2024 IEEE International Conference on Big Data (BigData) | 979-8-3503-6248-0/24/$31.00 ©2024 IEEE | DOI: 10.1109/BigData62323.2024.10825924

Fly-ABAC: Attribute Based Access Control for the Navigation of Unmanned Aerial Vehicles

Wynter Japp*, Victoria Lee†, Sai Avinash Vagicherla‡, and Carlos Rubio-Medrano‡
* University of Louisiana at Lafayette, Lafayette, LA, USA
wynter.japp1@louisiana.edu
† University of California, Berkeley, Berkeley, CA, USA
vlee@berkeley.edu
‡ Texas A&M University - Corpus Christi, Corpus Christi, TX, USA
svagicherla@islander@tamucc.edu / carlos.rubiomedrano@tamucc.edu

***Abstract*—With the increasing deployment of UAVs across more and more domains such as surveillance, agriculture monitoring, and commercial delivery, there is an increasing need for robust systems that respect both federal regulations and private property owners' constraints. To address these concerns, this paper presents a novel application of Attribute-Based Access Control (ABAC) for the navigation of Unmanned Aerial Vehicles (UAVs) to ensure compliance with both federal and public property owner preferences. Our research demonstrates that an ABAC system we call Fly-ABAC can effectively manage complex property restrictions by modeling positive and negative attributes as well as exclusion zones. Through simulations, our system has shown the ability to generate paths that avoid unauthorized zones and comply with specified access requirements, ensuring a high level of precision and customization in property access control.**

By leveraging the granularity of ABAC, Fly-ABAC can accommodate diverse property preferences and is easy to adjust to different scenarios, making it suitable for a wide range of applications, including enforcing regulations for flying over government buildings, airports, emergency operations, and private properties. Our findings indicate that this system not only enhances security and compliance but also provides property owners with a powerful tool to enforce their access policies. The implications of this research extend to improving the safety and operational efficiency of UAVs in complex environments, paving the way for the broader adoption of UAV for both commercial and private uses.

KEYWORDS

Access Control; Attribute-Based Access Control; Unmanned Aerial Vehicles; Permission.

I. Introduction

Unmanned Aerial Vehicles (UAVs), commonly known as drones, have embraced various roles in our modern world. In agriculture, they are used for tasks such as checking crop health or spraying and seeding crops [9]. They have found particular use in various roles in commercial jobs, such as in building inspection [16], as they can quickly reach high places, largely removing the possibility of human injury from such tasks. They are also well suited for other precarious roles such as search and rescue missions, as they can quickly scan large areas [10]. However, as drones develop into other commercial roles such as package delivery [3], they are increasingly likely to need to operate in a broader range of restricted, private, or public areas where they should be required to not only work within the restrictions of the Federal Aviation Administration (FAA) but also adhere to the limitations set by private property owners [6] [15].

Fig. 1. A Concept Image of a Package Delivery Drone. Designed by Freepik [8].

As many drones are now equipped with sensors such as LiDAR, audio sensors, and camera systems [11], it is imperative to ensure the privacy of property owners, whether they are government or civilian, is maintained as the drone travels both on its way to its destination and back to its origin. Complicating the matter is that there is no proven method for balancing security and privacy when using automated drones, leading to their potential abuse [4]. The most straightforward solution to protect the privacy of property owners is to mandate that drone paths cannot cross into the airspace of a property unless the owner explicitly allows it. To enforce these constraints, we propose a system where property owners can set their preferences regarding drones entering their property, which are enforced before creating the drone's path.

There exist measures that have been created in order to ensure that drones obey both federal regulation and the preferences of property owners such as B4UFLY [5], which was created to inform recreational drone users where they could fly their drones in accordance to federal regulations, and Privaros [14], which provides a few different ways of accessing properties that attempt to maintain privacy of the property owner giving them the option to blur images taken on their property by the drone or keep drones relegated to certain areas of the property. However, while both of these measures handle privacy to a certain extent, neither method covers the hypothetical case in which property owners refuse to allow drones to enter their property, as B4UFLY handles government regulations only and Privaros only gives property owners ways to restrict how drones interact with a property, not prevent them from entering entirely. Solving this issue is critical, as the consequences could result in a scenario where a non-consenting property owner could take action to harm the drone. Conversely, a drone that trespasses on a property without consent could damage the property, resulting in legal issues or any number of unfortunate scenarios that could hinder the advancement and adoption of automated drone technologies. Thus, we endeavor to answer this research question:

RQ1: Can an access control system be used to accurately enforce the relevant diverse requirements of property owners on a drone?

There are many access control systems that offer different levels of control and granularity, including Role-Based Access Control (RBAC) [12], Relationship-Based Access Control (ReBAC) [12], and Attribute-Based Access Control (ABAC) [12]. RBAC relies on roles which are attached to particular permissions. When roles are assigned to users, the users gain the permissions associated with those role. Relationship-Based Access Control assigns permissions based on the relationships between users and their resources. For example, if a user creates a resource, that user will have permission to access and edit that resource. Lastly, ABAC uses the combined attributes of users, resources, and the environment to determine whether a permission can be granted. Of all these systems, ABAC is the most granular, offering the most varied level of control over given permissions. Since the issue we face requires a high level of granularity, with potentially many users or drones needing to calculate access to many different zones, each with their own property owners who might set very specific requirements for entry, we only focus on exploring ABAC as the access control system for the navigation of UAVs in this paper. Thus, our research question is refined to:

RQ1: Can an ABAC system be used to accurately enforce the relevant diverse requirements of property owners on a drone?

Besides providing answers to this question, this paper provides the following contributions:

1) We introduce Fly-ABAC, the application of ABAC for the precise and customizable navigation of UAVs, addressing a critical need for property owner preferences and regulatory requirements to be respected, paving the way for the broader adoption and use of UAVs.
2) We present specific use-cases for how our Fly-ABAC system enhances security and compliance for UAV operations, demonstrating how Fly-ABAC can enforce regulations in these use-cases so that both federal regulations and private property constraints are respected.
3) We provide simulation code and results that validate the effectiveness of the ABAC system in adhering to specified property owner preferences, successfully avoiding unauthorized zones, and generating sample maps for robust demonstrations of adherence to preset rules. The fully commented Fly-ABAC system implementation used in this project, complete with visualizations and A* pathing algorithm is provided at https://github.com/snowNnik/REU-2024.

This paper is organized as follows: Section 2 explores related works and their relation and significance to this paper. Section 3 provides some additional background ideas on ABAC that were formulated into the methodology described in Section 4. Section 5 describes the results of our simulations with illustrative figures. Section 6 explains issues and possible further research in this subject matter, and Section 7 concludes the paper.

II. Related Works

Beck et al. [14] introduced Privaros, a framework that is similarly designed to enforce privacy policies on commercial delivery drones. However, instead of controlling where drones fly, Privaros uses mandatory access control to ensure that drones comply with the privacy requirements of the host airspaces that they visit. The framework is built on ROS, a middleware commonly used in drone platforms, and integrates with India's Digital Sky portal for policy specification. The evaluation shows that Privaros can robustly enforce various data privacy policies with minimal impact on communication latency and power consumption.

Kamal et al. [2] also attempted to establish no-fly zones by using attribute-based access control. However, their research focused more on restrictions imposed by government organizations like the FAA with their data focusing on how much time it takes for equipment to receive data. Our research aims to create a general solution for government and private property through a more customizable framework.

Pavlo and Valerii [12] delves into the prominent examples of Attribute-Based, Role-Based, and Relationship-Based Access Control models in order to determine the benefits and drawbacks of each. Their experiments conclude that each system has situations where it thrives, such as Role-Based Access Control being able to be easily scaled up. In contrast, RBAC would serve as a good balance between independent organizations. Despite each of the other two models having their benefits, however, we chose ABAC due to the paper finding it to be more flexible than the others, allowing our proposed system to be constantly updated to fit very specific, shifting preferences of property owners.

Candra et al. [1] details their experiment trying to measure the difference between the A-Star and Dijkstra's algorithms

```
ATTRS = <String, userName>;<String, exclusionZone>;<String,
    droneOwnedBy>
PERMS = <Entry>; <nonEntry>
PA = <userName, Grid0x0> : Entry - <userName, Grid0x2> : Entry -
        <userName, Grid1x0> : Entry - <userName, Grid1x1> : Entry
            -
        <userName, Grid1x2> : Entry
ENTITIES = <Drone>;<Grid0x0>;<Grid0x1>;<Grid0x2>;
          <Grid1x0>;<Grid1x1>;<Grid1x2>;<ENV>
AA = <userName, Grid0x0> : <Grid0x0> - <userName, Grid0x1> : <
    Grid0x1> -
        <userName, Grid0x2> : <Grid0x2> - <userName, Grid1x0> : <
            Grid1x0> -
        <userName, Grid1x1> : <Grid1x1> - <userName, Grid1x2> : <
            Grid1x2>
```

Fig. 2. A Sample Fly-ABAC Policy. ATTRS takes attributes, PERMS specifies possible permissions, PA connects permissions to attributes using relations(separated by dashes(-)), ENTITIES specifies entities such as the drone and grid spaces(zones), and AA connects entities to their attributes with relations(again, separated by dashes(-)), thus allowing for very granular control over zone permissions.

in determining which can find the shortest path faster. The paper concludes that the A-Star algorithm is, on average, faster than Dijkstra's method, which motivated us to use the A-Star algorithm in our experiment.

III. Background

Enforcing constraints such as airspace access rights requires an access control system that can be dynamically changed as property owners' preferences vary over time. This system must also account for the preferences of multiple property owners and government preferences, which might be based on ever-changing environmental factors such as time and weather. We used the Attribute-Based Access Control (ABAC) paradigm [12], which allows for very granular access control, to fulfill these requirements. ABAC is an access control policy that evaluates a user's access to a particular object using a policy evaluator by considering various attributes. These attributes are characteristics that can be both of subjects (such as users or admins), of resources (such as data/file systems or, in our case, zones), or even of the environment. Attributes of subjects might include username or organization, attributes of resources could be owner or location, while attributes of the environment could be time or weather. Fly-ABAC considers the sum of all these attributes to determine whether a drone is allowed access to a particular zone [12].

IV. Methodology

A. Fly-ABAC Policy Implementation

In order to ensure that our system can accurately enforce diverse and specific requirements by different property owners, we identified key features that a sufficiently compliant system must implement. We composed our system to account for three important features: Positive Attributes, Negative Attributes, and Exclusion Zones. For the purposes of this paper, Positive Attributes are the attributes required to be present within either the drone, the property, or the environment in order to access an entity. For example, a property owner might require drones to be owned by Company A to enter their property, or they may require that it must be a clear and sunny day or before 5:00 pm for a drone to enter the property. By contrast, Negative Attributes are attributes that exclude the drone of permission to enter the property if they are present in the combination of attributes between the drone, the property, and the environment. An example of Negative Attributes would be: a property owner might not allow a drone to enter their property during rainy and windy weather to avoid potential damage, or they might prohibit drones from Company A specifically while allowing others. Finally, Exclusion Zones are when property attributes from an entity affect the attributes required to access another entity. For an example of Exclusion Zones in the real world, drones are not allowed to fly near areas where rescue operations are taking place [7], and are not allowed to fly in specific restricted airspaces like airports and military bases without permission [15].

B. Base Implementation and Development

To test our system's effectiveness in keeping drones off properties where they are unwelcome, we developed code to allow us to model an ABAC system with these features.

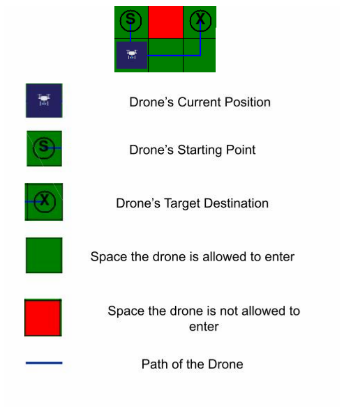
We developed a system that could intake objects such as permissions, users, and environment and also store their attributes within relation tables. To demonstrate how this attribute system can enforce these property rules, the code then evaluates every zone in a given grid to generate a grid of 'red' and 'green' zones, with red zones representing zones that the drone is not allowed access to and green zones representing zones the drone is allowed to fly over. Our model requires explicit permission for a drone to enter into a property, so if the property owner does not specify permissions for their property, it will be red by default.

In our simulation, once the policy is evaluated, a grid is formed with each tile being given either a green or red designation. The space is mapped as a 2D grid, a simple and effective method, despite the actual drone navigating a 3D environment. Our methodology simplifies the environment by only considering an aerial perspective.

Figure 2 provides an example of how a policy might look in our base implementation, specifying attributes, permissions, permission-attribute relations, entities, and object-attribute relationships. This very generalized system of policy description allows for very granular and specific control over property permissions. Figure 3 provides the map and pathing that is generated from the policy shown in Figure 2.

C. Implementation of Exclusion Zone and Negative Attributes

Our base implementation supports Positive Attributes by allowing entry if all attributes for entry are present (in the group of environment, user/drone, and zone) as seen in Figures 4 and 5 (where Figure 5 demonstrates a denial of access due to the lack of a needed attribute), however it did not support Negative Attributes or Exclusion Zones. We developed these two features on top of our base ABAC implementation. In our base code, to generate the grid every zones' $< Entry >$


S
X
Drone's Current Position
Drone's Starting Point
Drone's Target Destination
Space the drone is allowed to enter
Space the drone is not allowed to enter
Path of the Drone

Example.png

Fig. 3. A graphical depicting of the result of the policy found in Figure 2. The denial of access to the property represented at (0,1) [our code starts generating (0,0) from the top left corner, from the bottom left corner this would be (1,1)] is the result of there being no $< Entry >$ permission set up by the Fly-ABAC policy for that property. The other properties represent the property owner allowing all drones into their property with no restrictions as $< userName >$ attribute will always be in list of combined attributes between the drone, the environment used to verify whether or not the drone have permission to access that space.

permission was checked, which would return 0 or 1 to easily populate the grid. To add Negative Attributes as a feature we added a $< nonEntry >$ permission check right before the $< Entry >$ check, which if positive would skip the $< Entry >$ check and immediately populate the grid space with a 0 for denial of entry. If negative, the code could continue with an $< Entry >$ check as in the base code. Thus, if the requisite Negative Attributes are present entry is always denied, however if they are not satisfied the entry check proceeds as normal, and the drone is still required to have all the requisite attributes to enter the zone. An example of the application of Negative Attributes is shown in Figure 6 which denies access to the property because the drone is owned by Company A in the scenario and Company A drones are prohibited from entering that property due it's presence in the $< nonEntry >$ permission. This is different from Figure 5 because in that simulation the drone is required to be owned by Company A to enter the property due to it's presence in the $< Entry >$ action, but the drone is instead owned by a private citizen named Bob. In order to implement Exclusion Zones we allowed attributes of one zone to affect the attributes of other zones. Our code covers two examples of Exclusion Zones one where no drone is allowed fly regardless of the attributes have and another where the drone is only allowed access if it is owned by a certain entity. To do this, once the $< exclusionZone >$ declaration is found while assigning entity attributes, then, in a radius determined by the declaration of the $< exlcusionZone >$, either the name of the property is added to a $< nonEntry >$ permission which permanently denies access to every drone entering that territory or the $< Entry >$ permissions of the property are iterated through and adds the requirement that the drone must be owned by the value associated with the $< exclusionZone >$. An example

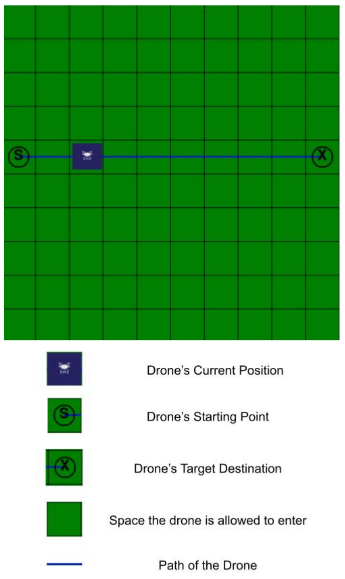


Fig. 4. This figure demonstrates what happens when the drone contains all of the attributes required to perform the $< Entry >$ action. With nothing to consider moving around the simulated drone moves in a straight line. This serves to prove that without obstacles our algorithm will move on the quickest path the A* algorithm can find on the way to its target.

of this feature's successful implementation can be seen in Figure 7 where the drone must be owned by the government to enter the property, but the drone in this scenario is owned by Company A so it is denied access to the area covered by the Exclusion Zone's radius, in this case for the 3 properties surrounding the origin of the Exclusion Zone.

D. Pathing Algorithm

Our project generated paths using the A* algorithm [1] on the grid our Fly-ABAC system generated. This ensured that the drone would actually respect the boundaries of our model and correctly follow given permissions that deny and allow entries when necessary. Our project adapted the A* algorithm code found at [13].

V. Results

Our implementation was able to accurately adhere to the preferences of property owners that we specified in every simulation we ran. We found that our Fly-ABAC system effectively modeled and enforced the various property preferences that owners might have, specifically positive and negative attributes, and exclusion zones, thus proving it to be an effective system for both federal and private property enforcement. Figure 5 illustrates how our Fly-ABAC successfully generated a path that avoids a zone with positive entry requirements that were not met by the "drone" (due to it lacking the necessary attributes). Figure 6 demonstrates how our model can successfully generate a path to avoid a zone where it meets attribute requirements for strict denial of entry. Lastly, Figure 7 shows how our system models exclusion zones. By adding a specific attribute to a particular square, a red zone is generated around it with a specified radius. This exclusion zone can be

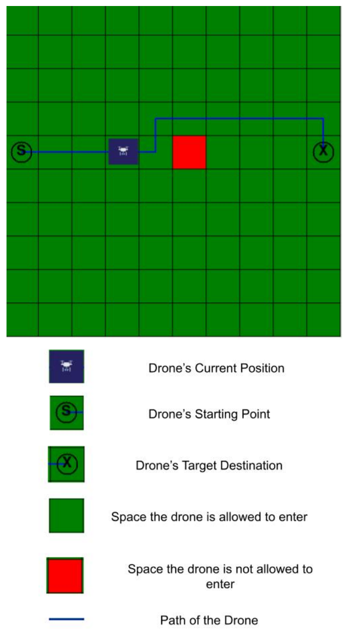

S
X
Drone's Current Position
Drone's Starting Point
Drone's Target Destination
Space the drone is allowed to enter
Space the drone is not allowed to enter
Path of the Drone

Fig. 5. This is an example of the Entry permission denying access. In this figure, the Drone is denied entry into the square at (5,4) because it lacks the required attributes needed to perform the Entry action on that space. Specifically the attribute indicating that the drone is owned by Company A, but, in this case, the drone is owned by Bob, so access is denied.

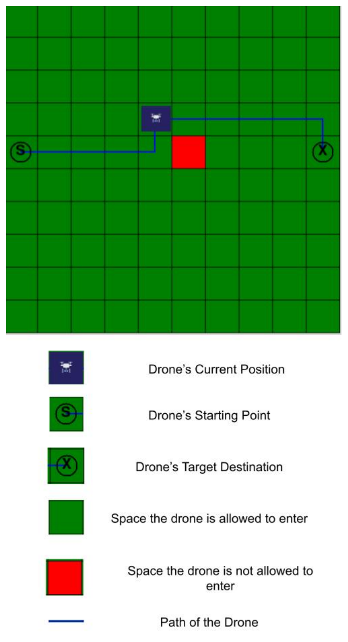

S
X
Drone's Current Position
Drone's Starting Point
Drone's Target Destination
Space the drone is allowed to enter
Space the drone is not allowed to enter
Path of the Drone

Fig. 6. This is an example of the nonEntry permission. Similar to Figure 5, the Drone is denied entry into the square at (5,4), but rather than lacking a required attribute, the drone has an negative attribute that disqualifies it from entering the property. In this case, the property owner refuses to allow drones owned by Company A onto his property and this drone has the attribute indicating that it is owned by Company A therefore access is denied.

configured to allow entry to drones with specific ownership, as might be needed for real world cases such as government or school zones.

Our implementation demonstrates Fly-ABAC's ability to handle complex property restrictions and preferences with accuracy and flexibility.

VI. Future Works

While our project demonstrated the successful construction of an ABAC policy when allowing access based on preset permissions, there are still areas in which this project could be improved. For example, our project only considers a two dimensional static scenario without regard to changing weather conditions or time. These factors would prove to be very important if property owners decide that drones shouldn't be on their property in turbulent weather conditions. Future experiments might integrate weather systems and a time system that updates in real time which would require the development of a navigational system which considers possible times saves that waiting for conditions to change might allow for. For example, waiting until a certain time to allow entry into a group of properties that allow drones on their premises only at select times of day might prove faster than attempting to fly around them in property dense areas like suburbs and cities leading to potentially saving time while traveling from the starting point to the destination.

Additional experiments could attempt to create more detailed simulations accounting for dynamic changes as mentioned earlier, but also incorporate navigating in the third dimension while dodging obstacles found above or on private properties. Our experiment only includes two dimensions and does not account for real-world situations where there are different rules for UAVs depending on how high the drone is flying (specifically regarding airspaces around military bases and airports) and what that might mean while attempting to access properties [15]. Allowing for operation in a three dimensional space would also allow the drone to move up and down to avoid obstacles creating more considerations for which way to avoid the obstacle would be more preferable for the drone while getting to and from its destination unharmed.

Additionally, our simulation doesn't take into account the specifications of the actual drone itself which would be important when determining how fast it would take a drone to go from the starting point and destination and how fast it could react to obstacles. A system could be created to hold the attributes of several drones consisting of data like their speed, weight, and how much power they can hold before needing a recharge to determine which one is best suited for the trip or perhaps detect if there is a better route available due to attributes of the specific drone. Furthermore, using these specifications the owners of the drone could determine if the drone was even capable of reaching its destination, not due to access permissions, but due to physical limitations like lack of charge or need of maintenance.

Finally, a more realistic simulation could be created by surveying property owners for their preferences relating to drones entering their property and accounting for uniquely

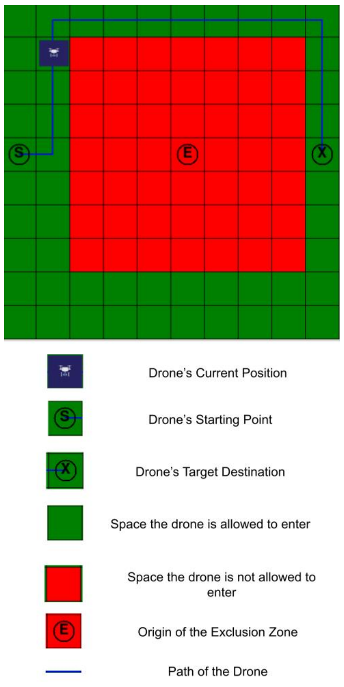

S
E
X
Drone's Current Position
Drone's Starting Point
Drone's Target Destination
Space the drone is allowed to enter
Space the drone is not allowed to enter
Origin of the Exclusion Zone
Path of the Drone

Fig. 7. This is the example of an Exclusion Zone which restricts access to properties in a certain radius around a property with the attribute exclusionZone. In this figure, the grid space (5,4) has the attribute exclusionZone with conditions that only drones owned by the government are allowed to fly in a radius of three properties from itself. The drone in this case is owned by Company A and as such is not permitted within the Exclusion Zone's radius. This type of designation would be useful in areas with temporary flight restrictions such as wildfires [7] or more permanent restrictions around places like airports and military bases [15].

shaped properties that might not fit in a grid. To accommodate this, an experiment could be conducted which integrates the back end and the front end of our implementation allowing for dynamic changes to permissions in real time and utilize maps which divide property lines from the real world to create more exact restrictions for the drone. This would allow for the creation for a more precise algorithm to be created or used to address the real world problem of navigating the skies without relying on precise shapes.

VII. Conclusion

Our system successfully modeled a range of different property restrictions, showing how the granularity offered by Fly-ABAC can allow for highly customized and precise control over property access.

The system's effectiveness in enforcing property preferences, including positive and negative attributes as well as exclusion zones, highlights its potential for widespread adoption in both federal and private contexts. Future work could explore integrating real-time environmental factors, 3D navigation, and user-driven dynamic policy updates to further enhance the system's capabilities. By ensuring UAVs can navigate within specified constraints, our system represents advancement in the field of property rights enforcement, paving the way for the increased adoption of UAVs for commercial and private use.

Acknowledgements

This material is based upon work partially supported by the National Science Foundation (NSF) under Grants No. 2150351 and 2131263. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of NSF.

References

[1] A. Candra, M. A. Budiman, and K. Hartanto, "Dijkstra's and A-star in finding the shortest path: A tutorial," 2020 International Conference on Data Science, Artificial Intelligence, and Business Analytics (DATABIA), pp. 28–32, Jul. 2020. doi:10.1109/databia50434.2020.9190342

[2] A. Kamal, J. Vidaurri, and C. Rubio-Medrano, "No-fly-zone: Regulating drone fly-overs via government and user-controlled authorization zones," Proceedings of the Twenty-fourth International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing, Oct. 2023. doi:10.1145/3565287.3617633

[3] "Amazon drone delivery is coming to Arizona," US About Amazon, Apr. 22, 2024. https://www.aboutamazon.com/news/transportation/amazon-drone-delivery-arizona

[4] B. Nassi, R. Bitton, R. Masuoka, A. Shabtai and Y. Elovici, "SoK: Security and Privacy in the Age of Commercial Drones," 2021 IEEE Symposium on Security and Privacy (SP), San Francisco, CA, USA, 2021, pp. 1434-1451, doi: 10.1109/SP40001.2021.00005. keywords: Privacy;Ecosystems;Security;Drones;Drones;Privacy;Security;Society,

[5] "B4ufly," B4UFLY — Federal Aviation Administration, https://www.faa.gov/uas/getting_started/b4ufly (accessed Aug. 5, 2024).

[6] "Can You Fly a Drone Over Private Property? State-by-State Guide 2024," JOUAV, Jul. 05, 2024. https://www.jouav.com/blog/can-you-fly-a-drone-over-private-property.html (accessed Jul. 29, 2024).

[7] "Drones and Wildfires are a Toxic Mix." Federal Aviation Administration

[8] Freepik, "Freepik - Free Graphic resources for everyone," Freepik, 2023. https://www.freepik.com/

[9] "How are drones used in agriculture?" Drone Launch Academy, https://dronelaunchacademy.com/resources/how-are-drones-used-in-agriculture (accessed Jun. 21, 2024).

[10] "How Drones Are Used for Search and Rescue," Skydio,

[11] J. Tran, "Drones Rely on Sensors to Perform Better and Become More Useful," Vision Systems Design, May 24, 2023. https://www.vision-systems.com/embedded/article/14294131/drones-rely-on-sensors-to-perform-better-and-become-more-useful

[12] P. Hlushchenko and V. Dudykevych, "Exploratory survey of access control paradigms and policy management engines," CMIS 2024 Computer Modeling and Intelligent Systems 2024, pp. 263–279, May 2024. Accessed: Jun. 24, 2024. [Online]. Available: https://ceur-ws.org/Vol-3702/paper22.pdf

[13] R. Belwariar, "A* Search Algorithm - GeeksforGeeks," GeeksforGeeks, Sep. 07, 2018. https://www.geeksforgeeks.org/a-search-algorithm

[14] R. R. Beck, A. Vijeev, and V. Ganapathy, "Privaros: A Framework for Privacy-Compliant Delivery Drones," Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security, Oct. 2020, doi: https://doi.org/10.1145/3372297.3417858.

[15] "The Federal Register," Code of Federal Regulations, https://www.ecfr.gov/current/title-14/chapter-I/subchapter-F/part-107#p-107.205(i) (accessed Jul. 24, 2024).

[16] "We are committed to using drones to inspect and maintain wind farms," Iberdrola, https://www.iberdrola.com/innovation/drones-wind-farms (accessed Jun. 21, 2024).