# Indoor Navigation with a Swarm of Flying Robots

Timothy Stirling and James Roberts and Jean-Christophe Zufferey and Dario Floreano

*Abstract*— **Swarms of flying robots are promising in many applications due to rapid terrain coverage. However, there are numerous challenges in realising autonomous operation in unknown indoor environments. A new autonomous flight methodology is presented using relative positioning sensors in reference to nearby static robots. The entirely decentralised approach relies solely on local sensing without requiring absolute positioning, environment maps, powerful computation or long-range communication. The swarm deploys as a robotic network facilitating navigation and goal directed flight. Initial validation tests with quadrotors demonstrated autonomous flight within a confined indoor environment, indicating that they could traverse a large network of static robots across expansive environments.**

## I. Introduction

Swarms of flying robots are promising for applications such as search because they can rapidly travel above obstacles, have elevated sensing, and facilitate task parallelisation and redundancy [1], [2]. However, indoor flying robots have severely limited on-board sensing and processing, and GPS is unreliable due to attenuated signals. Therefore, *Swarm Intelligence* [3] is promising in simplifying control and reducing sensing and processing requirements. This work is based on quadrotors [4] which can take-off and land in small spaces, hover over targets and have high manoeuvrability. However, rotorcraft experience drift due to turbulence and imbalances [2], which is frequently controlled using absolute positioning from external tracking systems [5], unavailable in unknown environments. On-board sensing approaches include illumination-dependent cameras or laser scanners that can fail in homogeneous environments. These approaches are computationally expensive, often requiring off-board processing which delays control feedback due to transmission times and requires reliable long-range communication. Additionally, navigation usually requires absolute positioning or localisation using environment maps [6]. Maps may be unknown *a priori* and their online creation requires significant processing. Such approaches also do not scale to large environments [7] or swarms.

In summary, alternative strategies are required to enable indoor flying swarms without absolute positioning, long-range communication, centralised processing or environment maps—collectively referred to as *Global Information*. Previously we presented an aerial swarm search strategy using a *Robotic Sensor Network* paradigm within 3-D simulation [1]. A network of robots embedded in the environment with local sensing, processing and communication can solve the complex navigation task without global information. This paper extends our prior simulation work [1] by presenting a new methodology for autonomous flight across the span of the robotic network validated with real flight tests. On-board relative positioning sensors are utilised with an approach that is computationally simple and robust to varied illumination. The related work is discussed next and then the autonomous flight and navigation behaviours are detailed before presenting the results. Finally, limitations and future work are discussed.

## II. Related Work

Previous research in indoor flying swarms usually used external tracking systems in-place of GPS, e.g. [8]. However, on-board sensing is desirable for operation in unknown environments. One approach is to use laser scanners to estimate the robot pose and motion, e.g. [9] and [10] demonstrated indoor navigation of a quadrotor, but required significant off-board processing. However, such approaches can fail in large homogeneous environments such as long corridors, or near glass [11]. Additionally, suitable laser scanners only operate in 2-D, but the robots move in 3-D, leading to failure if there are large variations in vertical environment structure [11].

Alternatively, using illumination dependent cameras, indoor navigation using *a priori* image-databases was demonstrated in [12] and [13]. However, such approaches frequently require off-board processing, and depend upon suitable features, so artificial features are often pre-installed [14]. Although recent work towards using on-board processing is promising [14], [15], vision-based approaches have many undesirable properties. They can suffer undesirable control feedback caused by errors in pose estimation introducing oscillations and increasing platform motion. This increases image-blur, further degrading pose estimation and increasing platform motion, which escalates into an uncontrollable feedback loop [16]. Similarly, images from flying robots suffer from vibrations [12]. Similar problems affect optic-flow approaches [17], [18], which require significant illumination and contrast. Therefore, vision-based methods are not entirely satisfactory.

Using large blimps, [19] achieved basic swarm behaviours such as leader-following and aggregation using on-board infrared sensors. However, blimps are susceptible to disturbances and their large size ($\approx 1.0\,\mathrm{m}$) makes them unsuitable

We thank François Gervaix and Julien Brahier (HEIG-VD) for the tracking system and Maja Varga who assisted in the paper. This work is part of the “Swarmanoid Project”, a Future Emerging Technologies (FET IST-022888) project funded by the EC. All work was conducted at the Laboratory of Intelligent Systems (LIS), Ecole Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland.
timothy.stirling@gmail.com, james@iuavs.com, jean-christophe.zufferey@epfl.ch, dario.floreano@epfl.ch

for many applications and environments.

To summarise, previous approaches to indoor flight have many limitations, requiring pre-installed sensors or landmarks, appropriate illumination, or off-board processing, etc. Moreover, prior navigation methods also required global information such as maps. An alternative navigation method is to use robotic sensor networks, e.g., a pre-deployed sensor network guided outdoor flying robots in [20]. Instead of using pre-deployed networks, the robots themselves can be used as sensor nodes [21]. Various approaches to deploying robot sensor networks exist but none are suitable for indoor flying robots. The most common are based on attraction and repulsion forces between robots, termed Social Potential Fields (SPFs) [22]. However, complex tuning of the force laws is required and determining the parameters for a desired group behaviour is computationally infeasible [22]. Therefore, in previous work we presented a new approach suitable for swarms of flying robots [1], which was analysed within a 3-D dynamics simulator [23]. In this paper, the autonomous flight behaviours are developed and tested on flying robots, validating the proposed swarm navigation and search strategy.

## III. Methods

To achieve navigation without global information we use a *Robotic Sensor Network* paradigm [1]. Robots operate in one of two control states: *beacons* or *explorers*. Beacons are passively attached to ceilings maintaining an elevated view [24], forming nodes in the network. Explorers start clustered underneath a pre-defined *base* beacon. Flying explorers consecutively deploy and are guided by nearby beacons, flying from beacon to beacon across the network, shown in Fig. 1. Beacons sense their local environment and communicate with neighbours to derive the navigation signals. Beacons next to unexplored space indicate adjacent locations where a new beacon is required. Explorers that arrive at these locations become beacons, expanding the network. The network is dynamic, with beacons redeploying as explorers to search new areas. The beacons compensate for the robots' limited capabilities since explorers simply follow local guidance signals and concentrate their resources solely on autonomous flight. Beacons also have static, stable sensing and calculate obstacle-free trajectories without real-time constraints. A video demonstrating this in simulation is available online[1].

To achieve autonomous flight a novel methodology is proposed using a new 3-D relative positioning sensor [25]. The beacons act as static references allowing flying explorers to estimate their egomotion. By virtue of the beacon's elevated position, the sensor covers a wide field-of-view unobstructed by obstacles. The sensor operates by emitting a short pulse (1.5 ms) of infrared (IR) in an approximate sphere using powerful LEDs. Nearby robots receive the IR transmission through an array of photodiodes and triangulate the emitting robot using received signal strength, providing range, bearing and elevation estimates at 50 Hz. Low-bandwidth 2.4 GHz radio synchronises transmissions and sends small data-packets (10 bytes). The sensor also perceives its own IR reflection from the environment for proximity and altitude sensing.

[1]http://www.youtube.com/watch?v=5ywTFCYYnqs

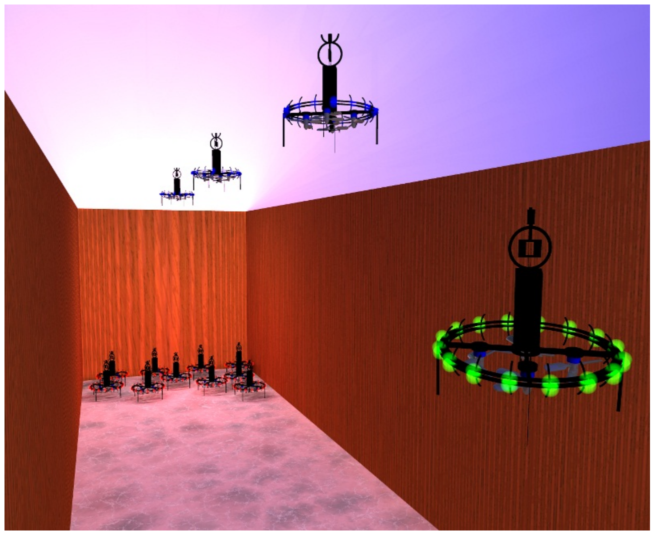


Fig. 1. Beacons provide simple navigation signals to flying explorers.

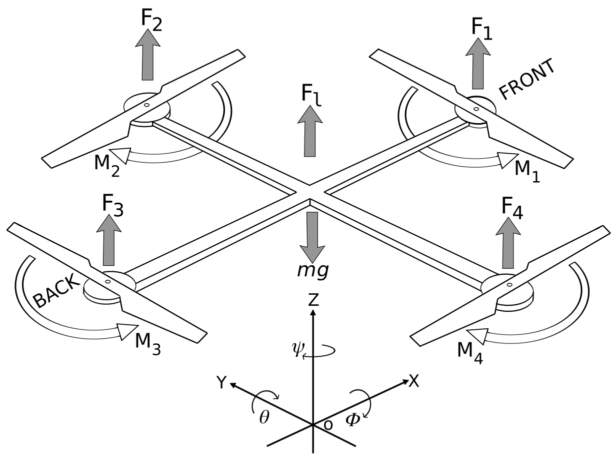


Fig. 2. Control forces ($F_i$) and torque moments ($M_i$) of a quadrotor.

### A. Goal-Directed Autonomous Flight

All processing is done on-board using micro-controllers. First, a navigation controller commands the robots's trajectory by calculating relative attitude angles to induce the desired motion. Secondly, a low-level controller stabilises the platform attitude and applies the attitude angles from the navigation controller [4]. Finally, motor controllers adjust the motor torques, inducing the rotor forces $F_1$–$F_4$ and torque moments $M_1$–$M_4$ to control the platform, see Fig. 2. Platform attitude is stabilised using inertial information from 3-axis accelerometers and gyroscopes by the stability controller at 500 Hz using a proportional-integral-derivative (PID) controller [26]. Altitude control is achieved with a PID controller using the altitude sensing of the relative positioning sensor. The robot bearing estimates are relative

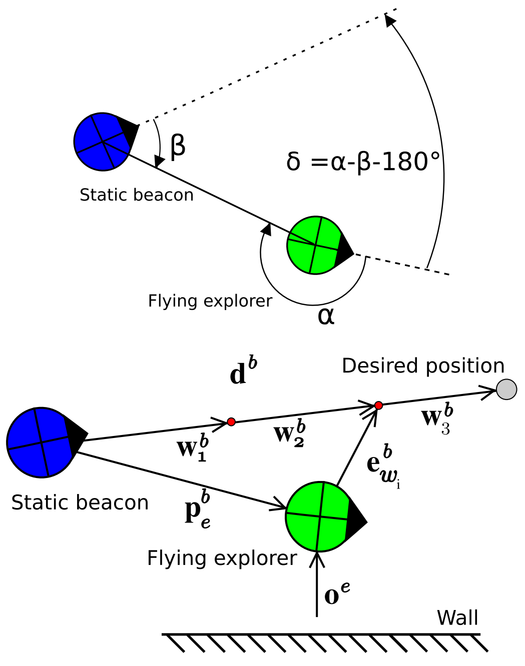


Fig. 3. Vectors used to achieve goal-directed autonomous flight.

to the robot headings, so two coordinate frames are defined for the explorer ($e$) and for the beacon ($b$) that the explorer uses for static referencing. The relative bearings between beacon and explorer, $\alpha, \beta$, are shown in Fig. 3 *top* (clockwise rotations are positive). The attitude angles are minor during low-speed indoor flight, so given $\delta = \alpha - \beta - 180°$, a rotation matrix between the coordinate frames is defined as

$$\mathbf{R}_b^e = \begin{bmatrix} cos(\delta) & -sin(\delta) & 0 \\ sin(\delta) & cos(\delta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad . \tag{1}$$

Robots use their relative positioning sensors to estimate the range *r* between beacon and explorer, and to exchange their bearings to calculate $\delta$. The position vector of the beacon in the explorer frame, $\mathbf{p}_b^e$, and the explorer in the beacon frame, $\mathbf{p}_e^b$, are given by

$$\mathbf{p}_b^e = (x_b^e, y_b^e, z_b^e)^T = (rcos(\alpha), -rsin(\alpha), z_b^e)^T \quad , \tag{2}$$
$$\mathbf{p}_e^b = (x_e^b, v_e^b, z_e^b)^T = (rcos(\beta), -rsin(\beta), z_e^b)^T \quad . \tag{3}$$

Autonomous flight is achieved with 4 different controllers: velocity control ($v$), yaw control ($\psi$), 3-D waypoint control ($w$) and obstacle avoidance ($oa$). The controllers are linearly combined giving the relative pitch ($\Delta\theta$), roll ($\Delta\phi$) and yaw ($\Delta\psi$) angles which are sent to the attitude stability controller:

$$\Delta\theta(n) = \Delta\theta_v(n) + \Delta\theta_w(n) + \Delta\theta_{oa}(n) \quad , \tag{4}$$
$$\Delta\phi(n) = \Delta\phi_v(n) + \Delta\phi_w(n) + \Delta\phi_{oa}(n) \quad , \tag{5}$$
$$\Delta\psi(n) = \Delta\psi(n) \quad . \tag{6}$$

Fig. 3 depicts the vectors between beacons, explorers and waypoints to the desired goal position. The navigation controller operates at 50 Hz, giving a control cycle time, $T_s$, of 20 ms. All gains and parameters were hand tuned.

*1) Velocity Controller:* The velocity controller is generally used to adjust the explorer velocity relative to the static beacon. However, in this work the velocity controller acted to dampen the platform dynamics and bring the platform to rest, unless other control signals are incorporated such as the waypoint controller. An estimated velocity vector $\bar{\mathbf{v}}^b(n)$ is computed by measuring the change in position $\mathbf{p}_e^b$ relative to the static reference beacon between control cycles. The estimated velocity vector in the $n$-th time step is given by

$$\bar{\mathbf{v}}^b(n) = \frac{1}{T_s}(\mathbf{p}_e^b(n) - \mathbf{p}_e^b(n-1)) \quad . \tag{7}$$

The control error is thus defined as

$$\mathbf{e}_v^b(n) = \mathbf{v}_{ref}^b(n) - \bar{\mathbf{v}}^b(n) \quad , \tag{8}$$

where $\mathbf{v}_{ref}^b(n)$ is the desired reference velocity. In this work, $\mathbf{v}_{ref}^b(n) = 0$ to dampen the platform dynamics. A first-order recursive low-pass filter is applied to the position estimate to increase robustness to sensor noise. A PID controller is used, given in discrete form for pitch by

$$\Delta\theta_v(n) = \Delta\theta_v(n-1) + K_p\Bigg[\left(1 + \frac{T_s}{T_i} + \frac{T_d}{T_s}\right)\mathbf{e}_v^b(n) - \left(1 + \frac{2T_d}{T_s}\right)\mathbf{e}_v^b(n-1) + \frac{T_d}{T_s}\mathbf{e}_v^b(n-2)\Bigg] \cdot \hat{\mathbf{x}}^b \quad , \tag{9}$$

where $K_p$, $T_i$ and $T_d$ are the P, I and D gains (all empirically tuned), and $\hat{\mathbf{x}}^b$ is the unit-vector $x$-dimension component in the beacon frame that assigns the relative portion of the control error to the desired pitch angle. The velocity controller for roll is similar in the $y$-dimension with $\hat{\mathbf{y}}^b$.

*2) Yaw Controller:* The yaw controller stabilises the heading and maintains a common heading with the reference beacon. The yaw control error is: $\mathbf{e}_\psi^b(n) = \delta = \alpha - \beta - 180°$. The PD yaw controller, with $K_{p\psi}$ and $K_{d\psi}$ as the respective P and D control gains, is given by:

$$\Delta\psi(n) = K_{p\psi}\left(\mathbf{e}_\psi^b(n)\right) + K_{d\psi}\left(\mathbf{e}_\psi^b(n) - \mathbf{e}_\psi^b(n-1)\right) \tag{10}$$

*3) 3-D Waypoint Controller:* The 3-D waypoint controller is used to navigate the explorer. The navigation signal of the beacon (see Sct. III-B), is interpreted into a vector $\mathbf{d}^b$ pointing towards the desired position (see Fig. 3 *bottom*). To facilitate goal-directed flight along $\mathbf{d}^b$, a set of *k* waypoints are created: $W = \{\mathbf{w}_1^b, \mathbf{w}_2^b, ..., \mathbf{w}_k^b\}$, $\sum_{i=1}^k \mathbf{w}_i^b = \mathbf{d}^b$. The waypoint controller commands the robot to fly towards the next waypoint. Once it arrives at a waypoint within a specified error margin, $\epsilon$, ($\epsilon = 20$ cm here) it briefly loiters for 1–2 s to allow the system to stabilise before selecting the next waypoint. The waypoints can be easily calculated by a linear subdivision, but were manually calculated here. The control error between the current waypoint $\mathbf{w}_i^b$ and the explorer position $\mathbf{p}_e^b$ is defined as $\mathbf{e}_{w_i}^b(n) = \mathbf{w}_i^b - \mathbf{p}_e^b(n)$. The waypoint P controller, with $K_{\theta wp}$ as the P gain, is given by:

$$\Delta\theta_w(n) = K_{\theta wp}\left(\mathbf{e}_{w_i}^b(n)\right) \cdot \hat{\mathbf{x}}^b \,. \tag{11}$$

When $\mathbf{e}_{w_i}^b$ is smaller than a threshold radius $R_t$ ($R_t = 30$ cm here), the waypoint $\mathbf{w}_i^b$ is filtered into the estimated robot position $\mathbf{p}_e^b(n)$ using a recursive filter, which improves controller response and stability as a form of gain-scheduling. The simple P controller is sufficient due to the stabilising influences of both the velocity controller and the attitude stability controller.

Explorers should select the closest beacon for referencing since it is expected to have the best signal quality. However, the utilised relative positioning sensor suffers from unreliable measurements when one robot is directly below another [25]. Therefore, explorers select the closest beacon in the opposite direction of the flight. When an explorer passes in-front of another beacon, the navigation controller updates a new vector $\mathbf{d}^b$ and waypoint set $W$. Waypoints are specified in 3-D and can also instruct the explorer to attach to the ceiling, achieved by gradually increasing the altitude causing it to perch with its passive attachment mechanism [24].

*4) Obstacle Avoidance Controller:* Beacons aim to signal collision-free trajectories, but the obstacle avoidance controller acts as a reactive safety system in-case of unobserved obstacles. This is achieved using the proximity sensing of the relative positioning sensor with a potential field behaviour that generates a vector $\mathbf{o}^e$ pointing away from the closest detected obstacles (see Fig. 3). For a sensor with $N$ uniformly spaced measurements (here $N = 8$), $\mathbf{o}^e$ is given by

$$\mathbf{o}^e = -\sum_{i=1}^{N} \left(1 - \left(\frac{d_i}{MAX_r}\right)\right) \cdot \begin{bmatrix} cos(\frac{i2\pi}{N}) \\ -sin(\frac{i2\pi}{N}) \\ 0 \end{bmatrix} \quad , \quad (12)$$

where $d_i$ is the distance to the $i^{\text{th}}$ measurement and $MAX_r$ is a normalisation constant. $\mathbf{o}^e$ is in the explorer frame and is transformed into the beacon frame using the rotation matrix (1): $\mathbf{o}^b = \mathbf{R}_b^e \mathbf{o}^e$. The PD controller for pitch is,

$$\Delta\theta_{oa}(n) = \left(K_{p_{oa}}(\mathbf{o}^b(n)) + K_{d_{oa}}(\mathbf{o}^b(n) - \mathbf{o}^b(n-1))\right) \cdot \hat{\mathbf{x}}^b \quad (13)$$

## B. Swarm Navigation Behaviour

To enable navigation through unknown environments the swarm behaves as a network of beacons [1]. It is desirable that the network has a regular topology with beacons approximately equally spaced to maximise their coverage (see Fig. 4 *top*). Beacons use 2 navigation behaviours to deploy the network: 1) local navigation to guide explorers to nearby empty locations requiring a beacon; 2) long-range navigation to guide explorers across the network to these locations.

Beacons derive local navigation signals by processing their surrounding environment using a simple mapping function shown in Fig. 4 *bottom*. First, range measurements from proximity or distance sensors are labeled as either *empty* or *obstacle* with respect to a threshold. Subsequently, the relative positioning sensor is used to determine nearby beacon locations. A 90° sector in the direction of each observed beacon is labeled *beacon*, unless that direction is already labeled *obstacle*. The mid-angle of the first sufficiently large *empty* sector is selected as the desired direction for the swarm to expand into. The expansion directions are selected systematically in clockwise order. In large open areas, detected using the distance sensors, the desired direction is confined to be in the cardinal directions to aid forming a square lattice.

Long-range navigation is afforded by the hop-counts of local communication signals propagated across the network [1]. This finds the shortest path across the network and is equivalent to Dijkstra's algorithm [27]. The hop-counts create a gradient that can be followed forwards or backwards. A "forwards" and "reverse" gradient are used with different properties. The forwards gradient is created by messages emitted from the base beacon starting from 1 and is incremented and propagated outwards to the network edge (see Fig. 4 *top*). This provides a direction leading away from the base if followed in increasing order, or towards the base in decreasing order. The reverse gradient emanates from beacons on the network edge with surrounding empty space and increment from 1 backwards across the network (see Fig. 4 *top*). This provides the shortest path to empty space in unexplored areas if followed in decreasing order, but not to the base in increasing order. All hop-counts are continuously updated as the network expands. If there are no beacons next to empty space, the reverse hop-counts continuously increment until they surpass a threshold, $RH_{MAX}$, and their propagation terminated.

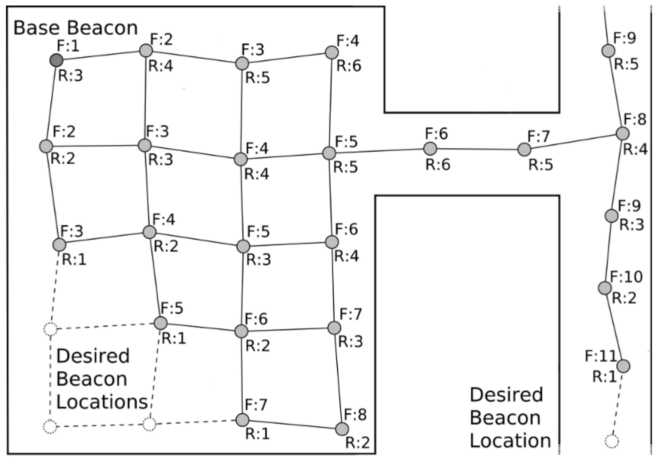
To expand the network, explorers are sent to the network edge using the reverse hop-count gradient. Beacons first label their surrounding environment as described above. If there is no empty sector then the direction to the beacon with the lowest reverse hop-count is selected as the desired direction. When an area is covered by beacons with no empty space, the reverse hop-count surpasses $RH_{MAX}$ and the beacons redeploy recursively to new unexplored areas.

Further details of the swarm algorithm are in our previously published simulation work [1]. Improvements shown here include the ability to perceive non-orthogonal environment geometry by way of an increased angular resolution distance scanner. Furthermore, within simulation all robots maintained a constant heading, but in this work explorers can operate with an arbitrary heading, however they will always try to align their heading with the nearest beacon using the yaw controller (see Sct. III-A.2).

## IV. Experimental Set-up

This work is based on the quadrotor robots (see Fig. 5 *bottom*) developed by James Roberts at the Laboratory of Intelligent Systems [4]. The robots can attach to ferromagnetic ceilings to maintain elevated sensing for prolonged periods [24]. Alternative methods using dry-adhesives or mechanical gripping are viable [28], or the robots could land on the ground. This work uses a 3-D relative positioning sensor specially designed for flying robots [25], [29]. The proximity sensing of the relative positioning system was used to sense the local environment. The hop-count gradients were implemented using messages sent through the relative positioning sensor.

Autonomous flight and navigation was tested using three robots in a $12\,\text{m}\times3.4\,\text{m}\times2.5\,\text{m}$ room. Two robots acted as beacons and were pre-positioned within the room. The first



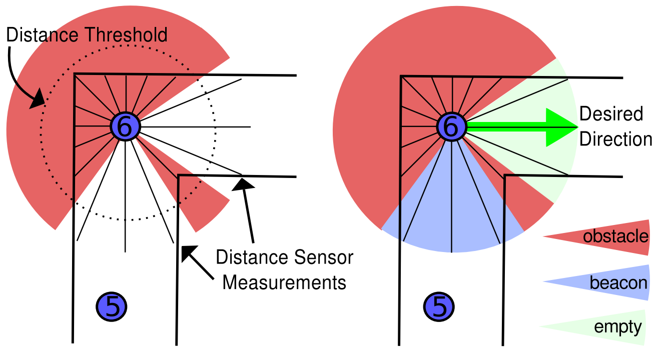


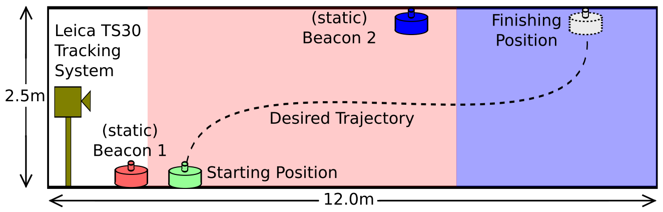
Fig. 4. *Top*) The deployed robotic network showing forwards (F) and reverse (R) hop-count gradients. *Bottom*) Beacons label their surrounding environment and selected a desired navigation direction.

was placed on the floor near one end of the room and was the base beacon, the second on the ceiling in the centre of the room. The third was used as an explorer, placed on the ground in front of the first robot. It was commanded to take-off, fly underneath the second robot and attach to the ceiling on the far side, shown in Fig. 5. Therefore, as the flying robot passes beneath the beacon on the ceiling, it must switch the static referencing from the beacon on the floor to the beacon on the ceiling. Flight trajectories over 15 trials were measured using a Leica TS30 tracking system at $\approx 5$ Hz with an accuracy of $< 1.0$ cm. This system uses a laser reflected off a glass prism placed near the top of the robot, 40 cm above the robot base.

## V. Results

Autonomous flight was successfully demonstrated for all 15 trials totalling 120 m of flight, shown in Fig. 6. The positions of the two beacons are shown as the green circles and the vertical dashed line represents the approximate distance where explorers switched their navigation control between beacons. Fig. 6 *top* shows a top-down view of the room length versus width. The $+$ symbols represent the ceiling attachment positions. Fig. 6 *bottom* shows the same trajectories from a side-view. A video demonstration is available online[2].

[2]`http://www.youtube.com/watch?v=qKJymdql05Q`



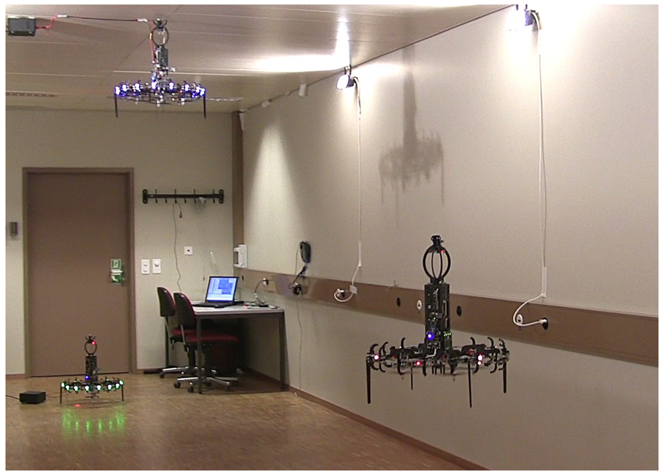


Fig. 5. *Top*) Experimental set-up. *Bottom*) Flight tests.

The trajectories were analysed between 1 m and 6 m to avoid the take-off and ceiling attachment portions. The mean height was 1.04 m and the mean standard deviation (s.d.) was ±0.16 m. The ideal flight trajectory is a straight-line along room length (the $x$-axis in Fig. 6 *top*). To understand deviations from the ideal trajectory, mean deviations in the perpendicular direction ($y$) were analysed. The mean $y$ position was $-0.03$ m and the mean s.d. was just ±0.11 m. The maximum distance was $-0.40$ m. On average, the flight over these 5 m took 7.4 s (±1.17 s) and the mean velocity was 0.69 $\mathrm{ms}^{-1}$ (±0.12 $\mathrm{ms}^{-1}$). The maximum velocity was 0.92 $\mathrm{ms}^{-1}$ and the minimum was 0.57 $\mathrm{ms}^{-1}$. The mean total flight time from take-off to ceiling attachment was 12.8 s (±1.05 s), the slowest was 14.1 s and the fastest 11.1 s. Analysing the ceiling attachment precision, the s.d. was ±0.15 m and the maximum distance from the desired point was 0.52 m. The mean travel distance was 7.97 m.

Importantly, switching the static referencing and navigation between the two beacons created only minor disturbances, similar to where the waypoints change at $\approx 1.5$ m (see Fig. 6 *top*). This was verified by comparing the perturbations at the first waypoint (1.0–2.0 m along the $x$-axis) to the second waypoint where the flying robot changes its static referencing (5.0–6.0 m) using the s.d. along the $y$-axis as a measure of perturbation. A Wilcoxon ranksum test indicated that there was no significant difference in perturbations between the two waypoints (ranksum = 209, n1=n2=15, $p > 0.34$). Therefore, although there is some slight visible disturbance, this is no stronger than when simply changing waypoints. The obstacle avoidance controller contributed to the trajectories persisting within a narrow space along the centre of the corridor.

In summary, the results show reliable goal-directed flight. There is no significant effect of the transfer of the static referencing between the two beacons, indicating that this is reliable. Therefore, in the future it would be possible to extend these results using a long chains of beacons.

## VI. Discussion

During the validation tests the explorer used a beacon on the floor and ceiling for static referencing, so the approach is generalisable to robots that cannot perch. This also demonstrates the 3-D capabilities, allowing operation in environments where approaches that assume 2-D flight might fail [11]. Although the relative positioning sensor is robust to varied illumination (up to $10,000\,\text{lux}$, [25]), the sensor is affected by reflections that can lead to interference when close to reflective surfaces, which is exploited for the proximity sensing. It may be possible to mitigate the interference by analysing the sensor's self-reflection. Furthermore, any interference did not significantly affect the flight results.

The obstacle avoidance behaviour is a reactive safety system in case of unobserved obstacles, with the explorers tasked with calculating obstacle-free trajectories using their stable sensing without real-time constraints. Therefore, we currently do not have result with obstacles blocking the desired trajectory. The potential field behaviour is not guaranteed to avoid collisions, and results are dependent on the platform dynamics. This is compounded by the short sensing range which limits the available time to react safely at high speeds. However, during empirical testing collisions were avoided in a majority of cases. If the desired trajectory is not updated then the explorer will continuously attempt to pass through the obstacle resulting in potentially destabilising oscillations. Future work should aim to resolve this, perhaps the explorer could become a beacon so it can better perceive the local environment.

Long-range navigation is achieved using two hop-count gradients. Additional gradients can be created when a beacon observes an event to which other agents must respond to. Beacons processed their surrounding environment to derive navigation signals using the proximity sensing in only the cardinal directions. To properly perceive irregular environments, a higher angular resolution sensor is needed such as a lightweight distance scanner. Such a sensor requires a much lower angular resolution, update rate and accuracy compared to laser scanners, and so can be much simpler and lighter.

Finally, in this work only one robot flew due to limited availability of robots. Future work involves scaling the results to larger environments using more robots.

## VII. Conclusions

This paper introduced a novel goal-directed autonomous flight methodology using relative positioning sensors. A corresponding swarm navigation algorithm was introduced that has been shown in simulation to successfully search large indoor environments. This integrated strategy operates entirely on-board without requiring external sensing, artificial landmarks or controlled illumination. The computational requirements are compatible with on-board microcontrollers. The algorithm is entirely decentralised and has low computational complexity that is constant with respect to the environment and swarm size—facilitating scalability to large swarms. Validation tests demonstrated navigation in a narrow room across the complete flight envelope including attaching to a ceiling. The flying robot switched its static referencing between two beacons without significant perturbations, so in the future it could follow a much longer chain of beacons across large environments and extend the beacon network. This work presents a promising integrated strategy for autonomous flight and navigation with swarms of indoor flying robots in applications such as search, environmental monitoring and disaster mitigation.

## References

[1] T. Stirling, S. Wischmann, and D. Floreano, "Energy-efficient indoor search by swarms of simulated flying robots without global information," *Swarm Intelligence*, vol. 4, no. 2, pp. 117–143, 2010.

[2] J.-C. Zufferey, S. Hauert, T. Stirling, S. Leven, J. Roberts, and D. Floreano, *Handbook of Collective Robotics*. Singapore: Pan-Stanford Publishing, 2011, ch. Collective Aerial Systems.

[3] E. Bonabeau, M. Dorigo, and G. Theraulaz, *Swarm Intelligence: From Natural to Artificial Systems*. Oxford University Press, 1999.

[4] J. Roberts, T. Stirling, J.-C. Zufferey, and D. Floreano, "Quadrotor using minimal sensing for autonomous indoor flight," in *Proceedings of the 2007 European Micro Air Vehicle Conference and Flight Competition (EMAV '07)*, Sep.17–21 2007.

[5] S. Lupashin, A. Schllig, M. Sherback, and R. D'Andrea, "A simple learning strategy for high-speed quadrocopter multi-flips," in *Proceedings of the International Conference on Robotics and Automation*. Piscataway: IEEE Press, May 3–7, 2010, pp. 642–648.

[6] S. Thrun, W. Burgard, and D. Fox, *Probabilistic robotics*. Cambridge, MA: MIT Press, 2005.

[7] A. Angeli, D. Filliat, S. Doncieux, and J.-A. Meyer, "2D simultaneous localization and mapping for micro aerial vehicles," in *Proceedings of the European Micro Aerial Vehicles conference (EMAV '06)*, 2006.

[8] M. Schwager, B. J. Julian, and D. Rus, "Optimal coverage for multiple hovering robots with downward facing cameras," in *Proceedings of the 2009 International Conference on Robotics and Automation (ICRA '09)*. Piscataway: IEEE Press, 2009, pp. 4016–4023.

[9] S. Grzonka, G. Grisetti, and W. Burgard, "Towards a navigation system for autonomous indoor flying," in *Proceedings of the International Conference on Robotics and Automation (ICRA '09)*. Piscataway: IEEE Press, 2009, pp. 2878–2883.

[10] A. Bachrach, R. He, and N. Roy, "Autonomous flight in unknown indoor environments," *International Journal of Micro Air Vehicles*, vol. 1, no. 4, pp. 217–228, 2009.

[11] M. Achtelik, A. Bachrach, R. He, S. Prentice, and N. Roy, "Stereo vision and laser odometry for autonomous helicopters in GPS-denied indoor environments," in *Proceedings of Unamanned Systems Technology XI (SPIE '09)*, vol. 7332. Orlando: The International Society for Optical Engineering, 2009, pp. 1901–1910.

[12] S. P. Soundararaj, A. K. Sujeeth, and A. Saxena, "Autonomous indoor helicopter flight using a single onboard camera," in *Proceedings of the 2009 International Conference on Intelligent robots and systems (IROS'09)*. Piscataway: IEEE Press, 2009, pp. 5307–5314.

[13] J. Courbon, Y. Mezouar, N. Guenard, and P. Martinet, "Visual navigation of a quadrotor aerial vehicle," in *Proceedings of the 2009 international conference on Intelligent robots and systems*. Piscataway: IEEE Press, 2009, pp. 5315–5320.

[14] G. Lee, F. Fraundorfer, and M. Pollefeys, "MAV visual SLAM with plane constraint," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA' 11)*, 2011.

[15] M. Achtelik, M. Achtelik, S. Weiss, and R. Siegwart, "Onboard IMU and monocular vision based control for MAVs in unknown in- and outdoor environments," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA' 11)*, 2011.

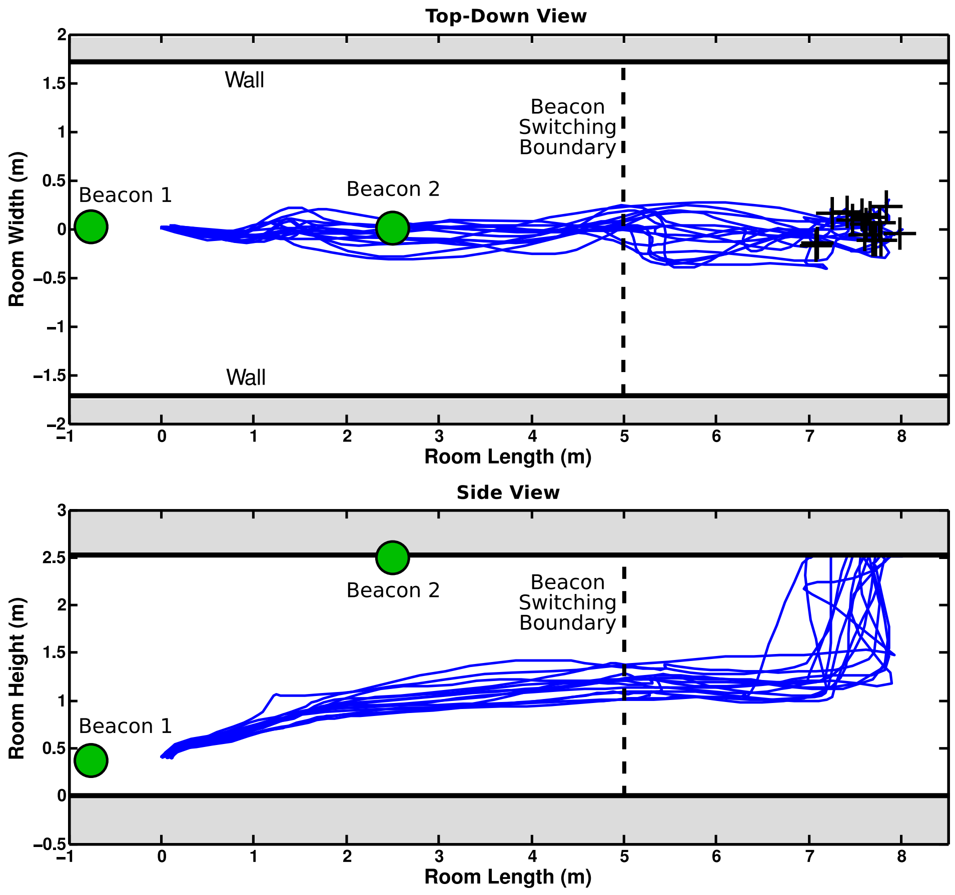


Fig. 6. Plots of the 15 flight trajectories. The 2 green circles represent the 2 beacons and the dashed line the approximate distance where explorers switched their static referencing. *Top*) Top-down view of the room, showing ceiling attachment positions (+). *Bottom*) Side view of the room.

[16] S. Ahrens, D. Levine, G. Andrews, and J. How, "Vision-based guidance and control of a hovering vehicle in unknown, GPS-denied environments," in *Proceedings of the International Conference on Robotics and Automation (ICRA '09)*. Piscataway: IEEE Press, 2009, pp. 2643–2648.

[17] J.-C. Zufferey, A. Beyeler, and D. Floreano, "Optic Flow to Steer and Avoid Collisions in 3D," in *Flying Insects and Robots*, D. Floreano, J.-C. Zufferey, M. V. Srinivasan, and C. Ellington, Eds. Berlin: Springer, 2009, pp. 29–50.

[18] W. E. Green and P. Oh, "Optic flow based collision avoidance on a hybrid MAV," *IEEE Robotics and Automation Magazine*, vol. 15, no. 1, pp. 96–103, 2008.

[19] C. Melhuish and J. Welsby, "Gradient ascent with a group of minimalist real robots: Implementing secondary swarming," in *Proceedings of the International Conference on Systems, Man and Cybernetics*, vol. 2. Piscataway: IEEE Press, 2002, pp. 509–514.

[20] P. Corke, R. Peterson, and D. Rus, "Localization and navigation assisted by networked cooperating sensors and robots," *The International Journal of Robotics Research*, vol. 24, no. 9, pp. 771–786, 2005.

[21] A. Howard, M. J. Mataric, and G. S. Sukhatme, "Mobile sensor network deployment using potential fields: A distributed, scalable solution to the area coverage problem," in *Proceedings of the 6th Distributed Autonomous Robotic Systems (DARS '02)*, vol. 5. Berlin: Springer, 2002, pp. 299–308.

[22] J. Reif and H. Wang, "Social potential fields: A distributed behavioral control for autonomous robots," *Robotics and Autonomous Systems*, vol. 27, no. 3, pp. 171–194, 1999.

[23] C. Pinciroli, V. Trianni, R. O'Grady, G. Pini, A. Brutschy, M. Brambilla, N. Mathews, E. Ferrante, G. D. Caro, F. Ducatelle, T. Stirling, A. Gutiérrez, L. Gambardella, and M. Dorigo, "ARGoS: a modular, multi-engine simulator for heterogeneous swarm robotics," in *Proceedings of the International Conference on Intelligent Robots and Systems (IROS '11)*. Los Alamitos: IEEE Press, 2011, pp. 5027–5034.

[24] J. Roberts, J.-C. Zufferey, and D. Floreano, "Energy management for indoor hovering robots," in *Proceedings of the International Conference on Intelligent Robots and Systems (IROS '08)*. Piscataway: IEEE Press, 2008, pp. 1242–1247.

[25] J. Roberts, T. Stirling, J.-C. Zufferey and D. Floreano, "3-D relative positioning sensor for indoor flying robots," *Autonomous Robots (forthcoming)*, 2012.

[26] D. Gurdan, J. Stumpf, M. Achtelik, K. M. Doth, G. Hirzinger, and D. Rus, "Energy-efficient autonomous four-rotor flying robot controlled at 1 kHz," in *Proceedings of the International Conference on Robotics and Automation*. Piscataway: IEEE Press, 2007, pp. 361–366.

[27] Q. Li, M. D. Rosa, and D. Rus, "Distributed algorithms for guiding navigation across a sensor network," in *Proceedings of the 9th annual international conference on Mobile computing and networking*. New York: ACM, 2003, pp. 313–325.

[28] M. Kovac, J. M. Germann, C. Harzeler, R. Siegwart, and D. Floreano, "A perching mechanism for micro aerial vehicles," *Journal of Micro-Nano Mechatronics*, vol. 5, no. 3–4, pp. 77–91, 2009.

[29] J. Roberts, T. Stirling, J.-C. Zufferey, and D. Floreano, "2.5D infrared range and bearing system for collective robotics," in *Proceedings of the International Conference on Intelligent Robots and Systems (IROS '09)*. Piscataway: IEEE Press, 2009, pp. 3659–3664.