Autonomous Flying WiFi Access Point

Gareth J. Nunns[†], Yu-Jia Chen[‡], Deng-Kai Chang[‡], Kai-Min Liao[‡], Fung Po Tso[†*], Lin Cui[§]
[†]Department of Computer Science, Loughborough University, UK
[‡]Department of Communication Engineering, National Central University, Taiwan
[§]Department of Computer Science, Jinan University, Guanzhou, China
gareth@garethnunns.com; yjchen@ce.ncu.edu.tw; {wl01420395, lkiamin}@gmail.com; p.tso@lboro.ac.uk; tcuilin@jnu.edu.cn;

Abstract—Unmanned aerial vehicles (UAVs), aka drones, are widely used civil and commercial applications. A promising one is to use the drones as relying nodes to extend the wireless coverage. However, existing solutions only focus on deploying them to predefined locations. After that, they either remain stationary or only move in predefined trajectories throughout the whole deployment. In the open outdoor scenarios such as search and rescue or large music events, etc., users can move and cluster dynamically. As a result, network demand will change constantly over time and hence will require the drones to adapt dynamically.

In this paper, we present a proof of concept implementation of an UAV access point (AP) which can dynamically reposition itself depends on the users movement on the ground. Our solution is to continuously keeping track of the received signal strength from the user devices for estimating the distance between users devices and the drone, followed by trilateration to localise them. This process is challenging because our on-site measurements show that the heterogeneity of user devices means that change of their signal strengths reacts very differently to the change of distance to the drone AP. Our initial results demonstrate that our drone is able to effectively localise users and autonomously moving to a position closer to them.

Index Terms—UAV access point, Flying access point, Drone based user localisation

I. Introduction

With continuous cost reduction and device miniaturisation the unmanned aerial vehicles (UAVs), also known as drones, have seen proliferated adoption for civil and commercial applications. Of which an increasingly more popular one is rapid deployment as relays to meet the networking needs in places where network infrastructure [1] [2] [3] [4] [5] [6]. Researchers have investigated a wide range of topics including architecture design [1] [3], placement [6], coverage [5], channel optimisation [4] and routing [7] [8]. However, existing works simply assume networking demands are known and hence once UAVs are deployed they either remain stationary or only move on predefined paths regardless the movement of the users on the ground.

We argue that the user mobility can greatly challenge the the quality of wireless connectivity as when users move both the demand and networking environment will change [9]. In fact, our real-world measurements demonstrate great radio heterogeneity among user equipments (see Section IV): different devices (i.e. a mobile phone and a laptop) can emit distinctively different strength radio signal and weaker signal emitting devices (e.g. mobile phones) are more prone to the

change of distance. This means in order to maintain overall best signal to all connected mobile devices, the drone should dynamically reposition itself according to the latest snapshot of users' locations. Repositioning of drone requires a mechanism to determine users current locations and to which direction that they are moving. This can be easily done if the drone has multiple uni-directional antennas that periodically send/receive probing signals [10]. Unfortunately, this is impractical in reality as doing so will not only increase the cost and sizes of the drone APs but also shorten their battery lives.

In this paper, we present our proof-of-concept implementation of a drone based AP that can autonomously readjust its position for providing best connectivity to users below. We have built a drone using off-the-shelf components with a Raspberry Pi[1], a single board computer, as a WiFi access point (AP). We have employ a novel technique to localise user devices. This include first flying a small square trajectory and then use the reference points gathered to trilaterate user locations. We have tested our algorithms and initial results show that drone can effectively localise users and autonomously reposition itself closer to them.

The remainders of this paper are organised as follows. Section II presents related work in drone-based internet provisioning and techniques for user localisation. Section III details our drone AP building experience, followed by Section IV which describes the experimental and analytical development of device localisation. Section V presents our initial implementation and preliminary evaluation result. Section VI discusses our future directions and concludes this work.

II. Related Work

Using drones for wireless communication has been a subject of significant interest recently. In this section, we present related literature and highlight our contributions to the state of the art.

UAV-enabled communication is one of the key emerging applications [3] [1]. However, most of existing works mainly focus on millimeter wave (mmWave) bands, for long range communication. On the contrary, we use off-the-shelf WiFi for its prevalence, technological matureness, and low-cost as well as reasonable distance of coverage.

When a drone is needed, it's deployment location can have vital impact on the effectiveness. A multi-drone placement

---

[1]https://www.raspberrypi.org/

problem is investigated in [6] with an aim to maximise the minimum throughput in the downlink communication by jointly optimising the traffic scheduling and the drones simple circular trajectory and power control. The problem is formulated as a mixed integer non-convex optimisation problem that is solved using an iterative heuristic. [5] studies a network coverage recovery problem which consists of two parts. First, drones navigate in predefined regions in zigzag patterns to scan for the presence of existing (damaged) networking infrastructure nodes. Then the drones send probing packets to discover partial network topology and routes. In both cases, fixed trajectories are defined and maintained throughout. To contrast, in our work new flying trajectories are determined by the drone autonomously using received signal strength.

Author in envisions a use case in which Besides UAV-aided communication, UAVs can be used in fog computing infrastructure [11] [12]. More specifically, [12] is the first technical work to address this and tackles the problem of optimising the bit allocation for communication in uplink and downlink and for computing at the cloudlet, the UAV's trajectory, with the goal of minimising the mobile energy consumption. In comparison, our drone AP is also a general computer, making it ideal for computation offloading and therefore will complement this branch of work.

Radio channel optimisation are investigated in [13] [4] [9]. [4] performs measurements and simulations to study the impact of interference and path loss when transmitting data to and from the UAV in LTE networks. Their results show that while a single drone can improve signal to interference ratio but increasing the density of drone deployment will hurt it. These findings are confirmed in [13]. Comparing physical and MAC layers optimisation, we fully focus on application layer.

There is also a body of work on improving routing [7] [8]. [7], for example, investigates a layered UAV swarm network architecture and proposes a low latency routing algorithm (LLRA) for this network architecture. Similarly [14] surveys and analyses the suitability of traditional mobile adhoc network protocols for flying adhoc networks. Our work will complement on these works.

Another domain of work focusing on localising users in order to best position drones with respect to the users' locations. [15] uses WiFi's probe requests, RSSI and random forest algorithm to localise users but its granularity of accuracy is coarse. One possible technique is triangulation [16] but this requires knowledge of the angle from the drone to the user. In comparison we have applied trilateration [17] as this technique is able to locate a target by using the distance information between the target and several reference points.

III. Building The Drone

We have built a drone, as shown in Fig. 1, with readily available commercial components including a Raspberry Pi as the AP, listed as follows:

1) Raspberry Pi 3 Model B in clear case.
2) Crius CN-06 v2.0 GPS module
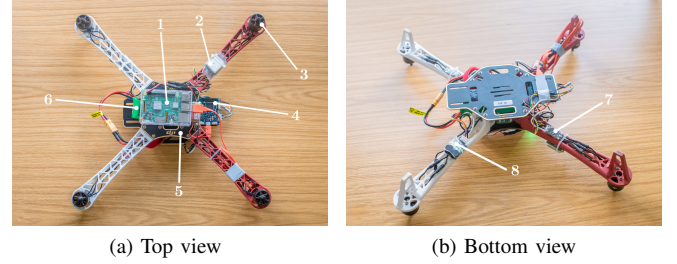3) DJI 2312E motors

(a) Top view   (b) Bottom view

Fig. 1: Completed drone

Fig. 2: General setup used for measuring RSS from the Pi

4) CRIUS All in One Pro Flight Controller (FC) v2.0
5) DJI Flame Wheel F450 ARF frame
6) Multistar 3S 3000mAh batteries
7) Electronic Speed Controller (ESC)
8) HobbyKing Lipoly Low Voltage Alarm (2s-4s)

IV. Localising User Devices

A. Test Setup

Fig. 2 shows the concept of the test setup: measurements on the floor with one metre increments, where the drone, with the Pi attached to it, is placed at one end and the laptop and/or phone moving away from the drone, along the measurements. The maximum distance was 40m as this was considered to be furthest away the drone would optimistically be from the user, in this proof of concept. In order for the tests to be consistent, we used the same laptop and smartphone throughout all experiments.

All of the tests were performed at a remote, empty, outdoor car park with a concrete/gravel surface, walled by hedges. This was chosen so that markings could easily be made on the ground and there would be little reflection from the area. Importantly, the test was not performed indoors, as that is not the intended purpose for this project. The drone remained directly on the floor in the same place (facing forward for all range tests). Similarly, all testing was recorded on the same day and where possible recording comparable sets successively (e.g. when rotating the drone at a fixed distance). All experiments were repeated 3 times. The logging of the data was all performed internally on the Raspberry Pi. The Pi was set to a static WiFi channel

All devices were left connected to the drone for at least a minute to ensure any initial handshaking protocols were avoided and did not affect the results. We have also attempted to minimise the human element: the devices were all held in a similar fashion and stepping between the markers was kept as uniform as practicable.

B. Range Tests

We first set to answer this question: *is there a discernible difference in the RSS when the user moves away from the*

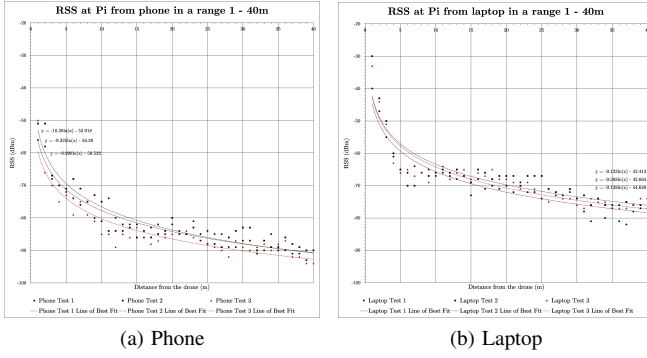(a) Phone   (b) Laptop

Fig. 3: The effect on the RSS received by the Pi, when varying the distance of the user devices

drone within a reasonable range of values? Every five seconds a metre step was taken along the line. The laptop was disconnected from the Pi's WiFi whilst the phone tests were executed.

Both Fig. 3a & and Fig. 3b show the strong correlation between distance and RSS up until 5 - 10m away from the drone, however beyond that there is a lack of fidelity - this means that ideally the drone needs to be as close as possible to devices to gain greater accuracy above the significant noise. Hence, the lower the height of the drone the better, but balanced with safety, a hovering height of approximately 5 - 6m would minimise error. Notably they are not at the same level - e.g. when the drone receives an RSS value of -72 dBm from the phone it is at approximately 5m away from the drone; conversely receiving this level from the laptop, it is more likely to be 20 - 30m away. This difference is most likely caused by a stronger transmitter in the laptop.

C. Directional Tests

Next we want to see if the WiFi antenna on the Raspberry Pi, when attached to the drone, was at all directional. If there is a strong correlation then the drone could simply rotate at a number of points, providing an angle at each and triangulation could be used to localise the user.

In a similar vein to the previous tests, the directional tests were setup in the same manner. The routine for these tests was to leave the laptop and phone connected whilst rotating the drone 90° clockwise every 2:05 minutes - in order to log 50 points of data every 2.5 seconds for each orientation. The test was performed with the laptop and phone 5m, 10m and 20m away from the drone as these would be common distances in flight. The tests were performed in the order: *forward, left, backward, right*, where left means the phone is to the left and vice-versa for the other directions.

The box plots shown in Fig. 4a – Fig. 4h show the results from these tests - with the box representing the 25th - 75th percentile data points, the central line as the median, the whiskers indicating the minimum and maximum values and the marker showing the mean of the values. Fig. 4a shows an ideal case of rotating the drone, where it should be possible

to discern the direction to the user from recording a few data points in each orientation. Unfortunately, the findings from the laptop at the same range, Fig. 4b, are slightly less conclusive as the mean values are less than 4dBm apart. There is a notable level of noise in the readings, meaning to draw meaningful conclusions a number of data points should be averaged. The reason the best reception is found towards the back of the drone is possibly due to the location of the WiFi antenna on the Pi board, hence receiving very little electromagnetic inference from other boards. In the context of the drone the antenna has few obstructions when it is directly behind and this is best reflected in data from when the laptop was 10m away, Fig. 4d – the greater the angle from the back of the drone, the worse the signal, but this is inconsistent with the signal received from the laptop at 20m, Fig. 4f.

Comparing the data at 10m against 20m, Fig. 4c & Fig. 4d and 4e & 4f respectively, there is a greater overlap in data but there is still a distinctly higher trend for the backward values. However, in Fig. 4f, the forward and backward data is quite close - meaning that it would not be reliable to base the drone's movements solely from these readings. Using the data from the aforementioned tests, Fig. 4g and Fig. 4h compare the forward facing values from the various distances for the phone and laptop. This determines the distance between the points used to trilaterate the user. Based on Fig. 4g, within a horizontal plane it may be possible to use a distance between the points of 5m, however due to the sizeable overlap, for greater certainty and considering the larger distances when the drone is airborne, a distance of 10m or more would mean there is likely to be a more identifiable difference in the readings.

D. Drone Operation States

Once the drone is flying, localisation can essentially be reduced down to a maximisation problem. For example, Fig. 5 shows how the RSS is perceived by the drone from two devices at two separate locations, where one device has a more powerful transmitter than the other, *e.g. a laptop and a phone*. Given this plot, it is trivial to locate the optimum position for the drone to provide the best signal to the two devices. This is the entire search space at only a single point in time, however the drone can only sample at single (x, y) coordinate. Furthermore, the devices can move, the drone's WiFi antenna is slightly directional and the drone's position is difficult to exactly locate in all six degrees of freedom. With these points in mind, the following algorithm has been designed.

The drone takes off and looks for the best direction to start searching, then performs a square, starting in that direction. Once the square is completed, the drone trilaterates the users and moves to the position which it decides will give the best signal to the users.

We have also defined the following states for our drone AP. *TAKEOFF*: The drone's initial state. Gets the drone up to its minimum height of 6m - this is based from the data shown in Fig. 3a & Fig. 3b: the best WiFi distance fidelity is within the 10m range, however for safety, a height of 6m is a balance. *LANDING*: Lands the drone in its takeoff position and then

(a) Phone RSS at 5m (b) Laptop RSS at 5m (c) Phone RSS at 10m (d) Laptop RSS at 10m

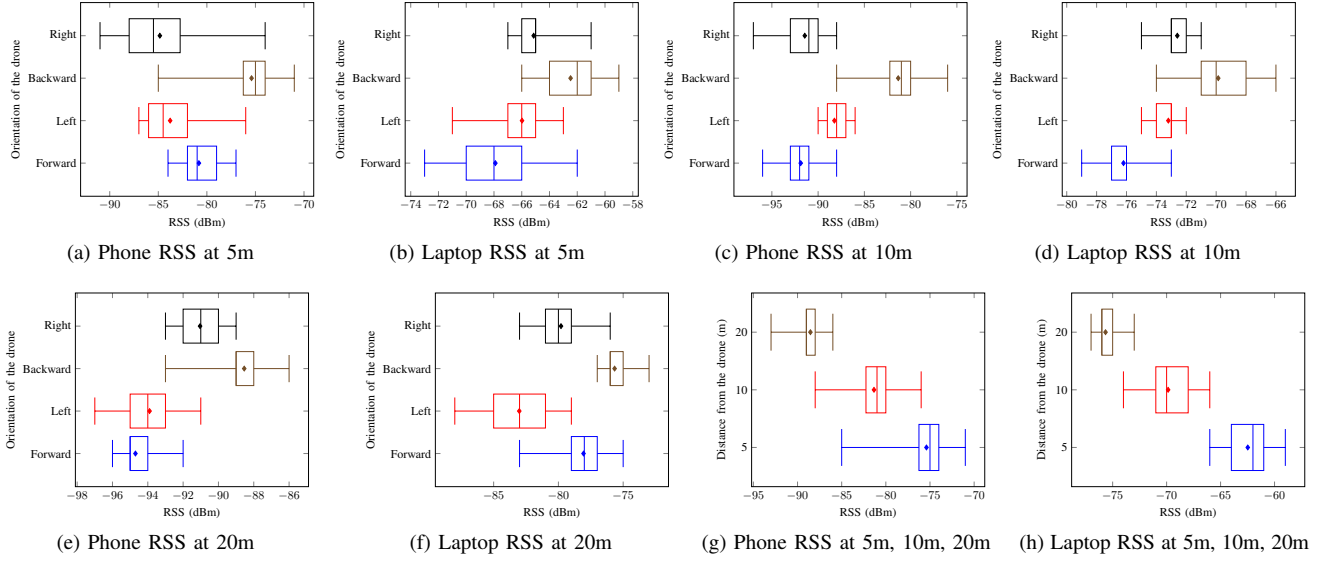(e) Phone RSS at 20m (f) Laptop RSS at 20m (g) Phone RSS at 5m, 10m, 20m (h) Laptop RSS at 5m, 10m, 20m

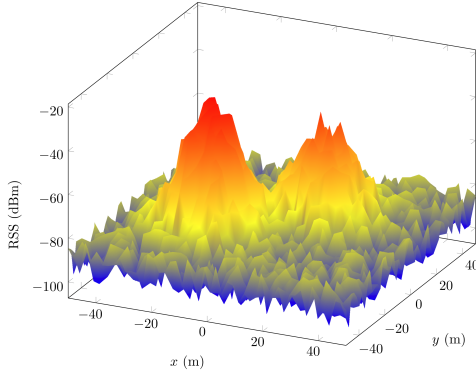Fig. 4: The effect of varying the distance on the RSS received by the Pi

Fig. 5: Representation of the RSS from two devices, with noise, when the drone moves in the horizontal plane

powers it down. *STARTSEARCH*: Rotates the drone and waits in each orientation for three seconds in order to log three RSS readings - this is in an effort to assess the direction where the most users are, utilising the slight directionality of the antenna on the Pi, shown in Fig. 4a – Fig. 4f. *SEARCH*: The search consists of performing a 10m square. This was chosen because there is approximately only 10m of fidelity in RSS values, shown in Fig. 4g & Fig. 4h. From the data collected whilst performing the square, the drone will trilaterate the position of each user using the lateration formula, equation 4. *OPTIMAL*: The drone moves to where it believes is the best position to provide the best coverage to the devices which have recently connected.

E. Finding the Optimal Location

As there is not a constant correlation between RSS and distance, shown by the difference in magnitudes between Fig. 3a & Fig. 3b, it is necessary to approximate the values for A, the RSS from the device at one metre, and n, the path loss exponent, in equation 1 [18] so that the only unknown is r

Line of Best Fit	Equation	n (3 d.p.)
Phone Test 1	$y = -23.670 \log_{10} x - 52.918$	2.367
Phone Test 2	$y = -21.472 \log_{10} x - 56.28$	2.147
Phone Test 3	$y = -20.719 \log_{10} x - 59.532$	2.072
Laptop Test 1	$y = -21.006 \log_{10} x - 44.648$	2.101
Laptop Test 2	$y = -21.571 \log_{10} x - 42.413$	2.157
Laptop Test 3	$y = -21.041 \log_{10} x - 42.664$	2.104
	Average	**2.157**

TABLE I: Lines of best fit from Fig. 3a & Fig. 3b in log_{10} *(where S is the RSS from the device)*, which is the distance from drone:

$$r = 10^{\frac{S-A}{-10n}} \quad (1)$$

Rearranging equation 1 to make A the subject gives equation 2:

$$A = S + 10n \log_{10} r \quad (2)$$

It will be assumed that the lowest absolute RSS values received during the duration of the flight are logged when the drone is above the user and hence both S and r will be known: the drone hovers at 6m and the user's device is assumed to be at approximately 1m from the ground, so $r = 6 - 1 = 5$. To reduce the effect of noise, shown by the box plots in Fig. 4a - Fig. 4f, three values will be averaged.

The value for n can be approximated from the test data shown in Fig. 3a & Fig. 3b.Converting these from expressions in the form $y = m \ln x + c$ to those with base 10 *(to match equation 2)* using the method $y = \frac{m}{\log_{10} e} log_{10} x + c$ produces table I:

From this, the value for n is approximated to 2.2, giving allowances for anything blocking line of sight and other extraneous factors. Filling in the previous unknowns in equation 2 gives equation 3, where S is the average of the three best RSS recordings:

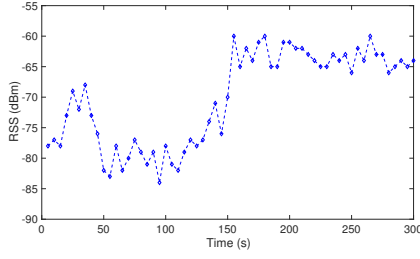$$A = S + 10 \cdot 2.2 \cdot \log_{10} 5 \quad (3)$$

Fig. 6: The drone AP is able to localise in 150s and maintain strong RSS for the smartphone

It is then a matter of applying equation 4 [17] and equation 5 to all devices:

$$\begin{bmatrix} r_1^2 - r_2^2 + x_2^2 + y_2^2 - x_1^2 - y_1^2 \\ r_1^2 - r_3^2 + x_3^2 + y_3^2 - x_1^2 - y_1^2 \\ \cdots \\ r_1^2 - r_N^2 + x_N^2 + y_N^2 - x_1^2 - y_1^2 \end{bmatrix} =$$

$$\begin{bmatrix} 2 \cdot (x_2 - x_1) & 2 \cdot (y_2 - y_1) \\ 2 \cdot (x_3 - x_1) & 2 \cdot (y_3 - y_1) \\ \cdots & \cdots \\ 2 \cdot (x_N - x_1) & 2 \cdot (y_N - y_1) \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \tag{4}$$

$$D = C \begin{bmatrix} x \\ y \end{bmatrix} \Leftrightarrow \begin{bmatrix} x \\ y \end{bmatrix} = (C^T \cdot C)^{-1} C^T D \tag{5}$$

C & D correspond to the matrices in equation 5.

V. IMPLEMENTATION AND EVALUATION

We have implemented the key algorithms of our autonomous flying AP on a Raspberry Pi, which controls the flight controller, using Python. We have performed a number of tests in one of the university's sports pitches, which are large and clear of people and obstacles. At the beginning of each test we connected a laptop and a smartphone to the AP, both were 10m away to the front of it.

Our preliminary testing results showed that the drone was capable of flight. The drone performed an approximate square whilst logging the RSS values once per second, from which it determined an estimate for the position of the users and moved itself appropriately, as demonstrated in Fig. 6. Unfortunately the precision of GPS positioning meant that the drone was not able to move to desired optimal location with great precision. Improving this remains a part of our future work.

VI. FUTURE DIRECTIONS AND CONCLUSION

A. Radio Frequency Source Localisation

We have demonstrated that it is challenging to localise the radio frequency (RF) source accurately due to the noisy nature of the RSS samples. In the following, we briefly introduce several methods that can be applied to improve design of flight trajectory for localising RF sources.

Markov Decision Processes (MDPs) [19]: MDP is a typical model for decision making in which an agent perceive the state in the environment and take actions accordingly. To
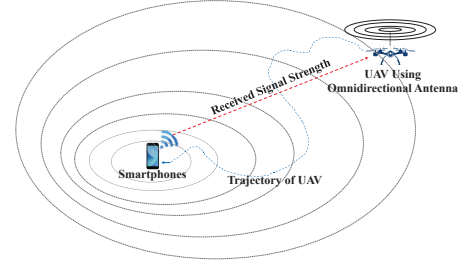

Fig. 7: The drone navigates to the RF source based on the RSS value.

Method	Flight Time (s)	RMS (m)
POMDP [19]	724	12.4
Q-learning [21]	200	≤ 10

TABLE II: **Performance Comparison of Different Localisation Methods**

estimate the state robustly, filtering modules can be used to remove the RSS outliers before making trajectory decisions. For example, the Baysian filtering can provide the probability information about the system state based on the last action and current observation. For some cases requiring real-time planning in uncertain environments, the partially observable Markov decision process (POMDP) can be used to solve these problems. However, it is known to be intractable for computing the optimal policy of a large-scale POMDP with nonlinear state space model.

Maximum Likelihood (ML) [20]: The aim of ML is to estimate the parameters of a statistical model by maximising a likelihood function. In this method, the joint likelihood function can be designed to estimate whether the target is detected by the drone or not. The applicability of ML is limited as it requires knowledge of the statistics of both the signal and noise.

Q-learning [21]: Q-learning is a type of machine learning method in which learning is achieved through the interaction with the environment. It can be viewed as a model-free solution to an MDP with unknown state transition model. In Q-learning, actions, states, and rewards are trained to maximise the long-term expected reward. Similar to applying Q-learning for RF source localisation in the indoor environment [21], the Q-learning algorithm can be used to navigate the drone to the RF source even in a GPS-denied environment. As shown in figure 7, the drone can autonomously determine flight direction based on the stored Q-table when receiving the RSS data. Table II compares the performances of flight time and root mean square (RMS) for drone to localise the RF source.

B. Drone-based Edge Caching

Further to being an AP, the drone can also be an edge node to cache the popular content in advance for improving network service performance. The wireless coverage can be dynamically expanded with the help of aerial-to-aerial (A2A) and aerial-to-ground (A2G) communications [22].
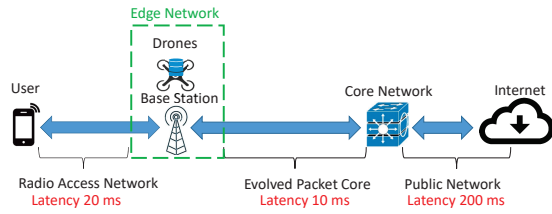
Fig. 8: The latency components with edge caching [24].

The optimal contents need to be cached at drones based on the users' cell association and content request distributions [23]. The caching strategy can be generally classified into two categories: reactive caching and proactive caching. In reactive caching, only the previously used data are stored. On the other hand, proactive caching predicts what data will be needed in the next period of time and stores these data in advance.

To provide low-latency vehicular transmissions, more research efforts are needed to evaluate the latency performance in the air-ground integrated vehicular network with edge caching. Figure 8 shows the latency components with mobile edge caching [24]. By caching the data close to the end user, the data can be delivered without routing to the central core network. With cache's help, the latency can decrease from hundreds of miliseconds to dozens of miliseconds. To further achieve performance guarantees, network theoretic approaches such as network calculus (NC) [25] can be applied to examine the delay bounds with the required violation probability.

C. Conclusion

In this paper we have presented our design and implementation of an autonomous flying WiFi AP. While AP is used as an example some potential applications include edge computing system that can be quickly deployed in ad-hoc fashion. Our initial evaluation has demonstrated that our drone can dynamically re-position itself to give best possible signal strength to user devices. We have also discussed two future directions of this work including better trajectory design for device localisation and using the AP for edge caching.

Another strand of the future work includes deploying multiple drones and develop algorithms and protocols for them to interact with one another.

VII. ACKNOWLEDGEMENT

REFERENCES

[1] Y. Zeng, R. Zhang, and T. J. Lim, "Wireless communications with unmanned aerial vehicles: Opportunities and challenges," *IEEE Communications Magazine*, vol. 54, no. 5, pp. 36–42, 2016.

[2] M. Mozaffari, W. Saad, M. Bennis, Y.-H. Nam, and M. Debbah, "A tutorial on uavs for wireless networks: Applications, challenges, and open problems," *arXiv preprint arXiv:1803.00680*, 2018.

[3] T. Andre, K. A. Hummel, A. P. Schoellig, E. Yanmaz, M. Asadpour, C. Bettstetter, P. Grippa, H. Hellwagner, S. Sand, and S. Zhang, "Application-driven design of aerial communication networks," *IEEE Communications Magazine*, vol. 52, no. 5, pp. 129–137, 2014.

[4] B. V. Der Bergh, A. Chiumento, and S. Pollin, "Lte in the sky: trading off propagation benefits with interference costs for aerial nodes," *IEEE Communications Magazine*, vol. 54, no. 5, pp. 44–50, May 2016.

[5] S.-Y. Park, C. S. Shin, D. Jeong, and H. Lee, "DroneNetX: Network Reconstruction through Connectivity Probing and Relay Deployment by Multiple UAVs in Ad-Hoc Networks," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 11, pp. 1–1, 2018.

[6] Q. Wu, Y. Zeng, and R. Zhang, "Joint trajectory and communication design for multi-uav enabled wireless networks," *IEEE Transactions on Wireless Communications*, vol. 17, no. 3, pp. 2109–2121, 2018.

[7] Q. Zhang, M. Jiang, Z. Feng, W. Li, W. Zhang, and M. Pan, "Iot enabled uav: Network architecture and routing algorithm," *IEEE Internet of Things Journal*, pp. 1–1, 2019.

[8] M. H. Tareque, M. S. Hossain, and M. Atiquzzaman, "On the routing in flying ad hoc networks," in *2015 Federated Conference on Computer Science and Information Systems (FedCSIS)*. IEEE, 2015, pp. 1–9.

[9] E. Yanmaz, R. Kuschnig, and C. Bettstetter, "Achieving air-ground communications in 802.11 networks with three-dimensional aerial mobility," *Proceedings - IEEE INFOCOM*, pp. 120–124, 2013.

[10] Y. Gu, M. Zhou, S. Fu, and Y. Wan, "Airborne WiFi networks through directional antennae: An experimental study," *2015 IEEE Wireless Communications and Networking Conference, WCNC 2015*, pp. 1314–1319, 2015.

[11] S. W. Loke, "The internet of flying-things: Opportunities and challenges with airborne fog computing and mobile cloud in the clouds," *arXiv preprint arXiv:1507.04492*, 2015.

[12] S. Jeong, O. Simeone, and J. Kang, "Mobile edge computing via a uav-mounted cloudlet: Optimization of bit allocation and path planning," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 3, pp. 2049–2063, 2018.

[13] R. Amorim, H. Nguyen, P. Mogensen, I. Z. Kovács, J. Wigard, and T. B. Sørensen, "Radio channel modeling for uav communication over cellular networks," *IEEE Wireless Communications Letters*, vol. 6, no. 4, pp. 514–517, 2017.

[14] M. H. Tareque, M. S. Hossain, and M. Atiquzzaman, "On the routing in flying ad hoc networks," in *2015 Federated Conference on Computer Science and Information Systems (FedCSIS)*, Sep. 2015, pp. 1–9.

[15] V. Acuna, A. Kumbhar, E. Vattapparamban, F. Rajabli, and I. Güvenç, "Localization of WiFi devices using probe requests captured at unmanned aerial vehicles," *IEEE Wireless Communications and Networking Conference, WCNC*, 2017.

[16] A. Rozyyev, H. Hasbullah, and F. Subhan, "Indoor Child Tracking in Wireless Sensor Network using Fuzzy Logic Technique," pp. 81–92, 2011.

[17] T. Dag and T. Arsan, "Received signal strength based least squares lateration algorithm for indoor localization," *Computers and Electrical Engineering*, vol. 66, pp. 114–126, 2017.

[18] J. Xu, W. Liu, F. Lang, Y. Zhang, and C. Wang, "Distance Measurement Model Based on RSSI in WSN," *Wireless Sensor Network*, vol. 02, no. 08, pp. 606–611, 2010.

[19] H. Van Nguyen, M. Chesser, L. P. Koh, S. H. Rezatofighi, and D. C. Ranasinghe, "Trackerbots: Autonomous uav for real-time localization and tracking of multiple radio-tagged animals," *arXiv preprint arXiv:1712.01491*, 2017.

[20] F. Koohifar, A. Kumbhar, and I. Guvenc, "Receding horizon multi-uav cooperative tracking of moving rf source," *IEEE Communications Letters*, vol. 21, no. 6, pp. 1433–1436, 2017.

[21] B. S. Ciftler, A. Tuncer, and I. Guvenc, "Indoor uav navigation to a rayleigh fading source using q-learning," *arXiv preprint arXiv:1705.10375*, 2017.

[22] I. Parvez, A. Rahmati, I. Guvenc, A. I. Sarwat, and H. Dai, "A survey on low latency towards 5g: Ran, core network and caching solutions," *IEEE Communications Surveys Tutorials*, pp. 3098–3130, 2018.

[23] H. Fang, S. S. Dayapule, F. Yao, M. Doroslovaki, and G. Venkataramani, "Prefetch-guard: Leveraging hardware prefetches to defend against cache timing channels," in *IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, 2018, pp. 187–190.

[24] Y. Tan, C. Han, M. Luo, X. Zhou, and X. Zhang, "Radio network-aware edge caching for video delivery in mec-enabled cellular networks," in *IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*, 2018, pp. 179–184.

[25] Y. Jiang and Y. Liu, *Stochastic network calculus*. Springer, 2008.