MicNest: Long-Range Instant Acoustic Localization of Drones in Precise Landing

Weiguo Wang[1], Luca Mottola[2], Yuan He[1], Jinming Li[1],
Yimiao Sun[1], Shuai Li[1], Hua Jing[3], Yulei Wang[3]
[1]Tsinghua University, [2]Politecnico di Milano, Italy and RI.SE Sweden, [3]Meituan
wwg18@mails.tsinghua.edu.cn, luca.mottola@polimi.it, heyuan@mail.tsinghua.edu.cn
{li-jm19,sym21,lis20}@mails.tsinghua.edu.cn, {jinghua03,wangyulei02}@meituan.com

ABSTRACT

We present MicNest: an acoustic localization system enabling precise landing of aerial drones. Drone landing is a crucial step in a drone's operation, especially as high-bandwidth wireless networks, such as 5G, enable beyond-line-of-sight operation in a shared airspace and applications such as instant asset delivery with drones gain traction. In MicNest, multiple microphones are deployed on a landing platform in carefully devised configurations. The drone carries a speaker transmitting purposefully-designed acoustic pulses. The drone may be localized as long as the pulses are correctly detected. Doing so is challenging: *i)* because of limited transmission power, propagation attenuation, background noise, and propeller interference, the Signal-to-Noise Ratio (SNR) of received pulses is intrinsically low; *ii)* the pulses experience non-linear Doppler distortion due to the physical drone dynamics while airborne; *iii)* as location information is to be used during landing, the processing latency must be reduced to effectively feed the flight control loop. To tackle these issues, we design a novel pulse detector, Matched Filter Tree (MFT), whose idea is to convert pulse detection to a tree search problem. We further present three practical methods to accelerate tree search jointly. Our real-world experiments show that MicNest is able to localize a drone 120 m away with 0.53% relative localization error at 20 Hz location update frequency.

CCS CONCEPTS

• **Information systems** → **Location based services**; • **Computer systems organization** → *Embedded systems*;

KEYWORDS

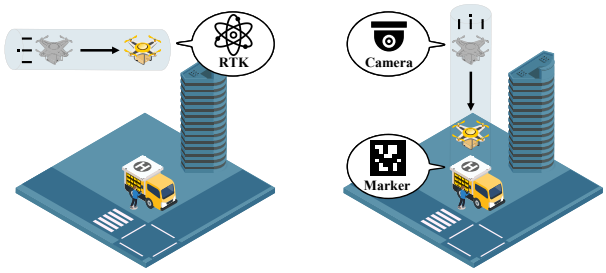
Drone, Acoustic Localization, Microphone Array

ACM Reference Format:
Weiguo Wang[1], Luca Mottola[2], Yuan He[1], Jinming Li[1], Yimiao Sun[1], Shuai Li[1], Hua Jing[3], Yulei Wang[3]. 2022. MicNest: Long-Range Instant Acoustic Localization of Drones in Precise Landing. In *The 20th ACM Conference on Embedded Networked Sensor Systems (SenSys '22), November 6–9, 2022, Boston, MA, USA.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3560905.3568515

*Yuan He is the corresponding author.

SenSys '22, November 6–9, 2022, Boston, MA, USA

ACM ISBN 978-1-4503-9886-2/22/11.
https://doi.org/10.1145/3560905.3568515



(a) Horizontal approach. **(b) Vertical descent.**

Figure 1: Two phases of drone landing.

1 INTRODUCTION

Aerial drone technology represents a new breed of computing platform [30], enabling applications in a range of fields such as agriculture, search and rescue, film-making, impromptu networking, and logistics. Landing is a key step in a drone's operation [16], one that is both delicate as the risk of damaging the drone itself or the surroundings is highest [52], and an essential component in many next-generation applications enabled by drone technology.

Target scenarios. With the roll-out of 5G networks, beyond-line-of-sight operation becomes possible: a drone flies autonomously over long distances without a physically co-located human pilot, but connected to an Internet back-end that monitors its operation in real time. This ability unlocks a range of potential applications, such as long-range visual inspections executed by multiple drones in a shared airspace [30] and instant asset delivery with drones.

Let us examine further the latter application. Compared to ground couriers, drones can bypass the complex urban traffic and deliver packages in a much shorter time. This is ideal for time-sensitive deliveries, such as medical supplies [79] and food [23, 27, 46], and also caters for contactless deliveries, which helps reduce the spread of viral diseases. Many companies are exploring the commercial feasibility of instant deliveries with drones; for example, Amazon [7], Alphabet Project Wing [6], Wal-Mart [26, 71], JD.com [38], Domino's [23], UPS [67], Ele.me [27], and Meituan [46].

Instant delivery with a drone unfolds as follows. After the package is placed onboard, the drone takes off, climbs to cruising altitude, and heads towards the destination. The latter is typically a self-collection station[1] near the customer. When the drone is close to the destination, a *precise landing* procedure is initiated, as pictorially depicted in Fig. 1. The drone first approaches the landing platform

[1]These self-collection stations are set up and operated by the drone delivery companies on their own, and are shared with the nearby residents.

horizontally to achieve vertical alignment, as in Fig. 1(a). Next, the drone starts the descent, shown in Fig. 1(b). Once the drone safely docks onto the landing platform, the package is dropped into the self-collection station, where the customer fetches it.

The consequences of not landing precisely can only be underestimated. Loss of the transported good is the most obvious. Should the drone miss the landing platform even partly, it would quickly lose control and crash, possibly damaging objects or hurting people [64]. There are the key requirements to avoid similar mishaps: *i) cm-level positioning accuracy ii) at 100+m altitude*, with positioning accuracy improving as the altitude decreases, and *iii) low-latency location updates* to feed the flight control loop. These requirements form the essence of our research problem.

State of the art. During the horizontal approach, shown in Fig. 1, the flight control loop mainly relies on Global Positioning System (GPS) or Real-Time Kinematic Positioning (RTK) [76]. During the descent, the flight control loop must keep the drone horizontally centered over the landing platform. As we further articulate in Sec. 2, settings exist, such as urban canyons, where the accuracy of GPS or RTK degrades with decreasing altitude up to making either system essentially unusable [18, 32, 34]. Auxiliary anchor-based systems may assist the drone during the descent such as visual markers [29, 40, 49, 70, 73], laser stations [13], ultra-wide band (UWB) stations [15, 22], or motion capture cameras [14].

Few of these systems, however, fulfill the requirements mentioned earlier. The only technique that may potentially do so is visual markers, read by downward-facing cameras the drone must be equipped with. According to our real-world experiences, we find that these techniques, however, exhibit key limitations:

1) They are *sensitive to lighting variations*; visual markers are difficult to detect in the fog or at night; equipping the drone with a light source may partly ameliorate the problem, however limiting the operational range to 20-30m and only obtaining unstable performance.
2) They are *constrained in horizontal coverage*; depending on the camera field of view and the drone's vertical alignment with the marker, the camera may not completely capture the marker, resulting in a localization failure [8].
3) They *limit system throughput*; as visual markers require line of sight between the camera and the marker, drones necessarily need to land one by one; otherwise, the first drone that commences the descent visually blocks the marker for all others.

MicNest. We present MicNest, an acoustic localization system to assist drones in precise landing, summarized in Sec. 3. A speaker carried by a drone broadcasts purposefully-designed acoustic pulses. Multiple microphones are deployed on the landing platform in carefully devised configurations. By localizing the speaker from the microphone signals, we localize the drone during the descent.

MicNest is rooted in the unique features of acoustic signals. The spatial resolution of a signal is proportional to its speed and inversely proportional to its bandwidth [59]. Thus, the slower a signal is, the finer spatial resolution it can provide. Acoustic signals with a limited bandwidth, say 24 kHz, can provide a fine-grained spatial resolution, around 0.71 cm when the sampling rate is 48 kHz. In comparison, the spatial resolutions of RF signals like UWB with a 1.3 GHz bandwidth or mmWave with a 4 GHz bandwidth are 10 cm [3] and 3.75 cm [37], respectively.

Compared to the visual techniques, using acoustic signals further allows MicNest to *i)* operate obliviously to lighting conditions and *ii)* provide much larger horizontal coverage. Further, we adopt Pseudo-Random Noise (PRN) to modulate acoustic pulses emitted by a drone. Because PRN pulses are orthogonal as long as they are statistically independent of each other, we can detect these pulses separately from the collided signal and identify which drone each pulse corresponds to. This makes MicNest able to *iii)* provide concurrent detection and localization of multiple drones.

MicNest provides, nonetheless, additional benefits. The use of PRN pulses makes MicNest friendly to human ear. As drones may operate in populated areas, pulses should not cause acoustic discomfort, yet PRN has the same acoustic characteristics as white noise and is almost imperceptible to human ears [48, 60, 72]. Finally, MicNest is resistant to impersonation attack, because pulses are (pseudo) randomly generated and it is difficult for third parties to generate the same pulse and impersonate a drone.

We want to point out that MicNest is a complement, not a replacement, to the existing localization solutions. The safety of commercial drones can not be overemphasized. To assist drones in precise landing, MicNest will not work alone but will cooperate with RTK and the visual marker to provide a more reliable and accurate localization service.

Challenges and contribution. Localizing drones via acoustics must tackle *three fundamental challenges*.

First, the SNR of acoustic pulses is inherently low. The transmission power of the speaker must be limited to avoid acoustic discomfort. MicNest needs to achieve long-range localization, thus acoustic pulses experience significant attenuation. Further, background noise in many cities is intrinsically strong, around 40-75 dB SPL [51], and when airborne, drone propellers generate much acoustic interference [10], possibly up to 104 dB SPL[57].

Second, acoustic signals experience non-linear signal distortions due to Doppler effects. The severity of this effect is inversely proportional to signal speed. Compared to RF signals, the sound speed is much lower. It can be expected that acoustic pulses experience serious distortion when drones are airborne. Modern flight control loops take flight decisions at 400+ Hz, rapidly changing the drone velocity. An acoustic pulse thus experiences various degrees of Doppler effect, ultimately undergoing non-linear distortions.

Third, signal processing must withstand the latency constraint imposed by the nature of flight control loops. The latter consumes location information as one of their most critical inputs. Evidence shows that increasing the latency of location updates may represent a source of system instability [52]. To make things worse, MicNest must provide low-latency location information at a time when the system dependability is most important: during landing.

The key enabling technology behind MicNest is a *novel pulse detector*: Matched Filter Tree (MFT), presented in Sec. 4. The key idea is to model pulse detection as a tree search problem. We cut one pulse into multiple short segments. The time span of each segment is short enough that the drone velocity in the three dimensions can be considered as constant. Therefore, Doppler distortion within each segment is linear. We then build a search tree, where

each level's nodes correspond to each segment's candidate drone velocities. For each segment, we check all possible velocities to compensate its distortion. If all segments are compensated with the right velocities, the problem of non-linear distortion is thus addressed. In addition, MFT allows us to increase pulse length on demand to further address the low-SNR problem. MFT addresses the first and second challenges above.

To enable low-latency localization, we present three techniques to accelerate tree search: *i)* tree pruning for reducing the search space by reducing the branching factor of the search tree; *ii)* correlation acceleration for reducing the time cost of each search by accelerating correlation; and *iii)* heuristic search for reducing the number of searches by exploiting past history to quickly dive towards the solution. The three techniques, presented in Sec. 5, are used jointly to address the third challenge above.

We report on the performance of MicNest in Sec. 6. Using a custom foldable landing platform we build, based on a total of 40 hours of real-world experiments with multiple drones, we provide evidence of how MicNest fulfills the requirements at stake. We demonstrate, for example, that MicNest provides a median error of just 0.043 m below 20 m altitude, that is, where accuracy matters the most for precise landing. In an experiment covering altitudes up to 120 m, nonetheless, the absolute localization error over the distance to the landing platform is only 0.53%. The mean latency to obtain a location update at the drone is 29.7 ms, which is compatible with use in flight control loops [17, 52]. We also show how MicNest is only marginally affected by factors it cannot control, such as background noise, and can localize multiple drones with limited degradation of accuracy.

2 RELATED WORK

Our work lies at the intersection of localization in mobile robotics and localization using acoustic signals.

Localization in mobile robotics. GPS is by far the most popular technique providing meter-level accuracy in outdoor settings. RTK is an improved but more expensive version of GPS and can achieve cm-level accuracy. Satellite-based systems become inaccurate or prevented from operating in urban canyons, because the surrounding buildings may reflect or block the GPS signals [18, 32, 34], resulting in the serious multipath issue or NLOS reception, respectively.

Because of these issues, several auxiliary techniques exist to localize mobile robots and especially aerial drones. AprilTags [40, 49, 73] may be used to localize drones by using a downward-facing camera to detect visual markers on the ground. Similar techniques also exist that are customized for testbed operation [4]. These systems can achieve cm-level accuracy. However, they are sensitive to lighting conditions and have a limited horizontal coverage depending on the camera field of view and distance to the marker.

Optics-based systems such as Lighthouse [13] and motion capture systems [14] can also localize drones with cm-level accuracy. However, their localization range is limited to a few meters, which makes them unsuited to enable precise landing. TrackIO [22] uses UWB tags for localizing drones. The median error is 1+ m, which may not be sufficient to support precise landing. Finally, similar to MicNest, Rabbit [45] deploys a speaker on the drone to emit frequency-modulated continuous wave (FMCW) signals and uses a mobile phone to track the drone. Although the localization error is less than 3 cm, the range is limited to 1.5 m.

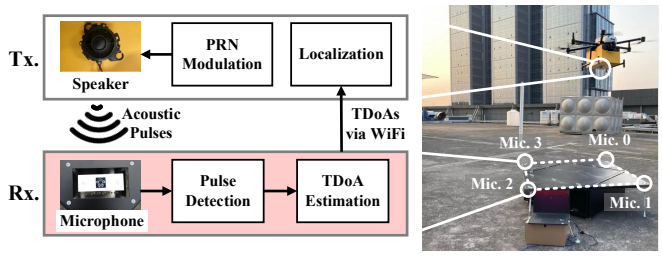


Figure 2: MicNest system architecture.

In summary, no existing localization system for mobile robots can simultaneously fulfill the three requirements outlined in the Introduction, thus enabling precise drone landing.

Localization with acoustic signals. Acoustic signals may be used for localization indoor [5, 20, 33, 35, 41, 43, 44, 58, 63, 66, 75] or outdoor [31, 42, 53, 78]. Many works utilize audible acoustic signals or actively transmit acoustic signals for indoor localization. For example, GuoGuo [20] proposes a fine-grained adaptive ToA (Time of Arrival) estimation approach to improve the location update frequency. VoLoc [58], Symphony [75], and MAVL [74] use a single microphone array to localize sources via wall reflections. Lin et al. [43] transmit the ultrasonic sound and exploit the non-linearity effect to localize devices. Some of these techniques achieve m-level accuracy, with a range limited to less than 20 m.

In outdoor localization, works exist that present the design of wireless acoustic sensor networks to achieve localization in vast areas with m-level accuracy [31, 53]. Li et al. [42] also present a machine learning technique to detect cars via acoustics and exploit the geometric information of roads to localize cars. These works deploy microphones across a vast area, while our deployment is limited to a landing platform not much larger than the drone.

Here again, the conclusion is that no existing technique can simultaneously fulfill the requirements at stake, either because of limited accuracy or coverage, or due to deployment constraints dictated by the target scenario.

3 MICNEST IN A NUTSHELL

Fig. 2 shows the system architecture of MicNest. The drone carries a speaker that transmits acoustic pulses continuously. Four distributed microphones are deployed at the corners of the landing platform to capture these pulses. MicNest localizes the drone by localizing the speaker.

PRN modulation. We adopt Pseudo-Random Noise (PRN) modulation to generate the pulses. Specifically, let

$$\boldsymbol{s} = [s_0, s_1, \ldots, s_n, \ldots, s_{N-1}]^T \quad (1)$$

indicate the acoustic pulse for a drone, where s_n denotes the PRN code of a pulse, and N is the pulse length. We take a drone's identifier as a random seed and generate a sequence of N Gaussian random variables as the N codes of the pulse. By doing so, the pulses transmitted by different drones are independent and thus orthogonal to each other. In MicNest, the code rate equals the sampling rate of the microphones, that is, 48 kHz. The corresponding

frequency band of pulses is 0-24 kHz. For continuously localizing the drone, pulses are transmitted repeatedly.

Pulse detection and TDoA estimation. First, we need to detect the pulses from the signals recorded by the microphones. Matched filters are a standard method to detect acoustic pulses. The idea is to consider the transmitted pulse as a template and to correlate it with the received signal. The signal received by a microphone is

$$\boldsymbol{x} = \alpha \boldsymbol{s} + \boldsymbol{w}, \tag{2}$$

where α is the attenuation factor[2], and $\boldsymbol{w} = [w_0, w_1, \ldots, w_{N-1}]^T$ denotes a vector of Gaussian white noise. To detect the pulse $\boldsymbol{s}$, the matched filter correlates $\boldsymbol{s}$ with the received signal $\boldsymbol{x}$. The output is

$$y = \boldsymbol{s}^T \boldsymbol{x} = \alpha \boldsymbol{s}^T \boldsymbol{s} + \boldsymbol{s}^T \boldsymbol{w}. \tag{3}$$

By feeding a stream of $\boldsymbol{x}$ into the matched filter, we obtain a stream of correlations y. Upon observing a correlation peak in the output, we consider a pulse to be detected.

Once an acoustic pulse is detected, we calculate ToAs, that is, the times when the pulse arrived at microphones, and time difference of arrivals (TDoAs), that is, the differences in ToAs. In MicNest, we compute TDoAs between opposite microphones on the landing platform, that is, <Mic. 0, Mic. 2> and <Mic. 1, Mic. 3> in Fig. 2. This is because opposite microphones have the largest inter-microphone distance and therefore yield the largest aperture [59]. The two TDoAs of pairs <Mic. 0, Mic. 2> and <Mic. 1, Mic. 3> are denoted as $\tau_{<0,2>}$ and $\tau_{<1,3>}$, respectively.

Localization. $\tau_{<0,2>}$ and $\tau_{<1,3>}$, are transmitted to the drone, for example, via WiFi. Based on these information, the drone can establish two hyperboloid equations.

Specifically, we build a 3D coordinate system and let the center of the landing platform be the origin. The coordinates of microphones are defined as $M_0 = (d, 0, 0)$, $M_1 = (0, -d, 0)$, $M_2 = (-d, 0, 0)$, and $M_3 = (0, d, 0)$, respectively. Given these, the two-sheeted hyperboloids oriented along the x-axis and y-axis are given by:

$$\begin{cases} |PM_0 - PM_2|^2 = 2a_0 = \text{abs}(\tau_{<0,2>} \times c) \\ |PM_1 - PM_3|^2 = 2a_1 = \text{abs}(\tau_{<1,3>} \times c) \end{cases}, \tag{4}$$

where $P = (P_x, P_y, P_z)$ denotes the drone coordinate, $|\bullet|^2$ is the Euclidean distance (2-norm), and c is the speed of sound.

Note that Eq. (4) is undetermined, since it provides only two constraints while drone coordinates have three unknowns, that are, P_x, P_y, and P_z. It would be possible to introduce additional constraints by using TDoAs of other microphone pairs. However, due to far-field effect, the solution to this system of equations would likely lead to oscillating behaviors, especially in the vertical direction, with the increase of drone distance from the landing platform. Differently, we estimate P_z based on sensors found aboard modern drones, such as barometers, ultrasound sensors, and downward-facing LIDARs. Similar techniques are routinely used in drone testbeds [4]. By determining P_z, we can use Eq. (4) to determine P_x and P_y.

4 TACKLING PULSE DETECTION

The *low SNR* of received pules and their *non-linear Doppler distortion* concur to make pulse detection difficult. This section elaborates on how we tackle these two challenges; then it introduces our novel pulse detector: Matched Filter Tree (MFT). As a reference through this section, Tab. 1 lists the mathematical symbols we use.

Table 1: Summary of mathematical symbols.

Term	Description
$T_c, T_c^{'}$	Code periods w/o and w/ Doppler effect
$G, G^{'}$	SNR gains w/o and w/ Doppler effect
v	Drone radial velocity
c	Sound Speed
Δ	Doppler time shift experienced by PRN
N	Number of PRN codes in a pulse
L	Number of synchronized PRN codes

4.1 Low SNR

Because of limited transmission power, propagation attenuation, background noise, and propeller interference, the SNR of received pulses is intrinsically low.

An effective solution to address this problem is to increase pulse length, thus improving the *SNR gain* of the matched filter. To understand the SNR gain as seen by the matched filter, we compare the SNRs of the received signal $\boldsymbol{x}$ and the output y. Considering Eq. (2), the SNR of $\boldsymbol{x}$ is

$$SNR_x = \frac{E\left[|\alpha \boldsymbol{s}|^2\right]}{E\left[|\boldsymbol{w}|^2\right]} = \frac{|\alpha|^2 \boldsymbol{s}^T \boldsymbol{s}}{E\left[\sum |w_n|^2\right]} = \frac{|\alpha|^2 \boldsymbol{s}^T \boldsymbol{s}}{N\sigma^2}, \tag{5}$$

where σ^2 is the noise variance. Similarly, the SNR of y is

$$SNR_y = \frac{E\left[|\alpha \boldsymbol{s}^T \boldsymbol{s}|^2\right]}{E\left[|\boldsymbol{s}^T \boldsymbol{w}|^2\right]} = \frac{|\alpha|^2 \boldsymbol{s}^T \boldsymbol{s} \cdot \boldsymbol{s}^T \boldsymbol{s}}{\boldsymbol{s}^T E\left[\boldsymbol{w}\boldsymbol{w}^T\right] \boldsymbol{s}} = \frac{|\alpha|^2 \boldsymbol{s}^T \boldsymbol{s}}{\sigma^2}, \tag{6}$$

where $E\left[\boldsymbol{w}\boldsymbol{w}^T\right]$ is the covariance matrix of $\boldsymbol{w}$, which is $\sigma^2 \boldsymbol{I}$. The SNR gain of the matched filter is thus given by

$$G = \frac{SNR_y}{SNR_x} = N. \tag{7}$$

Eq. (7) leads to a fundamental insight: *the SNR gain G equals the pulse length N*. This means that, at least in principle, we may trade time for a higher SNR. Unfortunately, doing so backfires in the presence of non-linear Doppler distortion, as explained next.

4.2 Non-Linear Doppler Distortion

The Doppler effect introduced by drone dynamics may seriously distort the acoustic pulses. The severity of Doppler effect is generally inversely proportional to signal speed. Due to the low speed of acoustic pulses, they heavily suffer from such distortion.

Code desynchronization. Doppler effect gradually makes the received pulses misaligned with the transmitted original pulses in the time domain, and ultimately desynchronize the received codes with the transmitted ones.

Consider the continuous-time waveform of the transmitted pulse $\boldsymbol{s}$ which is given by[3]

$$s(t) = \sum_{n=0}^{N-1} s_n \cdot \text{rect}(\frac{t - nT_c}{T_c}), 0 \le t \le NT_c. \tag{8}$$

[2]For simplicity, we assume all codes experience the same attenuation α.

[3]For simplicity, we use the Zero-Order Hold (ZOH) to model the Digital-to-Analog Converter (DAC).

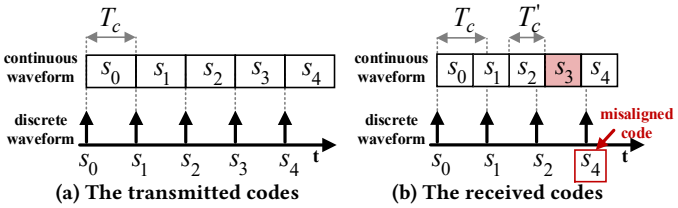


Figure 3: Illustration of code desynchronization.

Here, $\text{rect}(t)$ is the rectangular function that is 0 outside the interval $[0, 1)$ and 1 inside of it, whereas T_c denotes the code period, that is, $1/48kHz$ s in our implementation.

In the presence of Doppler effect, the codes of the received pulse expand or compress in time. The period of these codes can be calculated as [65]

$$T_c^{'} = \left(1 - \frac{v}{c}\right) T_c, \tag{9}$$

where v denotes the drone velocity[4]. Given this, the received pulse distorted by Doppler effect is

$$s^{'}(t) = \sum_{n=0}^{N-1} s_n \cdot \text{rect}\left(\frac{t - nT_c^{'}}{T_c^{'}}\right), 0 \le t \le NT_c^{'}. \tag{10}$$

However, the receiver simply samples the pulse with the original code period T_c. The discrete-time waveform of the received pulse is

$$s_n^{'} = s^{'}(t) \cdot \delta(nT_c), n = 0, 1, \ldots, \tag{11}$$

where $\delta(t)$ is the Dirac delta function. Note that the period of received codes is $T_c^{'}$ while the sampling period is T_c. This means that each sampled code is shifted in time by

$$\Delta = T_c - T_c^{'} = \frac{v}{c} \cdot T_c. \tag{12}$$

To make things worse, the time shift accumulates over multiple samples. Consider Fig. 3 as an example. It can be calculated that when $n \ge L = \lfloor c/v \rfloor$, the accumulated time shift that $s_n^{'}$ experiences is larger than the original code period T_c. This means that the codes received hereafter are desynchronized with the transmitted codes, therefore the following holds:

$$\begin{cases} s_n^{'} = s_n & \text{if } n < L \\ s_n^{'} \ne s_n & \text{if } n \ge L \end{cases} \tag{13}$$

Impact of code desynchronization. The derivation of the SNR gain in Eq. (7) assumes that the transmitted codes, that is, the template, are synchronized with the received codes. This assumption may not hold due to code desynchronization. In such a case, the SNR gain of the matched filter can be re-written as

$$G^{'} = \frac{SNR_y^{'}}{SNR_x^{'}} = \left(\frac{|\alpha|^2 \boldsymbol{s}^T \boldsymbol{s}^{'} \cdot \boldsymbol{s}^T \boldsymbol{s}^{'}}{\sigma^2 \boldsymbol{s}^{'T} \boldsymbol{s}^{'}}\right) / \left(\frac{|\alpha|^2 \boldsymbol{s}^{'T} \boldsymbol{s}^{'}}{N\sigma^2}\right) = N\left(\frac{\boldsymbol{s}^T \boldsymbol{s}^{'}}{\boldsymbol{s}^{'T} \boldsymbol{s}^{'}}\right)^2, \tag{14}$$

where $\boldsymbol{s}^{'}$ denotes the received pulse $[s_0^{'}, s_1^{'}, \ldots, s_{N-1}^{'}]^T$.

Based on Eq. (13), if the number of codes n is less than L, then $\boldsymbol{s}^{'} = \boldsymbol{s}$. In this case, there is no code desynchronization, and $G^{'} = G = N$. Instead, if $n \ge L$, then only the first L codes of $\boldsymbol{s}^{'}$ equals those of $\boldsymbol{s}$ and the remaining codes are desynchronized. Therefore,

$$G^{'} = N\left(\frac{\sum_{n=0}^{L-1}(s_n^{'})^2 + \sum_{n=L}^{N-1} s_n s_n^{'}}{\sum_{n=0}^{N-1}(s^{'})^2}\right)^2 = N\left(\frac{\sum_{n=0}^{L-1}(s_n^{'})^2}{\sum_{n=0}^{N-1}(s_n^{'})^2}\right)^2$$
$$\approx N(\frac{L}{N})^2 = \frac{L^2}{N}, \quad N > L. \tag{15}$$

Here, the reason for $\sum_{n=L}^{N-1} s_n s_n^{'} = 0$ and for the approximation in the second line is that codes in our pulse are a sequence of independent pseudo-random Gaussian variables [28]. As a result, once the pulse length N exceeds L, the SNR gain degrades with the increase of pulse length.

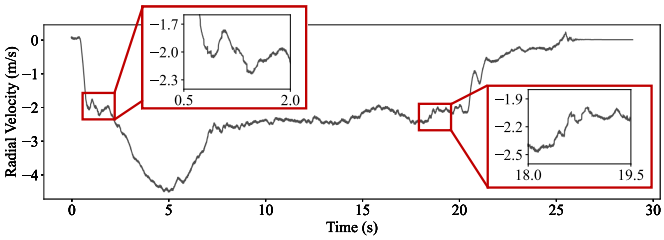


Figure 4: A drone's radial velocity during landing.

We are now facing a catch-22 situation. Eq. (7) suggests that a long pulse length helps mitigate the low-SNR problem. Conversely, Eq. (15) indicates that the maximum pulse length that can improve SNR gain is $L = \lfloor c/v \rfloor$, whose value is upper-limited by the (low) speed of acoustic signals c. Let us consider a concrete example to illustrate the problem. Suppose the sound speed is 343 m/s and the drone speed is 6 m/s. Thus, a complete code desynchronization will occur after $L = \lfloor 343/6 \rfloor = 57$ codes, as per Eq. (13). On the other hand, as we indicate in Sec. 6, the pulse duration should be at least 50 ms so as to localize a long-range drone, thus the corresponding pulse length N is 2400 at a sampling rate 48 kHz. The huge gap between $N = 2400$ and $L = 57$ indicates that Doppler distortion significantly hinders addressing the low-SNR problem by increasing the pulse length.

Physical drone dynamics. Should the drone velocity be constant, the corresponding Doppler distortion would be linear. If we were in this situation, we might simply determine what drone velocity compensates the Doppler effect. This is precisely the idea of the matched filter bank [19, 36].

In practice, however, the drone velocity is not constant, leading to non-linear distortion. As an example, Fig. 4 plots the radial velocity of a drone when it is landing automatically from a 50 m altitude, based on location and velocity information obtained by an RTK system in an open area and the on-board IMU. The drone velocity fluctuates rapidly during landing in response to commands from the on-board flight control loops [17], which rapidly change the drone motion to maintain stable flight on a predetermined route.

Flight control loops operate at 100Hz-32kHz [1, 2, 56]. This means that a drone may change its velocity at sub-10-ms scales. On the other hand, Eq. (7) shows that in order to take advantage of the SNR gain of the matched filter, we should increase pulse length. Our real-world experimental evaluation, reported in Sec. 6, indicates pulse duration should be no less than 50 ms to localize the drone robustly. We may thus expect that multiple motions occur

[4]Unless otherwise specified, drone velocity refers to radial velocity of drone with respect to the ground port. The velocity is negative if the drone moves towards the ground port.

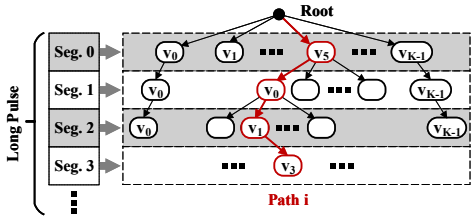


Figure 5: Illustration of a search tree.

within the duration of one pulse. Therefore, pulses undergo non-linear distortion and different codes experience different degrees of Doppler effect, hindering detection.

4.3 Matched Filter Tree

We present a novel pulse detector, Matched Filter Tree (MFT), to detect a low-SNR pulse subject to non-linear distortion. The key idea is to model pulse detection as a tree search.

Intuition. Fig. 5 illustrates the idea. We split one pulse into M equal *segments* denoted as Seg. 0, 1, ... with segment length $N_{seg} = N/M$. The segments are short enough that the drone velocity can be considered constant within the duration of a segment. Therefore, each segment experiences a *linear* Doppler distortion.

We build a search tree where the nodes at each level correspond to the possible drone velocities during the transmission of a segment. For each segment, we consider the K possible velocities to compensate the Doppler shift it experiences. Ideally, if all segments are compensated with the correct velocities, the new pulse spliced by the compensated segments restores its code synchronization with the received pulse, eliminating the non-linear distortion. Sec. 5 discusses the setting of the parameters at hand, including the choice of M, and the search resolution for velocity.

Searching the solution. Let velocities $<v^{(0)}, v^{(1)}, \ldots, v^{(M-1)}>$ be one possible combination of candidate velocities, for example, corresponding to path i in Fig. 5, where $v^{(m)}$ denotes the candidate velocity for Seg. m. We perform the following steps to check whether the candidate velocities compensate the non-linear distortion:

- **S1 (compensation):** for each velocity $v^{(m)}$, we estimate the Doppler shift that Seg. m suffers as $(v^{(m)}/c) \cdot T_c$, according to Eq. (12)); we compensate this Doppler shift by resampling this segment with spacing $(c - v^{(m)})/c$.
- **S2 (concatenation):** we concatenate the resampled versions of M segments into a new pulse, denoted as $s_i^{'}$.
- **S3 (correlation):** we take the new pulse $s_i^{'}$ as a new template and correlate it with the received signal.

If velocities $<v^{(0)}, v^{(1)}, \ldots, v^{(M-1)}>$ along path i match the actual drone velocities, the new pulse $s_i^{'}$ is again synchronized with the received pulse; thus it has maximum correlation with the received signal because the non-linear distortion is minimized. Therefore, the problem of detecting pulses corresponds to finding a solution path in the search tree that can *minimize the non-linear distortion.*

A straightforward solution is exhaustive search, that is, visiting every path of the search tree. For each such path, we use the velocities along the path to compensate the Doppler shift of the pulse, as per **S1** and **S2**, and to calculate the correlation, as per **S3**. After visiting all paths, we select the path whose corresponding correlation has the maximum value; the maximum correlation value means that the non-linear distortion is minimized.

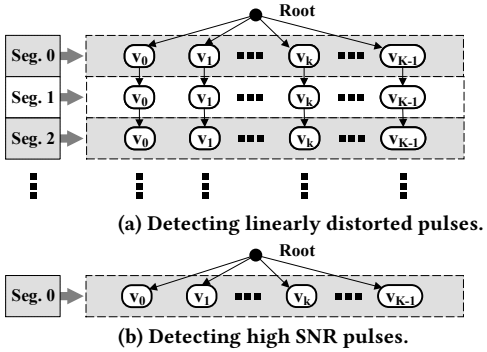


Figure 6: Two specific cases that MFT can be narrowed down to.

The processing overhead of an exhaustive search is, however, unacceptable for MicNest, because of the low-latency requirement discussed earlier. The search space indeed grows exponentially with the number of segments M. Given the computational complexity of the correlation operation in **S3**, it is nearly impossible to visit every path and return a solution at low latency. Note that it is also infeasible to search for the solution path incrementally or greedily, that is, choose the candidate velocity that can maximize the correlation at each level of the tree. This is because the length of each segment is so short that the SNR gain of segment-audio correlation is immaterial. The noise typically dominates the correlation. Sec. 5 explains how we accelerate the tree search.

With the help of MFT, we can compensate for the non-linear distortion in the received pulses. This means that the problem of code desynchronization can be tackled, and we can avoid the dilemma that a long pulse length can undermine the SNR gain. In other words, it is safe for us to choose a longer pulse on demand to boost the SNR of the MFT's outputs (see Eq. (7)). In Sec. 6, we study the pulse length that MicNest demands.

4.4 Other MFT Uses

Above, we show that MFT is feasible to detect pulses suffering from both *low SNR* and *non-linear distortion*, which is indeed a challenging task. In fact, MFT can also be used for detecting pulses in other cases, as shown in Fig. 6:

- **Detecting linearly distorted pulses:** Fig. 6(a) shows this case. When the received pulses suffer from linear distortion[5], all segments experience the same degree of distortion. Given this, we can remove the unfeasible search paths and prune the search space from an exponential one to a linear one. This kind of MFT is equivalent to matched filter bank [19, 36], which is widely used in GPS receivers.
- **Detecting high SNR pulses:** A high SNR of received pulses can lower the requirement for the SNR gain of MFT. This means that the pulse length can be reduced, and correspondingly the depth of the search tree can be reduced. As illustrated by Fig. 6(b), when the SNR is sufficiently high, the search tree can be chopped to a single layer, reducing the search space.

[5]In addition to Doppler effect, many hardware imperfections can also introduce linear distortion, such as carrier frequency offset (CFO) and sampling frequency offset (SFO).

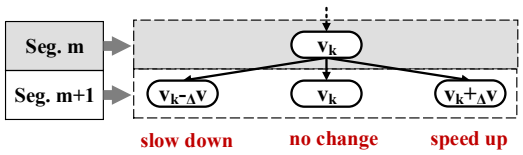


Figure 7: To prune the search space, the branching factor of the search tree is aggressively reduced to three.

MFT therefore enjoys the flexibility to adapt to detection tasks according to signal quality. Meanwhile, MFT is an extension to a matched filter. By replacing the template of MFT, MFT can detect not only PRN pulses used by MicNest, but also other types of signals, such as FMCW signals and OFDM signals. We believe MFT is a general detection tool and may be equally applied to other signal processing tasks.

5 TACKLING THE LATENCY CHALLENGE

We present three methods to accelerate the tree search.

5.1 Tree Pruning

We note that the drone velocity does not change abruptly: the velocity of the next segment is unlikely to considerably deviate from that of the current segment. This observation allows us to reduce the branching factor of the tree, that is, the number K of candidate velocities. We expect this to abate the processing overhead.

In our design, we aggressively reduce the branching factor to three, as illustrated in Fig. 7. Suppose v_k is chosen as the velocity of the m-th segment. There are only three candidate velocities for the next segment, that is $v_k - \Delta v$, v_k, and $v_k + \Delta v$, where Δv denotes the search resolution of velocity. By doing so, the search space is significantly reduced.

However, in order to safely prune the tree, we have to satisfy the following two constraints.

- **C1:** ensure that the solution path is within the pruned search space. That is
$$\Delta v \geq N_{\text{seg}} \cdot T_c \cdot a_{\max}, \tag{16}$$
where $a_{\max}$ denotes the maximum drone acceleration. Intuitively, this requirement expects a large Δv so that the candidate velocities along the considered paths can catch up with the rapid change of drone velocity.
- **C2:** ensure that the search resolution Δv is fine enough that the Doppler shift of each segment can be compensated. In fact, each candidate velocity is a numerical representation of the actual velocity with resolution Δv. Due to numerical error, the Doppler shift may not be completely compensated. According to Eq. (12), the accumulated Doppler shift of segment with length N_{seg} can be calculated as $\sum_{n=0}^{N_{\text{seg}}-1} \frac{v[n]}{c} \cdot T_c$, where $v[n]$ denotes the drone velocity during code s_n. After compensating the Doppler shift with v_k, the residual Doppler shift δ_{shift} is $\delta_{shift} = \sum_{n=0}^{N_{\text{seg}}-1} \frac{v[n]-v_k}{c} \cdot T_c$. Given the resolution Δv, the upper bound of δ_{shift} is $N_{\text{seg}} \cdot \frac{\Delta v}{2c} \cdot T_c$. To avoid introducing code asynchronization additionally, the residual Doppler shift δ_{shift} should be less than T_c:
$$N_{\text{seg}} \cdot \frac{\Delta v}{2c} \cdot T_c < T_c. \tag{17}$$

Obviously, constraints **C1** and **C2** are conflicting since **C1** expects a larger Δv while **C2** expects a smaller one.

We notice that segment length N_{seg} plays a key role in fulfilling both **C1** and **C2**. N_{seg} needs to be small enough that during each segment the drone velocity can be assumed as constant. This assumption also contributes to the residual Doppler shift because the actual velocity is not constant. The resulting maximum possible deviation between the actual velocity and the candidate velocity is $N_{\text{seg}} \cdot T_c \cdot a_{\max} + \Delta v$. Therefore, Eq. (17) should be modified as
$$N_{\text{seg}} \cdot \frac{N_{\text{seg}} \cdot T_c \cdot a_{\max} + \Delta v}{2c} \cdot T_c < T_c. \tag{18}$$

In a nutshell, we should satisfy the constraints of Eq. (16) and Eq. (18) before aggressively pruning the tree to three branches. In our implementation, $T_c = 1/48k$ s. The maximum acceleration of our drones $a_{\max}$ is 8 m/s^2. Given these, we set N_{seg} to 240 and Δv to 0.1 m/s, ensuring a sufficient search space and providing a promising search resolution.

5.2 Correlation Acceleration

Correlation is the most time-consuming operation during tree search (**S3** in Sec. 4.3). Here, we reduce its time cost.

Let us define $Cor_{<\boldsymbol{v_0},\boldsymbol{v_1}>}[\tau]$ as the correlation function between vectors $\boldsymbol{v_0}$ and $\boldsymbol{v_1}$, and $\tilde{\boldsymbol{s}}_{\text{seg}}^{(m)}$ as the compensated version of Seg. m, and $N_{\text{seg}}^{(m)}$ as the length of $\tilde{\boldsymbol{s}}_{\text{seg}}^{(m)}$.

We find that the correlation between the compensated pulse $\tilde{\boldsymbol{s}}$ and the received audio $\boldsymbol{y}$ can be calculated by summing the M compensated segment's correlation with the received audio $\boldsymbol{y}$:
$$Cor_{<\tilde{\boldsymbol{s}},\boldsymbol{y}>}[\tau] = \sum_{m=0}^{M-1} Cor_{<\tilde{\boldsymbol{s}}_{\text{seg}}^{(m)},\boldsymbol{y}>}[\tau + L^{(m-1)}], \tag{19}$$
where $L^{(m)}$ is defined as $\sum_{k=0}^{m} \tilde{N}_{\text{seg}}^{(k)}$. We can rewrite the compensated pulse $\tilde{s}$ as a concatenation of M compensated segments:
$$\tilde{\boldsymbol{s}}[n] = \sum_{m=0}^{M-1} \text{bool}[L^{(m-1)} \leq n < L^{(m)}] \cdot \tilde{\boldsymbol{s}}_{\text{seg}}^{(m)}[n - L^{(m-1)}], \tag{20}$$
where bool[*condition*] is a boolean function that equals 1 if *condition* is true and 0 otherwise. Substituting Eq. (20) into the definition of correlation leads to Eq. (19):
$$\begin{aligned}
Cor_{<\tilde{s},y>}[\tau] &= \sum_{n=0}^{\tilde{N}-1} y[n+\tau]\tilde{s}[n] \\
&= \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} y[n+\tau]\mathbf{1}[L^{(m-1)} \leq n < L^{(m)}] \cdot \tilde{s}_{seg}^{(m)}[n - L^{(m-1)}] \\
&= \sum_{m=0}^{M-1} \sum_{n=L^{m-1}}^{L^m} y[n+\tau] \cdot \tilde{s}_{seg}^{(m)}[n - L^{(m-1)}] \\
&= \sum_{m=0}^{M-1} \sum_{n=0}^{\tilde{N}_{seg}^{(m)}} y[n+\tau+L^{(m-1)}] \cdot \tilde{s}_{seg}^{(m)}[n] \\
&= \sum_{m=0}^{M-1} Cor_{<\tilde{s}_{seg}^{(m)},y>}[\tau + L^{(m-1)}].
\end{aligned} \tag{21}$$

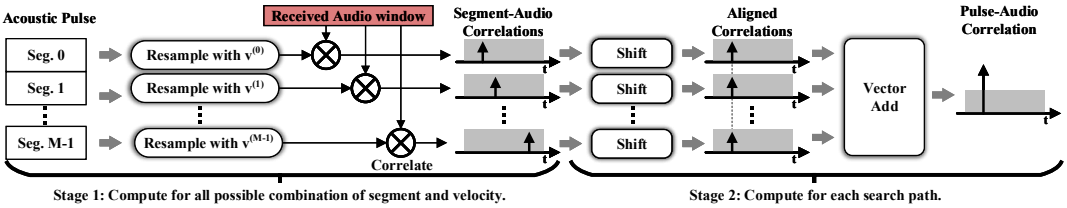


Figure 8: To accelerate the operation of correlation, we take two stages to do so: We first calculate and cache all possible segment-audio correlations. During the tree search, we efficiently compute one pulse-audio correlation by shifting and adding up the segment-audio correlations elementwisely.

Eq. (19) reveals a key fact that can be exploited to accelerate correlation operation. That is, the pulse-audio correlation $Cor_{<\tilde{s},y>}$ can be decomposed into multiple segment-audio correlations. In other words, if segment-audio correlations are available, pulse-audio correlation can be calculated by adding M vectors, that is, M segment-audio correlations.

Given this, we apply a two-step process, as shown in Fig. 8:

- **Stage 1:** this stage is performed before the tree search. We calculate and cache all possible segment-audio correlations. Specifically, for each Seg. m and for each candidate velocity $v^{(m)}$, we resample this segment with $v^{(m)}$ and then correlate it with a window of received signals. The resulting segment-audio correlation is then saved to a lookup table with key $<m, v^{(m)}>$.
- **Stage 2:** this stage is performed during the tree search. For one search path, we retrieve all required segment-audio correlations from the lookup table. For each Seg. m, we shift its segment-audio correlation by L^{m-1}. We then add up M shifted segment-audio correlations elementwisely.

Note that the total number of segment-audio correlations is not an exponential function of segment number M, but a linear function, that is, $\sum_{m=0}^{M-1}(2m+3)$. This is because after reducing the branching factor to three (see Sec. 5.1), Seg. m has only $2m+3$ candidate velocities. So Stage 1 can be finished in a short time (about 3.1 ms)

Also note that, the vector add in Stage 2 can be efficiently parallelized. We take advantage of native NVIDIA CUDA kernel[6] to further accelerate Stage 2. In our implementation, searching one tree path takes only 5.3 μs on the NVIDIA RTX 3070, on average.

5.3 Heuristic Search

Instead of visiting all tree paths in a brute-force way, we adopt a heuristic method to reduce the total visit count.

Our method is similar to Monte-Carlo Tree Search (MCTS) [21, 39, 62]. Our insight is: for each search path, its corresponding maximum correlation values contain the useful information, which can be exploited to guide towards the solution path in the search tree. The path that has a larger correlation value is more likely to be closer to the solution path, and thus the nodes along this path are more promising to be the nodes of the solution path. Therefore, we pay more attention to nodes that look promising, so as to avoid traversing the search tree exhaustively.

[6]Native CUDA kernel also allows us to perform shift operation efficiently. This is because by passing different memory offsets of vectors to the CUDA kernel, we can implicitly perform shift operation without memory copy.

Our method consists of repeated rounds. For each round, we perform three procedures:

- **Batch path selection:** we first generate a batch of tree paths in the search tree. The batch size is denoted as B. During the generation of each path, we start from the root node and choose the next child with the highest Upper Confidence Bound (UCB) value [39], defined as

$$\text{UCB(node } j) = \text{score}^j + K_{ucb}\sqrt{\frac{\log N_{\text{vis}}}{N^j_{\text{vis}}}}, \tag{22}$$

  where score^i denotes the *score* of node j, explained later, N_{vis} is the total number of paths that has been visited, N^j_{vis} denotes the number of times that node j have been selected, and K_{ucb} is an empirical parameter that used to trade off between *exploration* and *exploitation* [62].
- **Batch path evaluation:** For each path from the B generated tree paths, we compute its corresponding correlation using the accelerated method introduced in Sec. 5.2. We treat each path as a comparison game. The paths whose correlations are top B_0 largest are regarded as *win* ($B_0 < B$), and other paths as *lose*.
- **Backpropagation:** we then use the game results of B paths to update the score of nodes. The score of node j is simply defined as the winning rate of paths that pass through it: $\text{score}^j = N^j_{win}/N^j_{\text{vis}}$, where N^j_{win} denotes the winning times of paths passing through node j.

It can be expected that, as the number of rounds grows, the nodes of the solution path will be visited more and more frequently as their node score^j are gradually growing. Therefore, our method will converge to the solution path. In our implementation, the batch size B is empirically set to 20, B_0 is 1, and K_{usb} is 0.4. The total visit count is limited to 5000 (see Sec. 6.3). By doing so, the time cost of detecting one pulse is less than 30 ms.

6 EVALUATION

Our evaluation of MicNest is four-pronged and entirely based on real-world experiments. Following a description of the implementation we use and of the experimental setting in Sec. 6.1, we report in Sec. 6.2 on the crucial performance metrics we target: localization accuracy and processing latency. We proceed by investigating the influence of key system parameters in Sec. 6.3. Further, we study in Sec. 6.4 the impact on MicNest of external factors, such as drone speed, background noise, and ambient temperature. We conclude in Sec. 6.5 by demonstrating MicNest's ability to concurrently localize multiple drones.

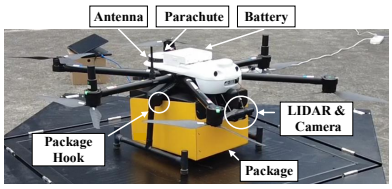


Figure 9: The delivery drone used for evaluation. Besides the speakers, the rest of the drone equipment is part of a standard drone configuration and would be present anyways on any professional-grade drone platform.

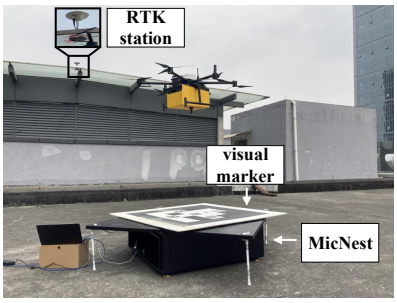


Figure 10: Experimental setting. Our website [47] presents a vivid demonstration.

The results we collect across a total of 40 flight hours lead us to five key conclusions:

1) MicNest provides a range up to 120 m and attains *cm-level* accuracy as the drone approaches the platform;
2) The rate of location updates returned by MicNest is *compatible with use in flight control loops*;
3) The performance of MicNest *improves as the drone approaches the platform*, that is, where it matters the most;
4) MicNest is *marginally affected* by factors it cannot control, such as drone speed, noise, and temperature;
5) MicNest can *localize multiple drones* with a limited degradation of the localization accuracy.

The rest of this section provides experimental evidence.

6.1 Implementation and Setting

We use drone equipment and a deployment setting that closely mimic actual applications.

Drone. As shown in Fig. 9, we use a custom drone manufactured by Meituan that are currently exploring the feasibility of instant deliveries with drones. The drone is equipped with six propellers each hooked to a brushless TMotor and is steered by the PX4 [9] flight controller, running at 400 Hz. The drone has a payload capacity up to 2.6 kg at liftoff, which is the maximum load that local regulations allow. The altitude is jointly estimated by an on-board barometer and a Benewake downward-facing LIDAR.

The only additional equipement on the drone, other than what would normally be present on any professional-grade drone, are the speakers. They are attached to the bottom of the drone and should be as light as possible not to negatively impact the payload capacity. We use a VISTEON speaker weighing a mere 47 g. The applied voltage and operating current are 12.6 V and 45 mA. According to our measurements, the speaker draws less than 0.1% of the total battery power. The speaker volume is empirically set to 70-75 dB SPL (measured at 1 m distance), which is arguably moderate.

Landing platform and software. We build a squared foldable landing platform, shown in Fig. 10, measuring 1 m x 1 m and 1.41 m x 1.41 m when folded or unfolded, respectively. Four omni-directional SPK0641HT4H digital microphones are installed at the corners of the platform. The distance between two microphones along the diagonal is 1.86 m.

We use an XMOS XU216 data acquisition board to drive and sample the microphones, so that the four signals are synchronized. The sampling rate is 48 KHz. The board then streams the audio signals to a laptop via USB UAC 2.0 with a latency lower than 0.5 ms. We use a high-pass filter with a cutoff frequency of 500 Hz to pre-process the audio signals.

MFT is implemented in C++ with the CUDA 11.0 library, running on a machine with an Intel i9-11900H CPU, 32 GB memory, and an NVIDIA RTX 3070 GPU.

Ground truth. We conduct the experiments in a secluded area on the roof of a building, where the reception of GPS signals is of very high quality. We deploy an RTK base station close to the experimental site, as shown in Fig. 10, which keeps rebroadcasting the phase of the GPS signal it observes.

In such a setting, the RTK processing on the drone works at high fidelity, especially because it does not experience the performance degradation or outage problems that occur in an operational site, for example, in a urban canyon, as mentioned in Sec. 2. Therefore, we use the localization results of RTK as ground truth. We compare the performance of MicNest with ArUco markers [50], a state-of-the-art visual localization system. We place an ArUco marker of 1.5 m × 1.5 m on top of the landing platform, as shown in Fig. 10.

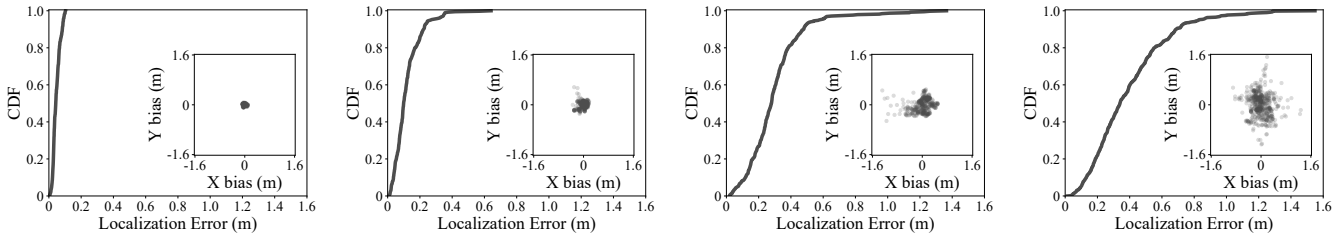
Flight trajectories. The drone operates automatically during the experiments, exactly as it would in an actual application. We use QGroundControl [25] as ground station control software to plan the flight trajectories. In addition to a hovering mode that keeps the drone stable at a given position, we consider three possible flight trajectories:

- In *vertical flight*, the drone takes off and vertically climbs to a given altitude; next, it lands back onto the platform by following the same trajectory in the opposite way.
- In a *squared spiral*, the drone takes off vertically, climbs to a given altitude, and flies twice along a squared spiral trajectory at constant altitude; next, it flies horizontally back to the starting point and lands vertically.
- In a *dense squared spiral*, the trajectory is the same as the squared spiral above, yet the side length of the square is increased by 2 m every two turns, instead of 10 m; the drone flies for ten rounds in total, rather than two.

6.2 Accuracy and Latency Performance

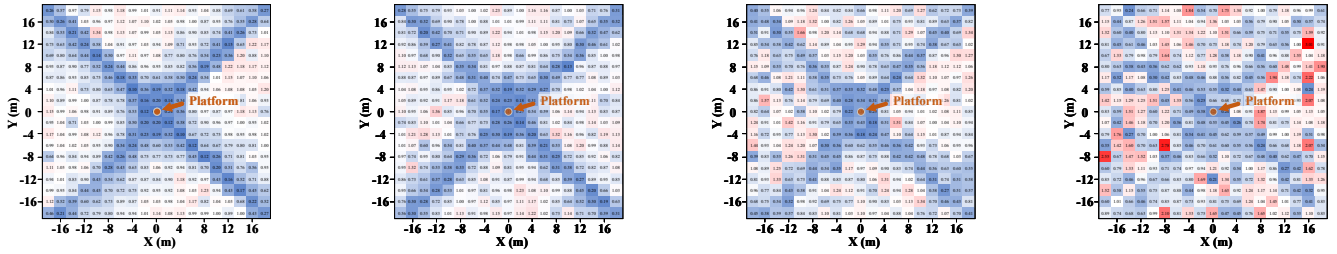
Localization accuracy is a function of several factors.

Impact of altitude. We program the drone to perform a *vertical flight* up to 120 m. We plot the cumulative distribution function (CDF) of the localization error, compared to RTK that represents ground truth, at different altitude intervals in Fig. 11(a)-(d). Note that MicNest calculates the horizontal coordinates of the drone



(a) 0-20 m altitude (b) 20-50 m altitude (c) 50-80 m altitude (d) 80-120 m altitude

Figure 11: Localization error compared to altitude.



(a) 25 m altitude (b) 50 m altitude (c) 75 m altitude (d) 100 m altitude

Figure 12: Heatmaps of localization error on the horizontal plane at different altitudes.

at a given altitude. The scattered plot in each figure shows all the localization biases of MicNest within the specified altitude interval.

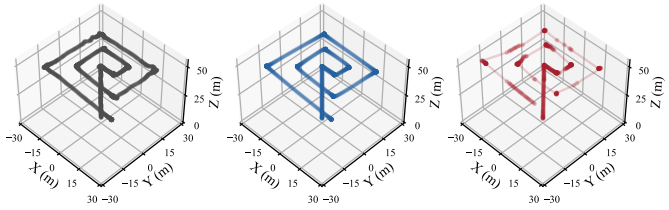
When the altitude is below (above) 20 m (80 m), the median error is 0.043 m (0.339 m). On average, the relative error, that is, the absolute localization error over the distance to the platform, is only 0.53%. The plots also demonstrate that the localization error *decreases as the drone approaches the platform.* This is indeed a desired characteristic for a localization system enabling precise landing, that is, *the performance improves when it matters the most.* In MicNest, this is due to: *i)* the far-field effect: when the drone flies low, its slight movements may result in a notable fluctuation in TDoA, whereas at higher altitudes, the TDoA fluctuations become less and less distinguishable; and *ii)* signal attenuation: acoustic signals experience more attenuation at longer distances and appear to be noisier, this also explains why there are more and more outliers as the altitude increases.

Impact of horizontal distance. We program the drone to fly a *dense square spiral*. Fig. 12 shows a heatmap representation of the distribution of the localization error across the bidimensional plane at four different altitudes.

Generally, MicNest 's localization error tends to decrease as the drone is horizontally close to the platform. MicNest enjoys the highest localization resolution when the drone horizontally aligns with the center of the platform. The reasons for this behavior are similar to the ones explaining the performance at different altitudes, discussed above.

An interesting observation is the visible "X" pattern in the heatmaps, which corresponds to the higher localization accuracy. This pattern corresponds to the two vertical bisectors of the diagonal microphones. TDoA-based localization has indeed the highest spatial resolution in these conditions.

Localization trajectory. As an example, Fig. 13(a)-(c) show the localization results as the drone flies a *squared spiral* trajectory at 50 m altitude. An illustrative video is available [47].



(a) MicNest (b) RTK (c) Visual marker

Figure 13: Drone trajectories localized by MicNest, RTK, and using the visual marker. We provide a illustrative video of this experiment as well as the results at other altitudes on our website [47].

The plots demonstrate how MicNest and RTK successfully localize the drone throughout the whole flight. In contrast, the visual marker works intermittently, because it is difficult for the camera on the drone to capture the visual marker, especially at higher altitudes. Results at different altitudes are nonetheless available on our website [47].

Latency. We measure the localization latency as the time between the moment a PRN pulse is transmitted and the moment the corresponding localization result is obtained. Four components contribute to this quantity: *i)* the *propagation delay*, that is, the time needed for acoustic signal to reach the microphones; *ii)* the *transmission delay* that equals the PRN pulse length (50 ms); *iii)* the *processing delay* that is dominated by the time for MFT to detect pulses, which we examine later; and *iv)* the *communication delay*, that is, the time for TDoAs to be sent back to the drone via WiFi.

Fig. 14 plots the aggregated latency of localization using MicNest at different altitudes. As expected, the only varying latency component is the propagation delay. Our measurements indicate that the mean processing delay is 29.7 ms with a standard deviation of 0.6 ms, whereas the mean WiFi delay is 11 ms. The transmission

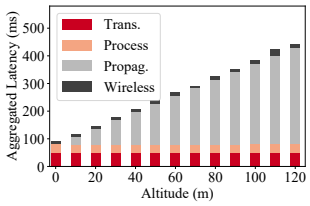


Figure 14: Aggregated latency.

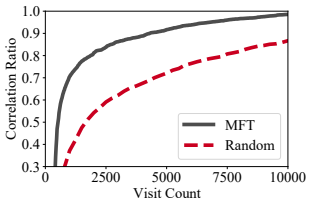


Figure 15: Visit count of MFT.

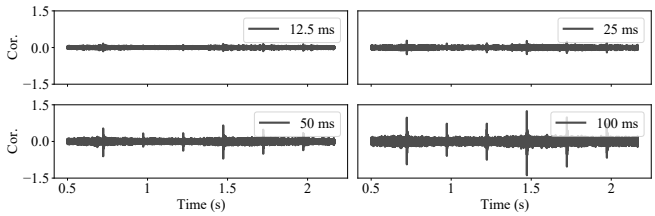


Figure 16: Pulse-audio correlations with different pulse lengths at 120 m altitude.

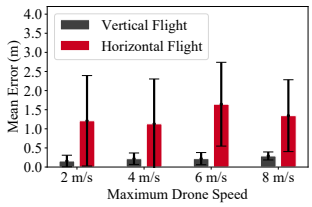


Figure 17: The impact of drone speed.

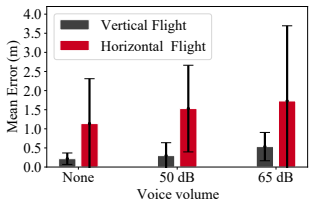


Figure 18: The impact of background noise.

delay is also fixed. The plot, however, shows that the aggregated latency changes with the distance linearly. Since the sound speed is relatively slow, the aggregated latency is dominated by the propagation delay when the drone is far from the platform. As a whole, however, MicNest can provide localization updates at a rate more than sufficient to feed the flight control loop [17, 52].

6.3 Ablation Study

Two system parameters are crucial in determining the latency. We focus on each of them in turn.

Maximum MFT visit count. The processing delay of MFT is determined by the number of tree paths that are visited looking for the solution. Intuitively, the more search paths we visit, the larger correlation value the MFT outputs and thus the larger SNR gain the MFT yields.

We execute an experiment flying a *squared spiral* at 120 m altitude. Fig. 15 plots the correlation ratio[7] of MFT outputs as a function of the maximum visit count. In comparison, the results of a random search, that is randomly picking a tree path that is not yet visited, are also shown. Fig. 15 shows that when the visit count is 5000, the correlation ratio of the MFT and if the random search is 0.92 and 0.72, respectively. In our implementation, we set the maximum visit count of MFT to 5000, which ensures that pulse detection can be completed in a deterministic period. The corresponding time required for detecting a pulse is 26.5 ms.

Impact of pulse length. We study how pulse length impacts pulse detection. We make the drone hover at 120 m altitude and repeatedly transmit acoustic pulses with 250 ms length. We correlate the received audio with pulse templates with varied lengths: 12.5 ms, 25 ms, 50 ms, and 100 ms.

Fig. 16 plots the pulse-audio correlations. We observe distinguishable correlation peaks when the pulse length is 50 ms or 100 ms. When the pulse length is 12.5 ms, the correlation is almost overwhelmed by the noise floor. Specifically, the missing rate of pulse detection for lengths 12.5 ms, 25 ms, 50 ms, and 100 ms are 23.25%, 5.70%, 1.08%, and 0.77%, respectively. We set the pulse length to 50 ms, striking a balance between detection accuracy and transmission delay. The resulting location update frequency is 20 Hz.

6.4 Impact of External Factors

Factors that are not under the direct control of MicNest may influence its performance, including drone speed, background noise, and sound speed.

Drone speed. The drone flies a *squared spiral* at 50 m altitude with different speeds: 2 m/s, 4 m/s, 6 m/s, and 8 m/s[8].

Since the whole trajectory consists of both vertical and horizontal parts, Fig. 17 plots the localization errors in either dimension. The error along the vertical parts appears lower than along the horizontal ones. This is because the drone is horizontally aligned to the center of the landing platform during the vertical parts. Most importantly, the plot provides evidence that the drone speed has a negligible impact on the accuracy performance of MicNest. In turn, this demonstrates that the MFT can search for the correct drone speed and compensate the Doppler effect effectively.

Background noise. We place a loudspeaker 1.5 m away from the center of the landing platform to emulate a source of noise. The speaker plays music continuously at a frequency between 200 Hz and 3.5 kHz with two volumes: 50 dB and 65 dB. Note how the latter setting is effectively close to the volume of the MicNest speaker aboard the drone. The drone flies again a *squared spiral* at 50 m.

Fig. 18 illustrates the performance degradation with increasing noise. The mean error of the horizontal flight is 1.13 m in the case of no noise and increases to 1.73 m at 65 dB noise, which represents a case of strong background noise. By adopting advanced noise reduction techniques, its impact can be further mitigated [12, 68].

Sound speed. The speed of sound changes with temperature. As a rule of thumb, a 1 °C increase corresponds to a 0.6 m/s increase in sound speed [77]. To investigate this aspect, we program the drone to perform a *vertical flight* up to 120 m altitude. We conduct the flight when the sound speed is 342 m/s. Then we parameterize the sound speed with different values: 336 m/s, 339 m/s, 342 m/s, 345 m/s, and 348 m/s. Fig. 19 shows the results in localization error at different altitudes. The performance difference at different sounds speeds is negligible, yet in real operations one can calibrate the

[7] For visit count N_{vis}, the correlation ratio is defined as the ratio of the best correlation values of the first N_{vis} paths to the maximum correlation value of the brute-force search.

[8] Note that the drone's speed cannot be perfectly fixed during flight; these values may be regarded as the maximum speed that the drone can reach while flying a predetermined trajectory.

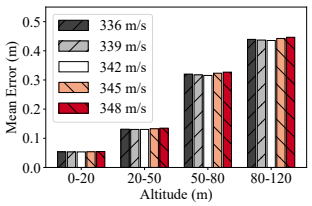


Figure 19: The impact of sound speed.

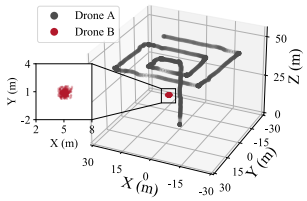


Figure 20: The localization trajectories of multiple drones.

parameter of sound speed in MicNest by measuring the sound speed[9] to further refine the performance.

6.5 Localization of Multiple Drones

We conclude the evaluation by studying MicNest's ability to concurrently localize multiple drones. Two drones, denoted as A and B, are involved in this experiment. Drone A flies along a *squared spiral* at 40 m altitude, while drone B hovers at 20 m altitude. The two drones play different PRN pulses.

Fig. 20 plots the localization results of the two drones during the experiment. MicNest can detect both drones' pulses from the collided signal. The mean localization errors for drones A and B are 1.77 m and 0.38 m, respectively. Below 20 m altitude, this metric for drone A is reduced to 0.17 m, that is, where accuracy is most important for precise landing.

Compared to the single-drone localization performance, MicNest performance is marginally degraded because the presence of multiple drones adds up the degree of background interference. Besides the countermeasures mentioned in Sec. 6.4 to tame background noise, one may also improve the coding scheme of pulse and adopt an adaptive power control scheme [54, 55, 69].

7 DISCUSSION

We elaborate next on the rationale behind some key design decisions in MicNest design and implementation.

Why not using ultrasound? The attenuation of a signal increases as the signal frequency increases. It can be expected that an ultrasound signal that spans the same bandwidth as an acoustic signal experiences much more attenuation. This would inherently limit the operating range of the system.

Is acoustic signal propagation a limitation? Fig. 14 shows that the signal propagation becomes a limiting factor for latency only at high altitudes. Here, the propagation delay can be tolerated to some degree as long as it can be estimated and reported to the navigation system [24]. As the drone approaches the platform, that is, where the highest location update rate is required, the contribution of signal propagation to processing latency becomes increasingly immaterial.

What about the number of microphones? Our implementation of MicNest uses four distributed microphones. We may further improve localization performance by deploying more microphones and, for example, using beamforming techniques to further enhance the signal as received by the landing platform [11].

Why not using frequency-modulated continuous-wave (FMCW) signals? FMCW signals are resistant to Doppler effect [61]. However, it is linear Doppler effect that FMCW can resists to, not non-linear one. In addition, FMCW signals cannot satisfy the practical requirements we outline in the Introduction, such as being friendly to human ear or resistant to impersonation attacks.

What about the performance of multi-drone localization? There exist methods to further improve the performance of multi-drone localization. For example, each drone may adopt an adaptive volume strategy: reducing the volume when the altitude decreases. Therefore, the PRN pulses transmitted by high-altitude drones are less interfered by the pulses of low-altitude drones. Another remedy is to improve the orthogonality of PRN pulses.

Why not accelerating tree search using information from drone IMU? Actually, it is the *radial* drone velocity with respect to the microphones that MFT searches for, not the velocity with respect to the Earth. Before converting the estimated velocity to the radial one, we should know the location of the drone with respect to the microphones. This actually leads to a "chicken-and-egg" problem: tree-search acceleration and drone localization are a prerequisite of each other.

Can we reversely deploy microphones on the drone and a speaker on the ground? This may be feasible, but a practical issue is that the drone size is limited, which sets an upper bound on the inter-microphone distance (aperture), resulting in a lower localization resolution. In addition, the computing resource of drones is generally not sufficient to support real-time pulse detection.

How does multipath effect impact MicNest? In general, MicNest can tolerate multipath effect as long as there is a line-of-sight (LOS) path between the microphones and the speaker. Thanks to the low sound speed, a slight difference in path lengths will lead to a distinguishable time difference of arrival. Therefore, the paths reflected from, for example, the surrounding buildings will not overlap with the LOS path in the time domain. Given that the LOS path is the strongest, we can implicitly determine the LOS path by choosing the most significant correlation peak.

8 CONCLUSION

MicNest enables precise landing of drones using acoustic signals. The key enabling technologies we present are MFT, a novel pulse detector that models the problem as a tree search problem, and its efficient low-latency implementation. These allow MicNest to localize a drone 120 m away with 0.53% relative localization error at 20 Hz location update frequency.

ACKNOWLEDGMENTS

We thank our anonymous shepherd and reviewers for their insightful comments. This work is partially supported by the National Science Fund of China under grant No. U21B2007, Tsinghua University - Meituan Joint Institute for Digital Life, the Swedish Science Foundation (SSF), the Digital Futures programme (project Drone Arena), the Swedish Research Council under grant 2018-05024, and KAW project UPDATE.

[9]Measuring sound speed is simple for MicNest. We can put a speaker in line with two diagonal microphones, and measure the TDoA of PRN pulses. The sound speed can be calculated by dividing the microphone distance by the TDoA.

REFERENCES

[1] 2021. Betaflight flight controller. https://github.com/betaflight/betaflight. (2021).
[2] 2021. Cleanflight Flight Controller. https://github.com/cleanflight/cleanflight. (2021).
[3] 2022. DWM1000 Datasheet. https://www.decawave.com/sites/default/files/resources/DWM1000-Datasheet-V1.6.pdf. (2022). Accessed: 2022-02-06.
[4] Mikhail Afanasov et al. 2019. FlyZone: A testbed for experimenting with aerial drone applications. In *Proceedings of ACM MobiSys*.
[5] Takayuki Akiyama et al. 2017. Time-of-arrival-based indoor smartphone localization using light-synchronized acoustic waves. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences* (2017).
[6] Alphabet. 2022. Project Wing. https://wing.com/. (2022). Accessed: 2022-02-06.
[7] Amazon. 2022. Amazon Prime Air. https://www.amazon.com/Amazon-Prime-Air/b?node=8037720011. (2022). Accessed: 2022-02-06.
[8] PX4 Autopilot. 2022. Landing Phases Flow Diagram. https://docs.px4.io/master/en/advanced_features/precland.html#landing-phases-flow-diagram. (2022). Accessed: 2022-02-06.
[9] PX4 Autopilot. 2022. PX4 Autopilot. https://docs.px4.io/master/en/. (2022). Accessed: 2022-02-06.
[10] Adeola Bannis, Hae Young Noh, and Pei Zhang. 2020. Bleep: motor-enabled audio side-channel for constrained UAVs. In *Proceedings of ACM MobiCom*.
[11] Jacob Benesty, Jingdong Chen, and Yiteng Huang. 2008. *Microphone array signal processing*. Springer Science & Business Media.
[12] Ruth Bentler and Li-Kuei Chiou. 2006. Digital noise reduction: An overview. *Trends in amplification* 10, 2 (2006), 67–82.
[13] Bitcraze. 2021. Lighthouse Positioning. https://www.bitcraze.io/documentation/system/positioning/ligthouse-positioning-system/. (2021).
[14] Bitcraze. 2021. Motion Capture. https://www.bitcraze.io/documentation/system/positioning/mocap-positioning/. (2021).
[15] Bitcraze. 2022. Loco Positioning. https://www.bitcraze.io/documentation/system/positioning/loco-positioning-system/. (2022). Accessed: 2022-02-06.
[16] BOEING. 2021. Statistical Summary of Commercial Jet Airplane Accidents. https://www.boeing.com/resources/boeingdotcom/company/about_bca/pdf/statsum.pdf. (2021). Accessed: 2022-02-06.
[17] Endri Bregu et al. 2016. Reactive Control of Autonomous Drones. In *Proceedings of ACM MobiSys*.
[18] Agus Budiyono. 2012. Principles of GNSS, inertial, and multi-sensor integrated navigation systems. *Industrial Robot: An International Journal* (2012).
[19] U. Cheng et al. 1990. Spread-spectrum code acquisition in the presence of Doppler shift and data modulation. *IEEE Transactions on Communications* (1990).
[20] Hao-Hua Chu et al. 2013. Guoguo: Enabling fine-grained indoor localization via smartphone. In *Proceedings of ACM MobiSys*.
[21] Rémi Coulom. 2006. Efficient selectivity and backup operators in Monte-Carlo tree search. In *Proceedings of Springer International conference on computers and games, Turin, Italy, May 29-31, 2006*. 72–83.
[22] Ashutosh Dhekne et al. 2019. TrackIO: Tracking First Responders Inside-Out. In *Proceedings of USENIX NSDI*.
[23] Dominos. 2016. Pizza-by-drone a reality with world-first customer deliveries in New Zealand. https://www.dominos.com.au/inside-dominos/media/november-2016-pizza-by-drone-a-reality-with-world-first-customer-deliveries-in-new-zealand. (2016). Accessed: 2022-02-06.
[24] Richard C.. Dorf and Robert H Bishop. 2008. *Modern control systems*. Pearson Prentice Hall.
[25] DroneCode. 2022. QGroundControl. http://qgroundcontrol.com/. (2022). Accessed: 2022-02-06.
[26] DroneUp. 2022. DroneUp. https://www.droneup.com/. (2022). Accessed: 2022-02-06.
[27] Ele.me. 2022. Ele.me Cleared To Use Delivery Drones In China. https://www.pymnts.com/news/delivery/2018/eleme-food-delivery-drones-china/. (2022). Accessed: 2022-02-06.
[28] Behrouz Farhang-Boroujeny. 2013. *Adaptive filters: theory and applications*. John Wiley & Sons.
[29] Mark Fiala. 2005. ARTag, a fiducial marker system using digital techniques. In *Proceedings of IEEE CPVR*.
[30] Dario Floreano and Robert J Wood. 2015. Science, technology and the future of small autonomous drones. *Nature* 521, 7553 (2015), 460.
[31] Lewis Girod et al. 2006. A self-calibrating distributed acoustic sensing platform. In *Proceedings of ACM SenSys*.
[32] Paul D Groves. 2011. Shadow matching: A new GNSS positioning technique for urban canyons. *The journal of Navigation* 64, 3 (2011), 417–430.
[33] Hao Han et al. 2016. AMIL: Localizing neighboring mobile devices through a simple gesture. In *IEEE INFOCOM*.
[34] Li-Ta Hsu. 2018. Analysis and modeling GPS NLOS effect in highly urbanized area. *Springer GPS solutions* 22, 1 (2018), 1–12.
[35] Wenchao Huang et al. 2017. Stride-in-the-Loop Relative Positioning Between Users and Dummy Acoustic Speakers. *IEEE JSAC* (2017).
[36] J.H.J. Iinatti. 2000. On the threshold setting principles in code acquisition of DSSS signals. *IEEE JSAC* 18, 1 (2000), 62–72.
[37] Texas Instruments. 2022. Introduction to mmwave Sensing: FMCW Radars. https://training.ti.com/sites/default/files/docs/mmwaveSensing-FMCW-offlineviewing_4.pdf. (2022). Accessed: 2022-02-06.
[38] JD.com. 2016. JD.com's Drone Delivery Program Takes Flight in Rural China. https://jdcorporateblog.com/jd-coms-drone-delivery-program-takes-flight-in-rural-china/. (2016). Accessed: 2022-02-06.
[39] Levente Kocsis and Csaba Szepesvári. 2006. Bandit based monte-carlo planning. In *Proceedings of the European Conference on Machine Learning*.
[40] Maximilian Krogius et al. 2019. Flexible Layouts for Fiducial Tags. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*.
[41] Fan Li et al. 2017. CondioSense: High-quality context-aware service for audio sensing system via active sonar. *Personal and Ubiquitous Computing* (2017).
[42] Sugang Li et al. 2017. Auto++ detecting cars using embedded microphones in real-time. *Proceedings of the ACM IMWUT* (2017).
[43] Qiongzheng Lin et al. 2019. Rebooting Ultrasonic Positioning Systems for Ultrasound-incapable Smart Devices. In *Proceedings of ACM MobiCom*.
[44] Atri Mandal et al. 2005. Beep: 3D indoor positioning using audible sound. In *IEEE Consumer Communications and Networking Conference*.
[45] W. Mao et al. 2017. Indoor Follow Me Drone. In *Proceedings of ACM MobiSys*.
[46] Meituan. 2021. Food Delivery Giant Meituan Unveils Drones for Delivery Service, Offering New User Experience. https://pandaily.com/food-delivery-giant-meituan-unveils-drones-for-delivery-service/. (2021). Accessed: 2022-02-06.
[47] MicNest. 2022. Supplemental Page of MicNest. https://micnest.github.io. (2022). Accessed: 2022-06-21.
[48] Matthew A Nobile. 1985. Identifying prominent discrete tones in noise emissions. *The Journal of the Acoustical Society of America* 78, S1 (1985), S33–S33.
[49] Edwin Olson. 2011. AprilTag: A robust and flexible visual fiducial system. In *Proceedings of the IEEE International Conference on Robotics and Automation, Shanghai, China, May 9-13, 2011*.
[50] OpenCV. 2022. Detection of ArUco Markers. https://docs.opencv.org/4.x/d5/dae/tutorial_aruco_detection.html. (2022). Accessed: 2022-02-06.
[51] World Health Organization. 2018. Environmental noise guidelines for the European region. (2018).
[52] Andrea Patelli et al. 2016. Model-based Real-time Testing of Drone Autopilots. In *Proceedings of DRONET*.
[53] Andrea Simone Pinna et al. 2017. Shooter localization in wireless acoustic sensor networks. In *IEEE Symposium on Computers and Communications*.
[54] Raghuveer M Rao and Sohail A Dianat. 2005. *Basics of code division multiple access (CDMA)*. Vol. 67. SPIE Press.
[55] Matti Rintamäki et al. 2005. *Adaptive power control in CDMA cellular communication systems*. Helsinki University of Technology.
[56] rs2k. 2021. Raceflight flight controller. https://github.com/rs2k/raceflight. (2021).
[57] Beat Schäffer et al. 2021. Drone Noise Emission Characteristics and Noise Effects on Humans—A Systematic Review. *International Journal of Environmental Research and Public Health* (2021).
[58] Sheng Shen et al. 2020. Voice localization using nearby wall reflections. In *Proceedings of ACM MobiCom*.
[59] Mehrdad Soumekh. 1999. *Synthetic aperture radar signal processing*. Wiley.
[60] Michael L Stanchina et al. 2005. The influence of white noise on sleep in subjects exposed to ICU noise. *Elsevier Sleep medicine* (2005).
[61] Andrew G Stove. 1992. Linear FMCW radar techniques. In *IEE Proceedings F-Radar and Signal Processing*, Vol. 139. IET, 343–350.
[62] Richard S Sutton and Andrew G Barto. 2018. *Reinforcement learning: An introduction*. MIT press.
[63] Stephen P. Tarzia et al. 2011. Indoor localization without infrastructure using the acoustic background spectrum. In *Proceedings of ACM MobiSys*.
[64] Aviation Today. 2019. Drone Delivery Crash in Switzerland Raises Safety Concerns As UPS Forms Subsidiary. https://www.aviationtoday.com/2019/08/08/drone-delivery-crash-in-switzerland-raises-safety-concerns/. (2019). Accessed: 2022-02-06.
[65] David Tse and Pramod Viswanath. 2005. *Fundamentals of wireless communication*. Cambridge university press.
[66] Yu-Chih Tung and Kang G. Shin. 2015. EchoTag: Accurate infrastructure-free indoor location tagging with smartphones. In *Proceedings of ACM MobiCom*.
[67] UPS. 2021. UPS operates first ever U.S. drone COVID-19 vaccine delivery. https://about.ups.com/us/en/our-stories/innovation-driven/drone-covid-vaccine-deliveries.html. (2021). Accessed: 2022-02-06.
[68] Saeed V Vaseghi. 2008. *Advanced digital signal processing and noise reduction*. John Wiley & Sons.
[69] Andrew J Viterbi. 1995. *CDMA: principles of spread spectrum communication*. Addison Wesley Longman Publishing Co., Inc.
[70] D. Wagner et al. 2008. Pose tracking from natural features on mobile phones. In *Proceedings of International Symposium on Mixed and Augmented Reality*.
[71] Walmart. 2021. Walmart Invests in DroneUp, the Nationwide On-Demand Drone Delivery Provider. https://corporate.walmart.com/newsroom/2021/06/17/walmart-invests-in-droneup-the-nationwide-on-demand-drone-delivery-provider.

(2021). Accessed: 2022-02-06.
[72] Anran Wang et al. 2019. Contactless infant monitoring using white noise. In *Proceedings of ACM MobiCom*.
[73] John Wang and Edwin Olson. 2016. April*f*Tag 2: Efficient and robust fiducial detection. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*.
[74] Mei Wang et al. 2021. MAVL: Multiresolution Analysis of Voice Localization. In *Proceedings of USENIX NSDI*.
[75] Weiguo Wang et al. 2020. Symphony: localizing multiple acoustic sources with a single microphone array. In *Proceedings of ACM SenSys*.
[76] Wikipedia. 2022. Real-time kinematic positioning. https://en.wikipedia.org/wiki/Real-time_kinematic_positioning. (2022). Accessed: 2022-02-06.
[77] Wikipedia. 2022. Speed of sound. https://en.wikipedia.org/wiki/Speed_of_sound. (2022). Accessed: 2022-03-25.
[78] Jingbin Zhang et al. 2007. Thunder: towards practical, zero cost acoustic localization for outdoor wireless sensor networks. *ACM SIGMOBILE Mobile Computing and Communications Review* (2007).
[79] Zipline. 2022. In the Ari With Zipline's Medical Delivery Drones. https://spectrum.ieee.org/in-the-air-with-ziplines-medical-delivery-drones. (2022). Accessed: 2022-03-25.