Fido: A String-Based Fuzzy Logic Mechanism for Content Extraction from UAV Data Lakes

Pallav Kumar Deb, *Graduate Student Member, IEEE*, Anandarup Mukherjee, *Graduate Student Member, IEEE*, and Sudip Misra, *Senior Member, IEEE*

Abstract—In this article, we propose – Fido – a fuzzy-based string comparison method for extracting content from data lakes in a camera-based Internet of Drones (IoD) environment. Data lakes support data dumping in their native format, which makes them suitable for real-time deployments such as IoD. However, parsing through these unstructured databases has its concerns, particularly data extraction. Existing works on image-based content extraction focus on complex geometric and matrix mathematical operations, which are expensive in terms of both computation and time. A straightforward and fast solution is necessary for overcoming such challenges. Towards this, we limit our operations to string comparisons between the user's request and tags in the data lake. In particular, we adopt a fuzzy-based approach as it is capable of efficiently handling spelling variations, out-of-order words, and partial matchings, compared to conventional string comparison methods. Through lab-scale experiments, we demonstrate the efficacy of Fido with an almost 80% similarity ratio between the request and response images and processing time of $80\mu s$. Further, we observe minuscule deviations between the coordinates of the request and response images (0.005 units).

Index Terms—Internet of Drones, UAVs, Fuzzy inference systems, String comparisons, FuzzyWuzzy

1 Introduction

Data lakes are gaining popularity in the current traction. They are useful in real-time applications, such as on-demand Unmanned Aerial Vehicle (UAV) image capturing and surveillance services [1] as it allows quick dumping of unprocessed data. This is because such services consist of an Internet of Drones/UAVs (IoD) from different manufacturers and they capture and store images from the same location with independent tagging schemes. Directly dumping the images and their tags on a storage platform (maybe terrestrial or air-borne) without pre-processing saves time and computational resources. While such storage practices require more space, they offer advantages such as flexibility, low organizational and filtration requirements, fast accessibility, and modifications, and are ideal for machine learning enthusiasts. Most importantly, data lakes remove the complexity of storing in specific formats/structures, which eventually removes significant overheads. Further, in contrast to data warehouses being application-specific, the content in data lakes may be used for multiple ones, which also increases the scope of an IoD deployment.

P. K. Deb, A. Mukherjee, and S. Misra are with the Department of Computer Science and Engineering, Indian Institute of Technology Kharagpur, India. e-mail: (pallv.deb,sudipm)@iitkgp.ac.in, anandarupmukherjee@ieee.org

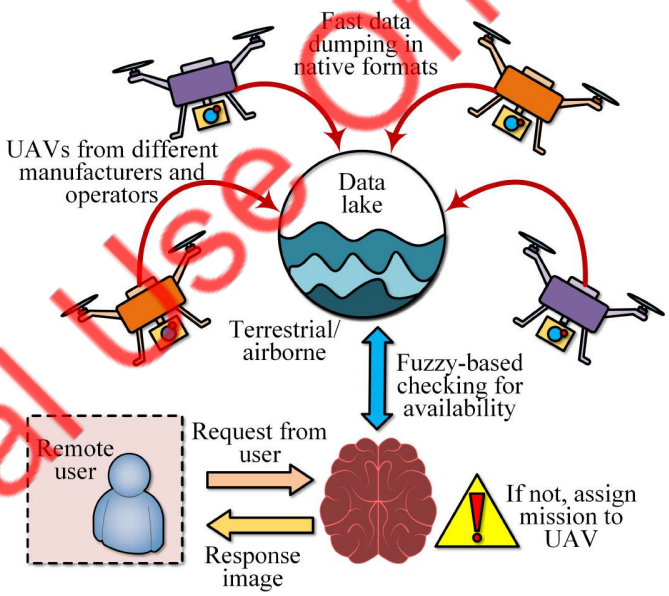

Figure 1: Overview of the proposed system.

1.1 Challenges

Although dumping data in their raw format has the mentioned advantages, the lack of a specific format has concerns. For instance, in applications such as UAV virtualization [2], the number of UAV operators and the end-users (that request services) in an IoD may be significantly high. Due to the heterogeneity of the operators and the UAVs, the data tagging schemes are also diverse. We elaborate on this issue with the application that we consider in this article. As shown in Fig. 1, we consider a UAV virtualization service or an IoD deployment that provides on-demand outdoor images as a service. These images may contain tags such as date, time, and UAV-centric data such as the x, y, and z coordinates along with the azimuth and zenith angles. Due to the lack of mandatory specificity, order, and consistency of the data fields in data lakes, the drone operators may choose to equally include/exclude the tags. They may include additional fields such as the name of the location, weather conditions, and many others. In a real-time scenario, particularly in IoD, although the identification of redundant requests from this vast vocabulary is daunting, it

is important to address this issue. This is because it enables the service providers to forward the images that are already in the data lake, instead of assigning a new mission to the UAV. Such solutions have the potential to reduce service delays, conserve energy, and increase flight time. Another important issue (apart from delays) is traversing the data lake as it is challenging to access and parse through its contents for an amateur operator.

1.2 Proposal: Fido

In this article, we propose Fido as a technique to decide whether to assign a new mission to a UAV or not based on the content available in a data lake. Fido is a Latin word meaning faithful. We will cover the possible deployment scenarios and network architecture in subsequent sections. Compared to the previous sections, we consider a simplified scenario for maintaining the simplicity of this article. As shown in Fig. 1, we consider a scenario where UAVs dump images to a data lake and remote users request for their image of interest. Based on the content and user request, we need to determine whether to send an existing image (or a portion of it) or assign a new mission to a UAV. Irrespective of the UAV type and operator, we impose uniform image tags. We consider these image tags in the form analogous to that of the Google Map URLs. For example: *26.136542,91.8025863,127m/data=!3m1!1e3*, which represents (from left to right) the latitude, longitude,altitude, and type of image. Depending on the use case, the operators may choose a different naming/tagging scheme. We determine the user's content of interest by comparing the request URL against those in the data lake, in contrast to the conventional methods of complex geometric and image matrix analysis. Since we consider homogeneous onboard cameras, information like the resolution and focal length is not necessary. However, they are important in the case of heterogeneous ones to identify if an already available image or part of it is suitable for a new incoming request. Since user requests with the exact coordinates as that of one of the images in the data lake are almost impossible, straightforward string comparison methods may not perform well. They fail to account for constraints such as spelling variations, partial matches, and others. Fuzzy-based string comparison methods on the other hand have the capability of bridging the gaps between the inconsistencies of the user request string URL and those in the data lake, which is the crux of this article.

1.3 Need for Fido

The primary motivation in proposing the Fido system is the complexity of extracting information from a data lake, particularly in real-time IoD deployments. In such applications, both fast data capture and its analysis are important. Towards this, data lakes only meet halfway as it allows easy data dumping without any stringent formatting rules. It also does not require the data lake host to create specific apriori structures or tables for the incoming data. However, complexities arise when retrieving the data. The user/operator/algorithm needs to understand the data and parse through each content while making sense of what they mean. Further, conventional methods opt for complex analytical or matrix manipulation methods. The proposed method is an attempt for simplifying this searching and locating the concerned content by limiting the operations to string comparisons.

1.4 Contribution

In this article, we propose Fido as a method for simplifying the search and identification of content from a data lake in IoD deployments. Towards this, the following highlight our major contributions:

- ***Fido:*** We propose a fuzzy-based string comparison method for determining the image for a user request.
- ***Simplification:*** We attempt to simplify the search in a data lake by limiting our operations to only string comparisons.
- ***Evaluation:*** We perform lab-scale experiments and discuss our observations in comparison to existing solutions.

It may be noted that the selection of the appropriate UAV in case of new missions is beyond the scope of this article. We focus only on making the decisions on the image content in response to the user requests.

2 Related Work

The most popular and primitive problem that attracts any researcher's attention in the case of camera-based applications is the coverage problem. Hoang *et al.* [3] solved a similar problem for inspecting surfaces using multiple UAVs. They devised methods for positioning each UAV and determining their trajectories based on a geometric primitive metric using particle swarm optimization. Their solution is primarily dependent on the image (from the camera onboard the UAVs) overlap percentage. The authors in [4] also used the image overlapping percentage to determine the UAV formations using the bird flocking method. For unknown terrains, Koutras *et al.* [5] designed a mechanism that plans the optimal positions for the drones on the fly by localizing unique landmarks from the UAV cameras. The proposed method and utility score are independent of the UAV flight characteristics and external environmental conditions.

UAVs have rapidly changing topologies and routing data among them (to the destined ground or aerial access points) is an important aspect. Towards this, Hong *et al.* [6] proposed a method for adapting to the changing topologies by finding nearby neighbors and routing accordingly. They defined two modes: formation keeping and formation reconfiguration. In the formation-keeping mode, the topology is consistent, and the previous routing mechanism works efficiently. On the contrary, the UAVs exchange hello messages in the case of formation reconfiguration mode to identify new neighbors and create the new network topology before transferring data packets. On the other hand, some approaches such as in [7] consist of strategically positioning UAVs to increase network throughput with minimal energy consumption. However, coverage problems or on-demand image capturing applications are not benefited from such solutions. This is because on receiving discrete user requests, the UAVs are bound to change positions. For the UAV routing mechanisms to work with high throughputs, the

UAVs must fly back to their designated positions, which reduces flexibility. While machine learning-based routing methods such as in [8] may be helpful in some cases, the time required in training them poses a tradeoff.

Caching in UAVs is a promising approach in saving network and hardware resources. The proposed Fido method requires the storage of metadata of the previously captured images. Although we do not depend on using high-speed memory, some content from the secondary is essential, which acts as our cached data. Among data caching in UAV networks, Ji *et al.* proposed a cache placement solution. They jointly optimized the cache placement and trajectory planning, which is of particular interest for this work. While implementation of the same is beyond the scope of this article, we plan to opt for the inclusion of the same in the future. A similar approach may be found in [9]. On the other hand, the authors in [10] emphasized the need for reliable data delivery and achieved it using caching in a blockchain. While they relied on the implicit security features of a conventional blockchain, they proposed a neural network-based blockchain. It is distributed among the devices in a conventional edge-cloud-based network and categorized according to the layers. The use of blockchain on UAVs and its application may be found in [11].

2.1 Synthesis

In this section, we observed different techniques that attempt to maximize network throughput while planning optimal flight trajectories. In the case of camera-based UAV networks, most of the research focuses on novel image stitching techniques. This is coupled with flocking and UAV swarm coordination techniques. Further, once the UAV positions are fixed, most literature focuses on devising methods for optimal routing and throughput maximization. The data caching approaches on the UAVs also have a similar motive. In contrast to the works covered in this section, the proposed Fido method focuses on making decisions whether the UAVs need to react to the incoming user request based on the coordinates. Non-repetition of the tasks helps in increasing flight time and makes time for serving other requests. Additionally, the exchange of information among the UAVs is limited to RESTful messages such as GET, POST, PULL, and DELETE, which helps in further resource conservation. At the entity responsible for performing computations, while most of the solutions depend on image correlation and isomorphism techniques, we rely on string comparison methods. In summary, the proposed Fido offers a reduction in processing complexities on the IoDs, network resource consumption, and serves more user requests, while avoiding repeated mission assignments.

3 Network Architecture

In this article, we consider an application scenario that provides outdoor images to the users on-demand and is facilitated by UAV virtualization [2]. Consequently, there is no restriction to the number and type of participating UAVs, their manufacturers, and operators on the IoD. Accordingly, irrespective of the type of UAV, we consider a set of UAV operators $U_{ops} = \{u_{o1}, u_{o2}, \ldots, u_{on}\}$ and each of them may have 'm' number of UAVs. For instance, for operator u_{op1}, the set of UAVs are $V_{uav}^{op1} = \{v_1^{o1}, v_2^{o1}, \ldots, v_m^{o1}\}$. These UAVs dump the captured images on a device hosting the data lake ($\mathcal{D}$), and as shown in Fig. 2, it may either be terrestrial (top left) or airborne (top right). Depending on the developer, the connection between the data lake and the UAVs may be in the form of any of the standard network topologies (mesh, star, token rings, and others). In this article, Fido is suitable for both deployment architectures. The UAVs dump the images along with their tags (refer Section 1.2) to the device hosting the data lake and the proposed method also occurs on it. Programmatically, we initiate new threads on every request for faster response. In case of the new request matches, the consequent image is sent to the user. In the case of no match, the system assigns a UAV to travel to the location and capture the image for storage and send a user response. Since the cameras cover a range of the area (which depends on the field of view), an exact match on the coordinate values is not necessary. We explain the decision mechanism in the next section. It may be noted that the airborne deployment architecture may also have a distributed data lake. However, this article does not deal with such scenarios and we will address this in our extended work.

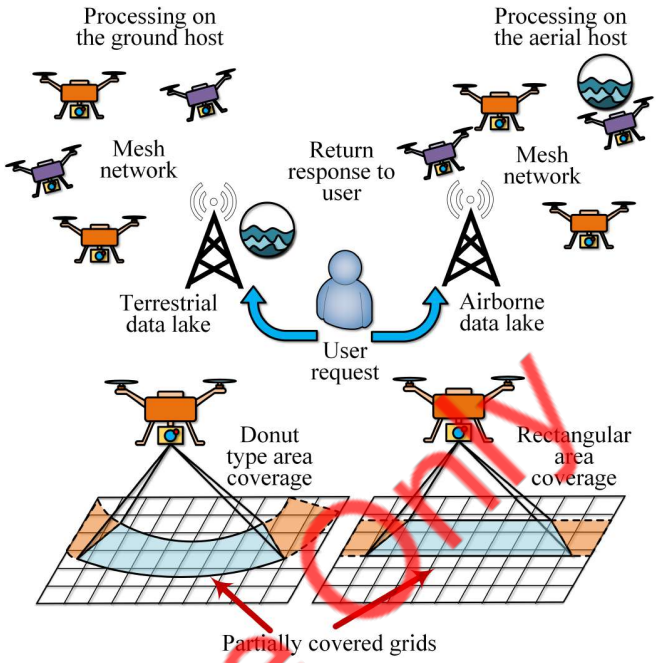

Figure 2: Possible network architectures.

4 Working Mechanism

Although this article is an attempt to simplify data retrieval from a data lake, the application scenario may seem similar to the coverage problem. From the list of possible coverage problems, such as point coverage, barrier coverage, area coverage, and others, the proposed work falls under the category of area coverage. The reason behind the bias towards area coverage is because a camera has a certain Field Of View (FOV) associated with its focal length. As shown in Fig. 2, there may be two possible FOV considerations. Both formats realistically represent the UAV camera's 3D

sector view on a 2D grid space. The problem with the first representation (left) is that the FOV is in the form of a donut. On the other hand, the second representation (right) is a square or rectangular FOV. We avoid curvatures of the FOVs and consider the coverage scenarios analogous to the one on the right. However, Fido will work for both cases. As mentioned earlier, we adopt a fuzzy-based string matching mechanism that computes the similarity ratio of the user requests against those available in the data lake that is developed on top of the Levenshtein distance method. We elaborate on how it works.

4.1 Levenshtein Distance

There exist multiple string comparison methods, such as but not limited to, Gestalt pattern matching, Sequence matcher, and Levenshtein distance. The Gestalt pattern matching tries to find the best contiguous matching sequence and the Sequence matcher is its recursive version. On the other hand, the Levenshtein distance, also known as the edit distance, calculates the distance between two strings in terms of the number of edits. This includes the number of insertions, deletions, and substitutions necessary in one of the strings to make it similar to the other. For interested readers, the expressions involved for the mentioned string comparison mechanisms are available at [12]. In this article, rather than the edit distance, we are more interested in how similar the user request strings are for the identification of the images. Consequently, a similarity score is an appropriate metric, which is possible to derive directly from the Levenshtein distance. For a Levenstein distance $d_{lev}(a, b)$ between two strings a and b, the Levenshtein similarity score is the ratio between the difference of the sum of the lengths of the two strings and $d_{lev}(a, b)$, divided by the sum of the lengths of the two strings.

4.2 Bias for the Fuzzy-Based Approach

While the Levenshtein ratio is a good and straightforward mechanism for finding string similarities, it demonstrates low efficiency in cases that fail to adhere to its scope. This is because it is designed based on the number of edits necessary for making a pair of strings similar to one another. Some of the cases when the Levenshtein ratio falls short in identifying similar strings are, but are not limited to, as follows:

- Spelling variations
- Out of order words
- Partially matching strings
- Strings with different lengths

In such scenarios, a method that works beyond binary assessments is necessary. Towards this, fuzzy-based inference systems are appropriate as they focus on appropriate reasoning, rather than an accurate one. Such methods, in addition to relatively simple computation routines, help in dealing with inconsistencies with a more human-like approach than a discrete machine. However, the Levenshtein distance has its own set of advantages. We account for these arguments and attempt to exploit both Levenshtein and fuzzy-logic methods, we opt for a Levenshtein-based fuzzy method for Fido. Some applications of fuzzy-based string matching solutions include, but are not limited to, listener transcripts [13] and plagiarism detection [14].

Algorithm 1: Fido

```
Result: Response image
initialization;
for Each entry in 𝒟 do
    if Entry image altitude ≥ Request image altitude then
        Perform the fuzzy-based string comparison;
    else
        pass;
    end
end
if Match found then
    Send image from 𝒟 with highest similarity as the
    response image;
else
    Assign mission to UAV;
end
```

4.3 Fido: Decision Making

The decision-making process primarily depends on the position of the UAVs and the type of the camera on-board. By type, we mean its focal length and megapixels. The focal length is necessary for calculating the FOV and angular FOV (AFOV), and the resolution/megapixels for quantifying the digital zoom limits. This is because as we perform a digital zoom on an image, the megapixel count decreases, which affects the image clarity. The digital zoom is important to facilitate using a part of an image to serve a user request in case it falls under an already covered area. It may be noted that for the same number of pixels, the FOV and resolution (spatial) are inversely related, meaning if we increase the FOV, the spatial resolution decreases. Since the non-deteriorated zoom limit depends on the camera and its mode, we recommend including it as one of the tags in the image. However, since we do not have control over the tags, by default, we recommend reducing the size of the image and then performing digital zoom up to a certain limit (while maintaining the spatial resolution). If the image quality degrades on zooming, it is beneficial to assign a UAV to capture a new image to maintain user satisfaction. Another issue that needs attention is if the user request's region of interest is partial coverage of the grid spaces (refer Fig. 2) of an already existing image. Typically, the coverage ratio is the metric of concern. However, since these are the extreme corners of the images, we recommend capturing new images irrespective of the coverage ratio. We consider performing zooms/scaling only on the areas that completely cover the grid cells. This is because on an already captured image, optical zoom is not possible and digital zoom may significantly degrade the content. In summary, while accounting for the UAV constraints, on receiving a new request, we assign new missions to a UAV in the case of any one the following conditions:

- The image does not exist in the data lake.
- The user has requested a real-time/current image.
- The quality of the image degrades on performing digital zoom.
- Partially covered grid cell.
- The user requested an image from a higher altitude than those available in the data lake.

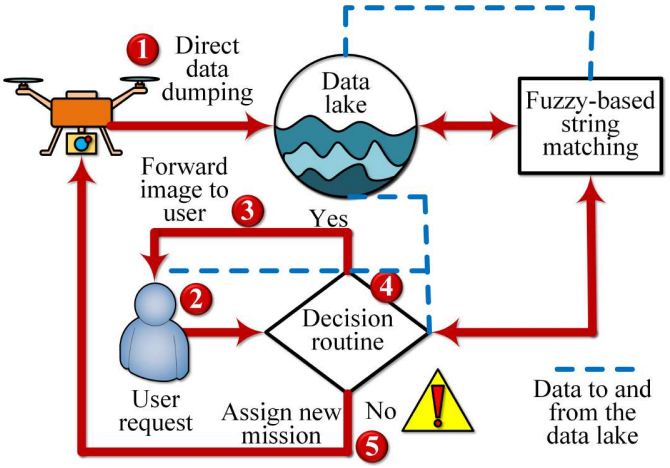

Figure 3: Information flow in Fido.

The last condition is important because a higher altitude covers a larger FOV. Those with lower altitudes may not be able to cover those areas. A possible solution is opting for an image stitching technique such as in [15]. In this work, we consider altitudes up to 150 meters, which is the maximum height at which a commercially available camera-based UAV may achieve. We limit the search scope of the images in the data lake up to two decimal places of the xy coordinates. The bias for this decision is after a stochastic study of the maximum variation of the coordinate values from a height of 150 meters. Algorithm 1 represents the proposed method.

4.4 Fido: Information Flow

As shown in Fig. 3, the UAVs capture images and dump them into a data lake (Step 1) according to the tags in Section 1.2.On receiving new requests from the users, we first do some pre-processing, which is a simple conversion of the entire text to lower case. On receiving requests from the user (Step 2), we start traversing the data lake. In the case of a close match, the corresponding image is sent to the user (Step 3). Based on the user request, an image that is already present in the data lake may require tuning (crop, pan rotate, or zoom) before sending it to make the response more appropriate (Step 4). Otherwise, Fido assigns a new mission to one of the UAVs based on the conditions in Section 4.3 (Step 5).

5 Experiment Setup

We use a personal laptop with an i5 processor. We realize the proposed work using Python 3.8 and use the FuzzyWuzzy package[1] for the fuzzy-based string comparison. We use tags similar to those in Google maps (refer example in Section 1.2) for naming our images and generate random strings to mimic user requests. We present our results on using the fuzzy-based string comparison method (FZ) in comparison to popular string matching techniques like *Sequence Matcher* (SM) and *Levenshtein Distance* (LD). Details on the two are available in [12].

1. https://chairnerd.seatgeek.com/fuzzywuzzy-fuzzy-string-matching-in-python/

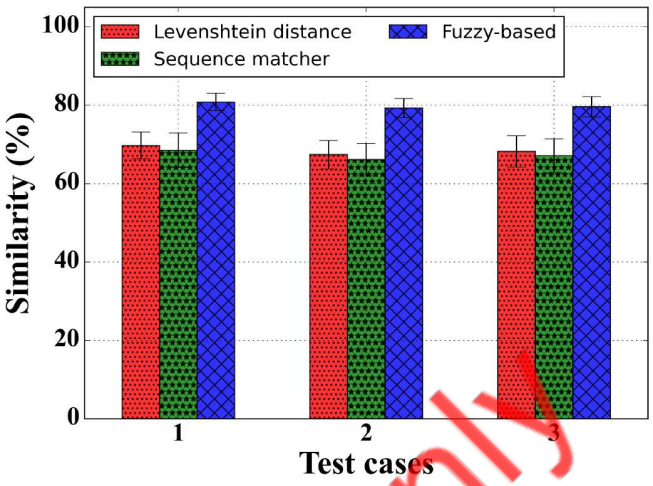

Figure 4: Similarity of the resulting image string in comparison to the user request on using different methods.

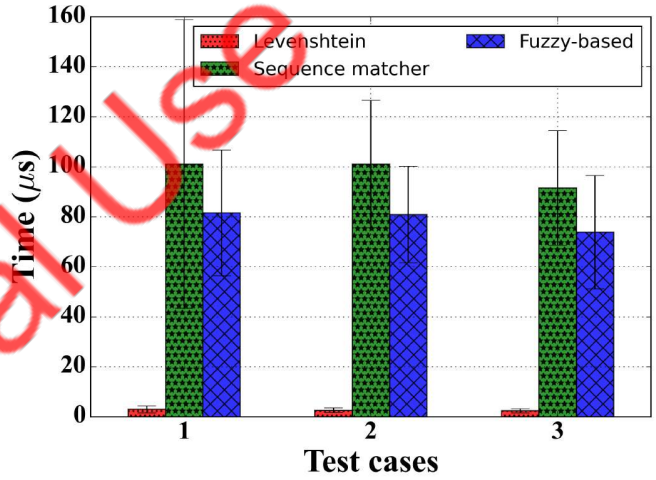

Figure 5: Processing time required by the different string comparison methods.

6 Result and Discussions

We present our observations in terms of the similarity ratios, processing time, and deviation of the results from the user requests. It may be noted that the results are an average from 3 arbitrarily chosen test cases run 50 times each. The test cases represent variation in the locations (different states).

6.1 Similarity Percent

We compute the similarity percentage of the expected resulting image for the user requests on using each of the previously mentioned string comparison methods. As expected, the proposed FZ method offers better similarity measures by almost 80% (refer Fig. 4). We attribute this superiority to the flexibility of fuzzy-based solutions. They are capable of overcoming uncertainties in a much better manner, compared to standard arithmetic methods (such as LD and SM in this work). Both LD and SM demonstrate similar results, with LD slightly higher scores. Interestingly, each of the methods shows consistent results in all three

cases, which illustrates the stability of each of the string comparison methods and that of the proposed work. The observations in Fig. 4 strengthens our bias for selecting FZ for realizing this work.

6.2 Processing Time

We record the time required by each of the string comparison methods for finding each of the similarity scores and present them in Fig. 5. Since SM is relatively straightforward compared to LD and FZ as it focuses on finding the longest common subsequence, it requires significantly low time. On the other hand, LD focuses on finding the number of modifications (deletion/insertion/substitution) required for making the two strings similar. The iterative comparisons cause an increase in the processing time, leading to the highest necessary time ($100\mu s$ on average). This will cause delays, which is not suitable for real-time applications. It also exhibits high variation in its processing time. Although we may attribute the variation to the background applications running on the PC, we observe stable variations in the case of the other methods. Interestingly, FZ shows a lower processing time ($80\mu s$) with much higher stability in its values. This observation together with that in Section 5 confirms the selection of FZ for this work.

6.3 Deviation

We present the deviation of the images suggested by Fido, in comparison to the actual request. Since Fido is a string comparison method, its deviation from the actual metrics is an important parameter to demonstrate its efficiency from a mathematical standpoint. Towards this, Fig. 6 depicts the variation of the xy coordinates and the altitude values (average of absolute values). For ease of representation, we present the altitude in $\times 10^4$ (in meters). On the other hand, the x and y coordinates are the numerical differences. We observe a minuscule variation of the result from the user request, particularly in the range of less than 0.006 for both x and y coordinates, which shows the efficiency of our method. While the altitude difference is not a concern as we extract the region of interest only from images captured from higher altitudes, it highly depends on the camera resolution to get a clear image after cropping, zooming, or scaling. We leave setting the image clarity threshold for the developers and operators.

6.4 Discussion

The observations in this section establish the feasibility of Fido, both with respect to its efficiency and processing requirements. The results in Sections 6.1 and 6.2 suggests how the Fuzzy method identifies the similarity in the user requests within tolerable time. On the other hand, we observe in Fig. 6 (Section 6.3) how close the results by Fido are with respect to the user requests.

From our observations, we comment that the proposed Fido method is a potential attempt to simplify the search in data lakes, particularly on location-aware image services. The fuzzy-based string comparison technique avoids other complex operations such as image or location comparison, which are typically dependent on complex mathematical

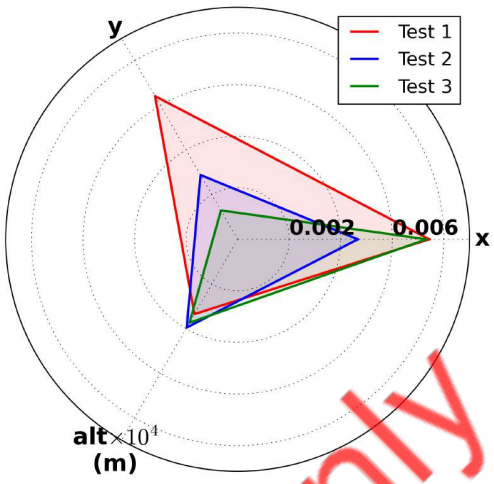

Figure 6: Deviation of coordinate parameters from the user request compared to result.

operations. It simplifies the search routine, making it suitable for real-time applications. Since the UAVs do not need to fly to the locations on every request, Fido helps in reducing energy consumption, which also increases the UAV availability for serving more requests. However, Fido will work efficiently in cases when partially or completely similar and out-of-order tags are used. In the case of random and arbitrary tags, Fido will under-perform. Additionally, Fido requires extra memory for keeping track of the metadata. However, compared to the size of the data lakes, Fido's requirement is minuscule.

7 Conclusion

In this article, we presented Fido, a method for extracting information or image from a data lake in an IoD environment. We focused on overcoming the complexities of analytical methods and demonstrated the use of string comparison methods. Since exact match of coordinates from the users and that in the data lake is almost impossible, we proposed using fuzzy-based string comparison methods instead of any other comparison routine. We performed lab-scale experiments and presented the efficiency of the proposed work in comparison to standard methods in literature along with the time required by each. Further, we presented the deviation of the tags of the response images in comparison to the user request string.

In the future, we plan to extend this work by developing methods for determining the necessary actions before sending the images in response to the user requests. For instance, decisions such as cropping, scaling, and magnifying the images. Also, metrics for determining the quality of user satisfaction with respect to the resolution of the response image are also important, which we will address in our extended work.

References

[1] O. Cetinkaya, D. Balsamo, and G. V. Merrett, "Internet of MIMO Things: UAV-Assisted Wireless-Powered Networks for Future Smart Cities," *IEEE Internet of Things Magazine*, vol. 3, no. 1, pp. 8–13, 2020.

[2] N. Pathak, S. Misra, A. Mukherjee, A. Roy, and A. Y. Zomaya, "UAV Virtualization for Enabling Heterogeneous and Persistent UAV-as-a-Service," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 6, pp. 6731–6738, 2020.
[3] V. T. Hoang, M. D. Phung, T. H. Dinh, and Q. P. Ha, "System Architecture for Real-Time Surface Inspection Using Multiple UAVs," *IEEE Systems Journal*, vol. 14, no. 2, pp. 2925–2936, 2020.
[4] J. Lwowski, A. Majumdar, P. Benavidez, J. J. Prevost, and M. Jamshidi, "Bird Flocking Inspired Formation Control for Unmanned Aerial Vehicles Using Stereo Camera," *IEEE Systems Journal*, vol. 13, no. 3, pp. 3580–3589, 2019.
[5] D. I. Koutras, A. C. Kapoutsis, and E. B. Kosmatopoulos, "Autonomous and Cooperative Design of the Monitor Positions for a Team of UAVs to Maximize the Quantity and Quality of Detected Objects," *IEEE Robotics and Automation Letters*, vol. 5, no. 3, pp. 4986–4993, 2020.
[6] L. Hong, H. Guo, J. Liu, and Y. Zhang, "Toward Swarm Coordination: Topology-Aware Inter-UAV Routing Optimization," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 9, pp. 10 177–10 187, 2020.
[7] S.-F. Chou, A.-C. Pang, and Y.-J. Yu, "Energy-Aware 3D Unmanned Aerial Vehicle Deployment for Network Throughput Optimization," *IEEE Transactions on Wireless Communications*, vol. 19, no. 1, pp. 563–578, 2020.
[8] P. K. Deb, A. Mukherjee, and S. Misra, "XiA: Send-it-Anyway Q-Routing for 6G-Enabled UAV-LEO Communications," *IEEE Transactions on Network Science and Engineering*, pp. 1–1, 2021.
[9] J. Ji, K. Zhu, D. Niyato, and R. Wang, "Joint Cache Placement, Flight Trajectory, and Transmission Power Optimization for Multi-UAV Assisted Wireless Networks," *IEEE Transactions on Wireless Communications*, vol. 19, no. 8, pp. 5389–5403, 2020.
[10] V. Sharma, I. You, D. N. K. Jayakody, D. G. Reina, and K.-K. R. Choo, "Neural-Blockchain-Based Ultrareliable Caching for Edge-Enabled UAV Networks," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 10, pp. 5723–5736, 2019.
[11] N. Pathak, A. Mukherjee, and S. Misra, "AerialBlocks: Blockchain-Enabled UAV Virtualization for Industrial IoT," *IEEE Internet of Things Magazine*, vol. 4, no. 1, pp. 72–77, 2021.
[12] G. A. Rao, G. Srinivas, K. V. Rao, and P. P. Reddy, "Characteristic Mining of Mathematical Formulas from Document-A Comparative Study on Sequence Matcher and Levenshtein Distance Procedure," *International Journal of Computer Sciences and Engineering*, vol. 6, no. 4, pp. 400–403, 2018.
[13] H. R. Bosker, "Using Fuzzy String Matching for Automated Assessment of Listener Transcripts in Speech Intelligibility Studies," *Behavior Research Methods*, pp. 1–9, 2021.
[14] M. S. R. K. Nag, G. Srinivas, K. V. Rao, S. Vakkalanka, and S. Nagendram, "Comparative and Experimental Study in Identifying the Similarity between Languages for Plagiarism Detection and Efficient Language Translation," *Materials Today: Proceedings*, 2021.
[15] H. Ismail, A. Rahmani, N. Aljasmi, and J. Quadir, "Stitching Approach for PV Panel Detection," in *Proceedings of Advances in Science and Engineering Technology International Conferences (ASET)*, 2020, pp. 1–4.

BIOGRAPHIES

Pallav Kr. Deb is a Ph.D. Research Scholar in the Department of Computer Science and Engineering, Indian Institute of Technology Kharagpur, India. He received his M.Tech degree in Information Technology from Tezpur University, India in 2017. Prior to that, he has completed the B. Tech degree in Computer Science from the Gauhati University, India in 2014. The current research interests of Mr. Deb include UAV swarms, THz Communications, Internet of Things, Cloud Computing, Fog Computing, and Wireless Body Area Networks. Further details about him are available at https://pallvdeb.github.io/

Anandarup Mukherjee is a Ph.D. research scholar at the Smart Wireless Networking and Applications (SWAN) Laboratory, Department of Computer Science and Engineering at the Indian Institute of Technology, Kharagpur (IIT Kharagpur). He is also the Director and Co-Founder of the IoT startup, SensorDrops Networks Private Limited (http://www.sensordropsnetworks.com). His research interests include, but are not limited to, networked robots, unmanned aerial vehicle swarms, Internet of Things, Industry 4.0, 6G, and THz Networks, and enabling deep learning for these platforms for controls and communications. His detailed profile can be accessed at http://www.anandarup.in

Sudip Misra (M'09 SM'11) is a Professor and Abdul Kalam Technology Innovation National Fellow in the Department of Computer Science and Engineering at the Indian Institute of Technology Kharagpur. He received his Ph.D. degree in Computer Science from Carleton University, in Ottawa, Canada. His current research interests include Wireless Sensor Networks and Internet of Things. Dr. Misra has been serving as the Associate Editor of different journals such as the IEEE Transactions on Mobile Computing, IEEE Transactions on Vehicular Technology, IEEE Transactions on Sustainable Computing, IEEE Network, and IEEE Systems Journal. Further details about him are available at http://cse.iitkgp.ac.in/ smisra/.

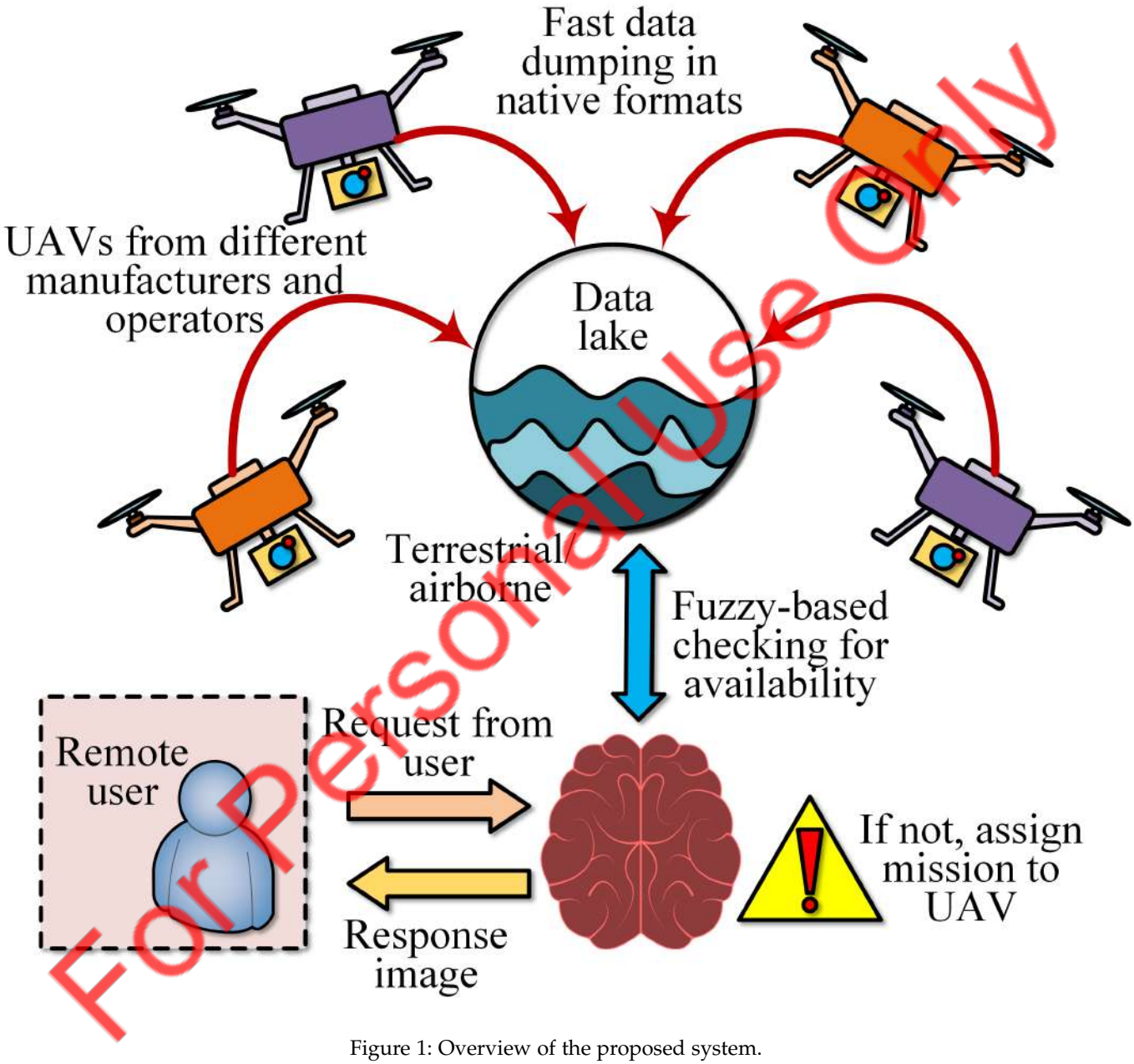

Figure 1: Overview of the proposed system.

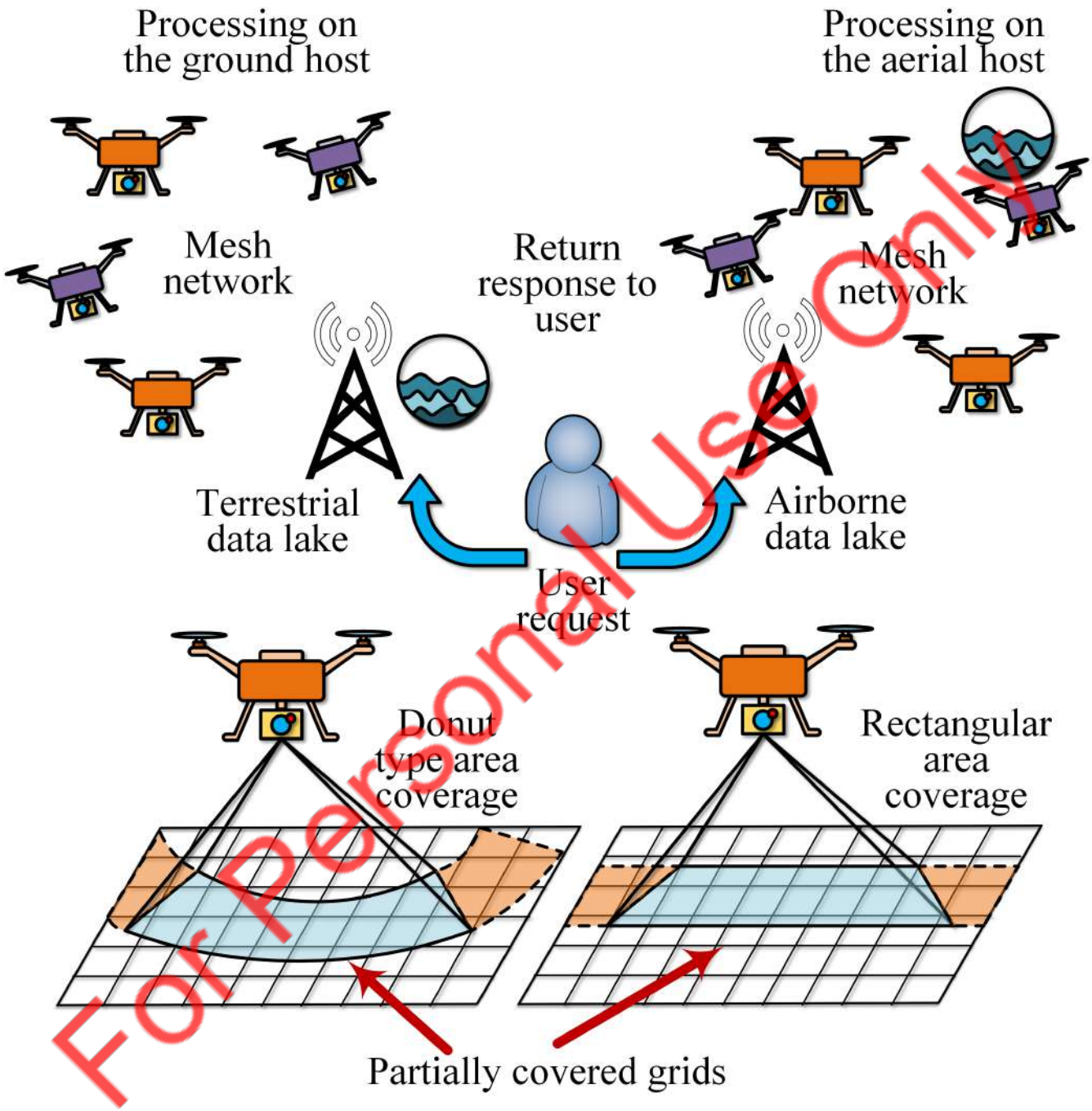

Figure 2: Possible network architectures.

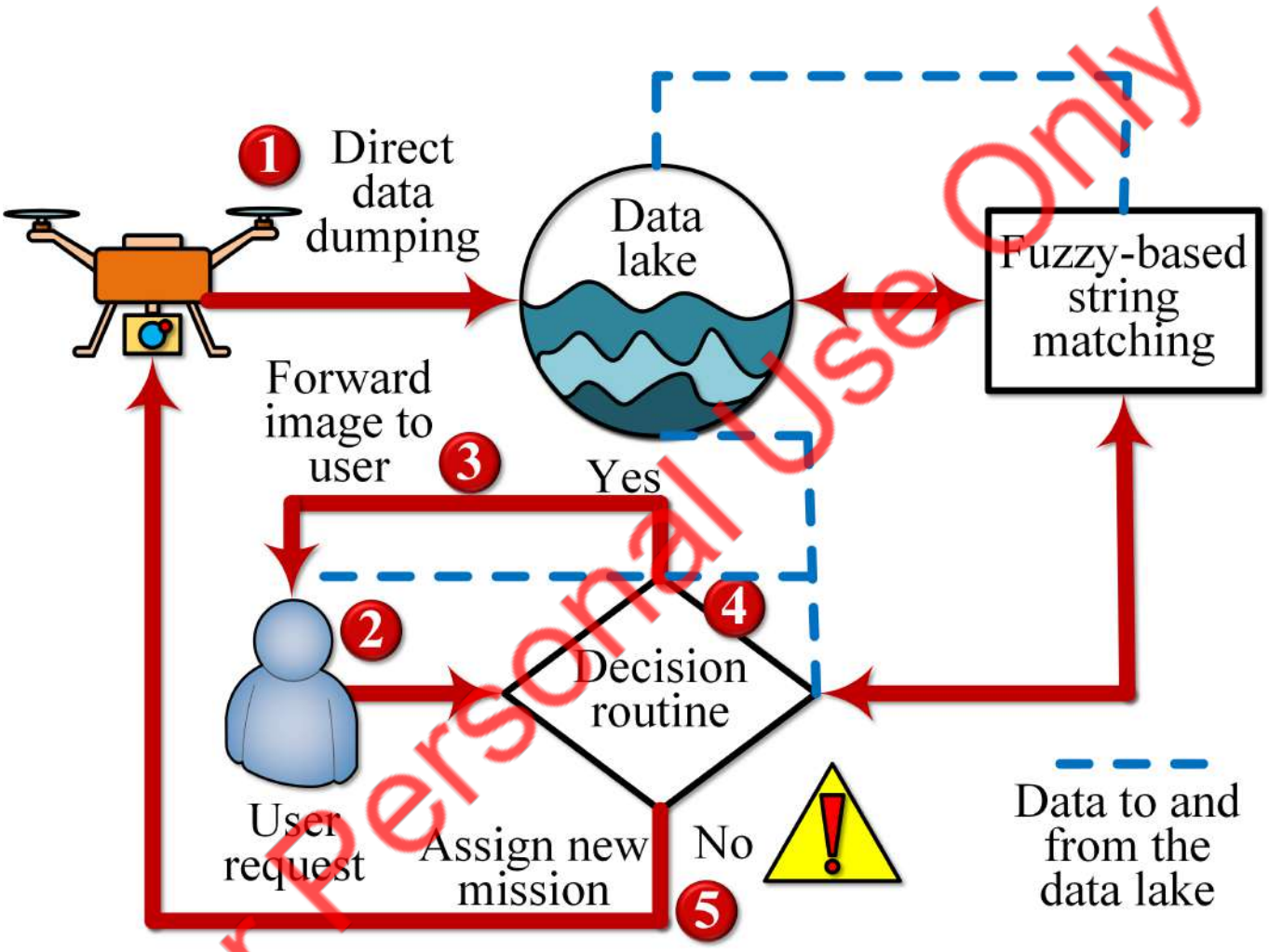

Figure 3: Information flow in Fido.

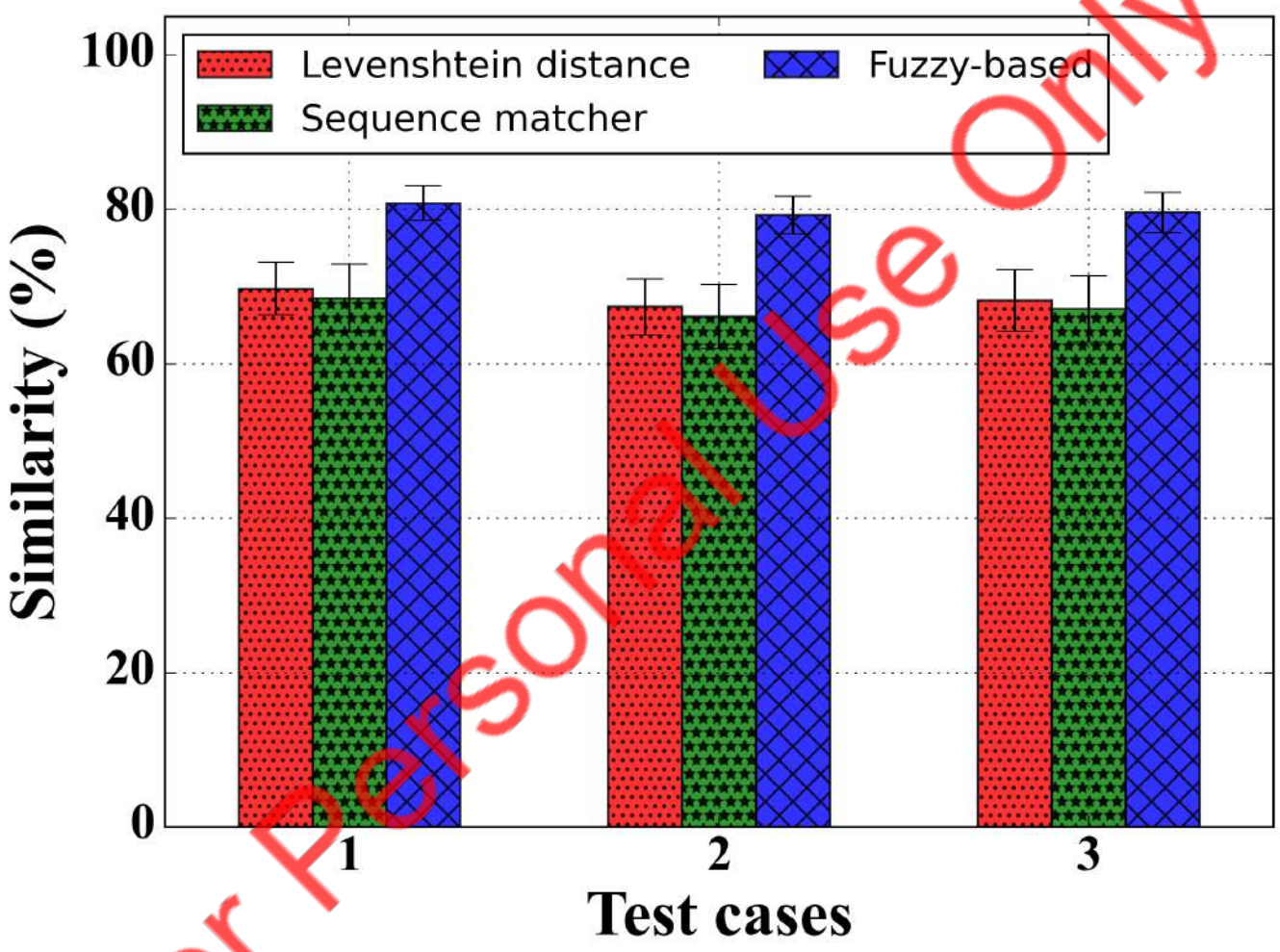

Figure 4: Similarity of the resulting image string in comparison to the user request on using different methods.

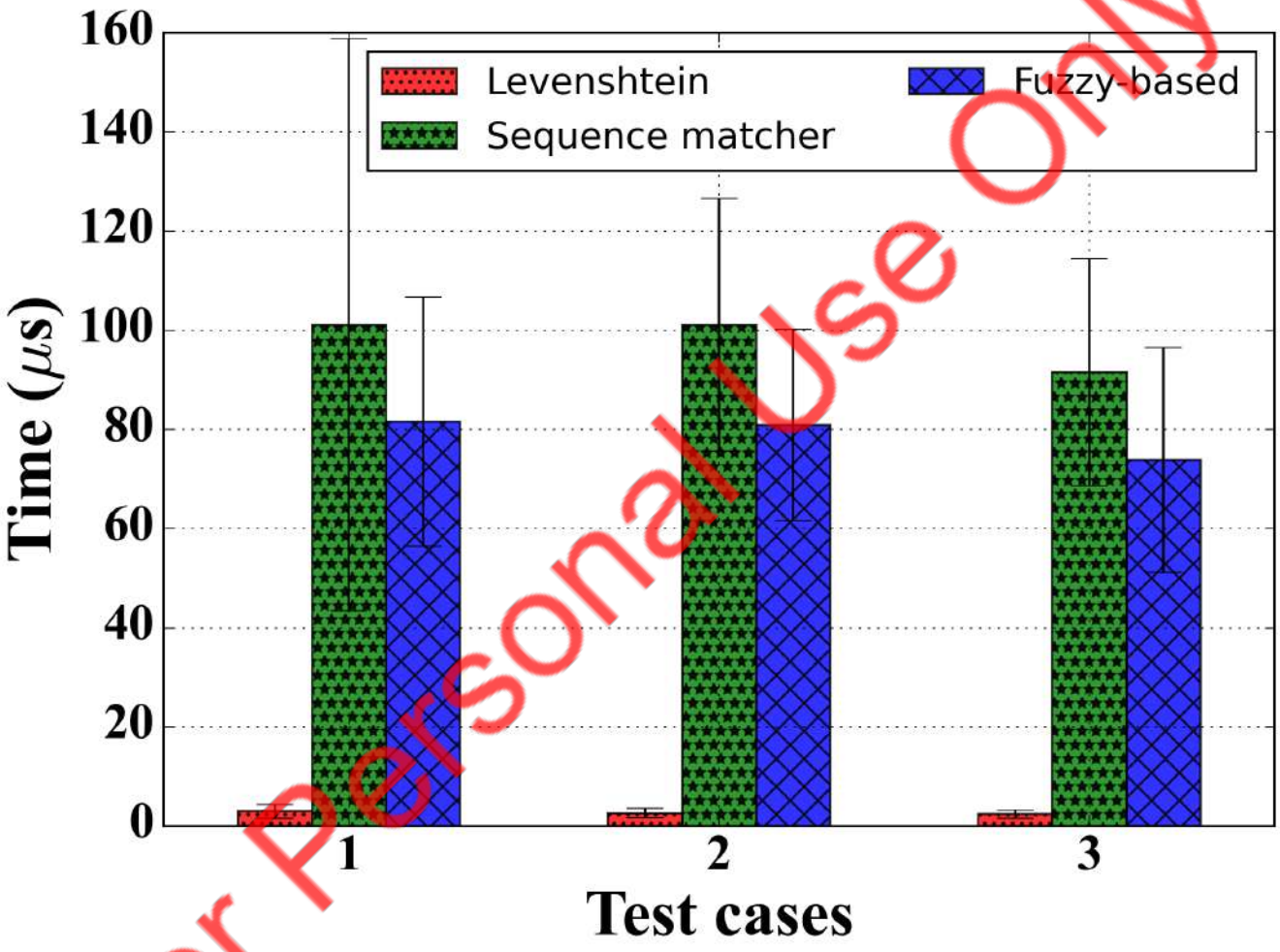

Figure 5: Processing time required by the different string comparison methods.

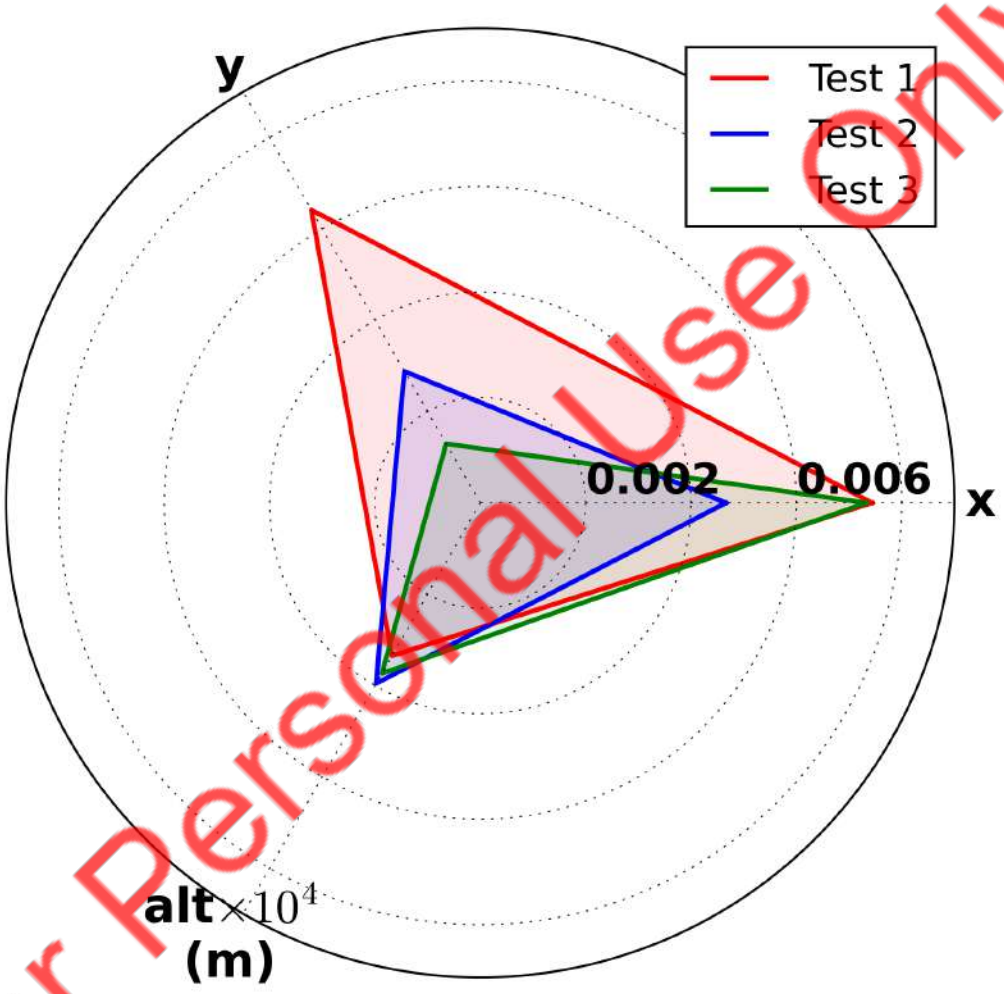

Figure 6: Deviation of coordinate parameters from the user request compared to result.