as fire rescue. Environmental factors such as temperature and smoke will affect the performance of the system. The individual SensorFly nodes could fail due to these factors, but the overall swarm of SensorFly should function through redundancy. The wind affects the motion of SensorFly, which could be solved by PID control.

Temperature and smoke affect the performance on the following aspects. (1) The SensorFly body frame may be damaged under high temperature. This could be solved by adopting some fire resistant material. (2) The radio propagation may also be affected in real scenarios. The performance characteristic could be found in the datasheet of radio module [44]. (3) The IMU sensors will also be affected under different temperatures. A calibration algorithm based on temperature could be designed to deal with this problem. All the previous effects could be modelled as a node failure model. To make our physical simulate more realistic, we incorporate the failure model, which describes failure rate of nodes corresponding to environmental conditions, as shown in Figure 13. The failure model provides a configurable stochastic model for node failures. It allows a failure rate to be specified corresponding to maximum threshold for environmental variables, such as temperature and smoke, which is predicted by CFAST. This allows testing of redundancy schemes and sensing performance for a realistic deployment. In addition, the failure SensorFly nodes also indicate these factors during system operation.

9 CONCLUSION

In this article, we presented a novel 29g controlled-mobile aerial sensor network platform for indoor emergency fire monitoring applications. We identify the challenges in low-cost low-weight aerial sensing platform design and propose an architecture that utilizes limited-capability resource-constrained individual sensing nodes to autonomously and quickly achieve network-wide sensing objectives.

We evaluated the platform in the fire-monitoring scenario using realistic CFAST indoor fire simulation models. We show that autonomously deployed SensorFly nodes can achieve performance in both sensing quality and coverage that matches or exceeds pre-deployed static network infrastructures. The autonomy and adaptability of SensorFly-like networks can eliminate the cost of building large sensing infrastructures and reduce the risk to firefighters, as compared to prior static sensor network approaches.

REFERENCES

[1] FIRE project. Retrieved from http://fire.me.berkeley.edu/.

[2] RoboBees. Retrieved from http://robobees.seas.harvard.edu.

[3] USFA firefighter fatality reports and statistics. Retrieved from http://www.usfa.dhs.gov/fireservice/fatalities/statistics/index.shtm.

[4] Tarek Abdelzaher, Yaw Anokwa, Peter Boda, Jeff Burke, Deborah Estrin, Leonidas Guibas, Aman Kansal, Samuel Madden, and Jim Reich. 2007. Mobiscopes for human spaces. *IEEE Pervas. Comput.* 6, 2 (April 2007), 20–29. DOI: http://dx.doi.org/10.1109/MPRV.2007.38

[5] J. Allred, A. B. Hasan, S. Panichsakul, W. Pisano, P. Gray, J. Huang, R. Han, D. Lawrence, and K. Mohseni. 2007. An airborne wireless sensor network of micro-air vehicles. In *Proceedings of the 5th International Conference on Embedded Networked Sensor Systems*. ACM, 129.

[6] B. Bethke, M. Valenti, and J. P. How. 2008. Uav task assignment. *Robot. Automat. Mag. IEEE* 15, 1 (2008), 39–44.

[7] J. Boan and Jonathan Alexander. 2007. Radio experiments with fire. *IEEE Anten. Wireless Prop. Lett.* 6 (2007), 411–414. DOI: http://dx.doi.org/10.1109/LAWP.2007.902809

[8] Xinlei Chen, Aveek Purohit, Carlos Ruiz Dominguez, Stefano Carpin, and Pei Zhang. 2015. DrunkWalk: Collaborative and adaptive planning for navigation of micro-aerial sensor swarms. In *Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems*. ACM, 295–308.

[9] Digi International. 2009. XBee Multipoint RF Modules Datasheet. Retrieved from www.digi.com.

[10] CDR H.R Everett. 1989. Survey of collision avoidance and ranging sensors for mobile robots. *Robot. Auton. Syst.* 5, 1 (May 1989), 5–67. DOI: http://dx.doi.org/10.1016/0921-8890(89)90041-9

[11] Tian He, Chengdu Huang, Brian M. Blum, John A. Stankovic, and Tarek F. Abdelzaher. 2005. Range-free localization and its impacton large scale sensor networks. *ACM Transactions on Embedded Computing Systems (TECS)* 4, 4 (2005), 877–906.

[12] Hobbytron. Intelli mini RC helicopter. Retrieved from http://hobbytron.com/.

[13] G. Hoffmann, D. G. Rajnarayan, S. L. Waslander, D. Dostal, J. S. Jang, and C. J. Tomlin. 2004. The Stanford testbed of autonomous rotorcraft for multi agent control (STARMAC). In *Proceedings of the 23rd Digital Avionics Systems Conference (DASC'04)*, Vol. 2.

[14] O. Holland, J. Woods, R. De Nardi, and A. Clark. 2005. Beyond swarm intelligence: The ultraswarm. In *Proceedings of the Swarm Intelligence Symposium, 2005.* 217–224. DOI: http://dx.doi.org/10.1109/SIS.2005.1501625

[15] Honeywell. 2008. HMC6346 3-Axis Compass Datasheet. Retrieved from www.honeywell.com.

[16] Haomiao Huang, Gabriel M. Hoffmann, Steven L. Waslander, and Claire J. Tomlin. 2009. Aerodynamics and control of autonomous quadrotor helicopters in aggressive maneuvering. In *Proceedings of the 2009 IEEE International Conference on Robotics and Automation.*

[17] Xiaodong Jiang, Nicholas Y. Chen, Jason I. Hong, Kevin Wang, Leila Takayama, and James A. Landay. 2004. Siren: Context-aware computing for firefighting. In *Proceedings of 2nd International Conference on Pervasive Computing.* 87–105.

[18] Philo Juang, Hidekazu Oki, Yong Wang, Margaret Martonosi, Li S. Peh, and Daniel Rubenstein. 2002. Energy-efficient computing for wildlife tracking: Design tradeoffs and early experiences with zebraNet. In *Proceedings of the 10th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS'02)*, Vol. 37. ACM Press, New York, NY, 96–107. DOI: http://dx.doi.org/10.1145/605397.605408

[19] Aman Kansal, Mohammed Rahimi, W. J. Kaiser, Mani B. Srivastava, Gregory Pottie, and D. Estrin. 2004. Controlled mobility for sustainable wireless networks. *Proceedings of the IEEE Conference on Sensor and Ad Hoc Communications and Networks (SECON'04).*

[20] Markus Klann, Till Riedel, Hans Gellersen, Carl Fischer, Gerald Pirkl, Kai Kunze, Monty Beuster, Michael Beigl, Otto Visser, and Mirco Gerling. 2007. LifeNet: An ad-hoc sensor network and wearable system to provide firefighters with navigation support. In *Proceedings of the International Conference on Pervasive and Ubiquitous Computing (Ubicomp'07).*

[21] L. Klingbeil and T. Wark. 2008. A wireless sensor network for real-time indoor localisation and motion monitoring. In *Proceedings of the 7th International Conference on Information Processing in Sensor Networks.* IEEE Computer Society Washington, DC, 39–50. DOI: http://dx.doi.org/10.1109/IPSN.2008.15

[22] Hengchang Liu, Jingyuan Li, Zhiheng Xie, Shan Lin, Kamin Whitehouse, John A. Stankovic, and David Siu. 2010. Automatic and robust breadcrumb system deployment for indoor firefighter applications. In *Proceedings of the International Conference On Mobile Systems, Applications, and Services.* 21–34.

[23] Ting Liu, Christopher M. Sadler, Pei Zhang, and Margaret Martonosi. 2004. Implementing software on resource-constrained mobile sensors: Experiences with impala and zebraNet. In *Proceedings of the 2nd International Conference on Mobile Systems, Applications, and Services.* ACM, 256–269.

[24] Rahul Mangharam, Anthony Rowe, and Raj Rajkumar. 2007. FireFly: A cross-layer platform for real-time embedded wireless networks. *Real-Time Syst.* 37, 3 (Dec. 2007), 183–231. DOI: http://dx.doi.org/10.1007/s11241-007-9028-z

[25] MaxBotix. 2007. LV-MaxSonar-EZ1 High Performanxe Sonar Range Finder Datasheet. Retrieved from www.maxbotix.com.

[26] Ryan Morlok and Maria Gini. 2007. Dispersing robots in an unknown environment. *Distributed Autonomous Robotic Systems 6* (2007), 253–262.

[27] Petter Muren. 2008. Passively stable rotor system for indoor hovering UAS. *Proceedings of the International Powered Lift Conference.*

[28] Nanotron Tecnologies Gmbh. 2007. Real-time location systems white paper version 1.02 (May 2007), 1–20. Retrieved from http://www.nanotron.com.

[29] Nanotron Tecnologies Gmbh. 2008. nanoLOC AVR Module Technical Description Retrieved from http://www.nanotron.com/.

[30] Renzo De Nardi, Owen Holland, John Woods, Adrian Clark, and Wivenhoe Park. 2006. SwarMAV: A swarm of miniature aerial vehicles. *Proceedings of the 21st Bristol International UAV Systems Conference.*

[31] D. Niculescu and B. Nath. 2001. Ad-hoc positioning systems (APS). *Proceedings of the IEEE GlobeCom*, 2926–2931.

[32] Richard D. Peacock, Walter W. Jones, Paul A. Reneke, and Glenn P. Forney. 2005. CFAST, consolidated model of fire growth and smoke transport (version 6) user guide. *NIST Special Publication* 1041 (2005).

[33] Amruta Purohit, Zheng Sun, and Pei Zhang. 2013. Sugarmap: Location-less coverage for micro-aerial sensing swarms. In *Proceedings of the ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN'13).* IEEE, 253–264.

[34] Aveek Purohit and Pei Zhang. 2011. Controlled-mobile sensing simulator for indoor fire monitoring. In *Proceedings of the 7th International Wireless Communications and Mobile Computing Conference (IWCMC'11).* IEEE, 1124–1129.

[35] P. Zhang et al. 2005. Habitat monitoring with zebraNet: Design and experiences. *Wireless Sensor Networks: A Systems Perspective.* 235–257.

[36] D. Rutledge. *Investigation of Indoor Radio Channels from 2.4GHz to 24GHz.* IEEE. 134–137. Retrieved from DOI: http://dx.doi.org/10.1109/APS.2003.1219197.

[37] Andreas Savvides, C. C. Han, and M. B. Strivastava. 2001. Dynamic fine-grained localization in ad-hoc networks of sensors. In *Proceedings of the 7th Annual International Conference on Mobile Computing and Networking.* ACM, 166–179.

[38] Arun A. Somasundara, Aman Kansal, David D. Jea, Deborah Estrin, and Mani B. Srivastava. 2006. Controllably mobile infrastructure for low energy embedded networks. *IEEE Trans. Mobile Comput.* 5, 8 (August 2006), 958–973. DOI: http://dx.doi.org/10.1109/TMC.2006.109

[39] Syma. 2013. Syma RC helicopter. Retrieved from http://www.symatoys.com/.

[40] Robert Szewczyk, Alan Mainwaring, Joseph Polastre, John Anderson, and David Culler. 2004. An analysis of a large scale habitat monitoring application. In *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems (SenSys'04).* ACM Press, New York, NY, 214–226. DOI: http://dx.doi.org/10.1145/1031495.1031521

[41] Rochan Upadhyay. 2007. Towards a tailored sensor network for fire emergency monitoring in large buildings. *Proceedings of the 1st IEEE International Conference on Wireless Emergency and Rural Communications.*

[42] Geoff Werner-Allen, Konrad Lorincz, Jeff Johnson, Jonathan Lees, and Matt Welsh. 2006. Fidelity and yield in a volcano monitoring sensor network. In *Proceedings of the 7th Symposium on Operating Systems Design and Implementation (OSDI'06).* USENIX Association, Berkeley, CA, 381–396.

[43] Robert J. Wood. 2008. The first takeoff of a biologically inspired at-scale robotic insect. *IEEE Trans. Robot.* 24, 2 (2008), 341–347 .

[44] OEM Xbee. 2012. ZigBee/802.15. 4 OEM RF modules by maxstream. *Inc. Specifications* (2012).

[45] P. Zhang, J. Gu, E. E. Milios, and P. Huynh. *Navigation with IMU/GPS/Digital Compass with Unscented Kalman Filter.* IEEE. 1497–1502. DOI: http://dx.doi.org/10.1109/ICMA.2005.1626777

[46] Pei Zhang, Christopher M. Sadler, Stephen A. Lyon, and Margaret Martonosi. 2004. Hardware design experiences in zebraNet. In *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems.* ACM, 227–238.

Received February 2016; revised March 2017; accepted August 2017