future internet

Article

Distributed Mobility Management Support for Low-Latency Data Delivery in Named Data Networking for UAVs

Mohammed Bellaj [1,*], Najib Naja [1,*] and Abdellah Jamali [2]

[1] National Institute of Posts and Telecommunications, Rabat 10000, Morocco
[2] National School of Applied Sciences of Berrechid, University of Hassan I, Settat 26000, Morocco; abdellah.jamali@uhp.ac.ma
* Correspondence: bellaj.mhammed@taalim.ma (M.B.); naja@inpt.ac.ma (N.N.)

Abstract: Named Data Networking (NDN) has emerged as a promising architecture to overcome the limitations of the conventional Internet Protocol (IP) architecture, particularly in terms of mobility, security, and data availability. However, despite the advantages it offers, producer mobility management remains a significant challenge for NDN, especially for moving vehicles and emerging technologies such as Unmanned Aerial Vehicles (UAVs), known for their high-speed and unpredictable movements, which makes it difficult for NDN to maintain seamless communication. To solve this mobility problem, we propose a Distributed Mobility Management Scheme (DMMS) to support UAV mobility and ensure low-latency content delivery in NDN architecture. DMMS utilizes decentralized Anchors to forward proactively the consumer's Interest packets toward the producer's predicted location when handoff occurs. Moreover, it introduces a new forwarding approach that combines the standard and location-based forwarding strategy to improve forwarding efficiency under producer mobility without changing the network structure. Using a realistic scenario, DMMS is evaluated and compared against two well-known solutions, namely MAP-ME and Kite, using the ndnSIM simulations. We demonstrate that DMMS achieves better results compared to Kite and MAP-ME solutions in terms of network cost and consumer quality-of-service metrics.

Keywords: named data networking; producer mobility management; distributed mobility

check for updates

Citation: Bellaj, M.; Naja, N.; Jamali, A. Distributed Mobility Management Support for Low-Latency Data Delivery in Named Data Networking for UAVs. *Future Internet* **2024**, *16*, 57. https://doi.org/10.3390/fi16020057

Academic Editors: Peiying Zhang, Haotong Cao and Keping Yu

Received: 9 January 2024
Revised: 29 January 2024
Accepted: 1 February 2024
Published: 10 February 2024

1. Introduction

UAVs, commonly called drones, have revolutionized the way aerial operations are conducted. Originally developed for military surveillance purposes [1], UAVs have experienced a drastic increase in usage during recent years and have become a valuable tool in various civilian industries [2] (e.g., traffic management, agriculture, surveillance and security, delivery services, and search and rescue, etc.) due to advancements in technology that have made them inexpensive, easy to deploy, and operate in hazardous or inaccessible environments. However, despite the benefits of UAVs, there are some limitations that can affect their operational effectiveness. The current IP architecture is identified as one of these limitations [3].

The current IP architecture was originally designed for communication among fixed devices on the Internet, and it focuses on the location of the devices rather than the data being transmitted. However, in recent years, it has become evident that the current IP architecture is not as well-suited to the demands of the modern Internet [4], which includes heterogeneous communication and a wide variety of highly mobile devices and vehicles such as satellites and UAVs, which have different requirements and characteristics compared to traditional fixed devices. In addition, many modern aircraft, including UAVs, are often employed to gather sensitive data for surveillance, research, and monitoring purposes. Collecting such data requires robust security measures to safeguard against security attacks and ensure their confidentiality and integrity. To this end, several overlay patches have been implemented to address security and mobility issues. Nevertheless, the IP architecture still faces challenges in providing seamless and efficient mobility support for all types of devices and applications. Moreover, overlay patches might not be sufficient for protecting against emerging threats [5]. This hinders the safe and secure use of UAVs for sensitive applications like military operations or privacy-sensitive contexts.

In the last decade, the NDN architecture [6] has gained much research attention. It has been proposed as one of the promising future Internet architectures to overcome the current vulnerabilities and cope with the challenges [7] (e.g., addressing, name resolution, security, and privacy) of the IP Internet architecture. NDN is an information-centric networking architecture that changes the communication semantics by focusing on retrieving data by a given unique name rather than location address. Thanks to the built-in security, multi-path forwarding, consumer-driven communication, in-network caching, and name-based routing features, NDN has proven to be a suitable information-centric networking architecture choice for UAV communication [8,9].

As a result of the built-in features of NDN, the mobility of devices requesting data (consumers) is inherently supported, thanks to its consumer-driven communication feature [10], where consumers only need to reissue requests from their new location without updating their locations or establishing a new connection. In contrast, producer (devices generating data) mobility management remains one of the significant challenges due to NDN's content-centric design, where data retrieval is based on content names rather than the location of devices. Accordingly, when the producer changes their Point of Attachment (PoA), they may become unreachable for a while until the relevant routers in the network update their information to the producers' new location. Furthermore, UAVs, as producers of data, introduce additional challenges for mobility management due to their real-time communication requirements, the ineffective NDN in-network caching feature to support low-latency data, and the UAVs' ability to move quickly and unpredictably, leading to frequent handoffs, which may result in increased latency and packet loss.

Many strategies have been proposed in the literature to address the producer mobility issue, which can be divided into two broad approaches: Anchor-less [11–16] and Anchor-based [17–21]. In the Anchor-based approach, a fixed network node is used as a Domain Name System (DNS) server, a mapping server, or a rendezvous node to redirect packets toward the Mobile Producer (MP) or as a depot to store the producer's data before a handoff occurs. In contrast, in the Anchor-less approach, producer mobility is generally supported using updates of the forwarding information base table in the routers either between producers and consumers or between the producer's new and old PoA after handoff occurs.

This paper aims to design a strategy for supporting UAV mobility and ensuring seamless low-latency data delivery in NDN architecture. Table 1 lists the key abbreviations used in this article with their definitions.

Table 1. List of abbreviations used throughout the paper.

Acronyms	Definition
NDN	Named Data Networking
IP	Internet Protocol
UAV	Unmanned Aerial Vehicle
DMMS	Distributed Mobility Management Scheme
PoA	Point of Attachment
DNS	Domain Name System
MP	Mobile Producer
Inpt	Interest Packet
PIT	Pending Interest Table
FIB	Forwarding Information Base
CS	Content Store
AS	Autonomous System

Table 1. *Cont.*

Acronyms	Definition
RVS	Rendezvous Server
RP	RemovePath
CP	CreatePath
CPD	CreatePathData
RSS	Received Signal Strength
KF	Kalman Filter
GM	Gauss–Markov
WMN	Wireless Mesh Network

The main contributions of this work are as follows:

(a) We propose a low-overhead strategy based on strategically deployed decentralized cooperative Anchors, which proactively forwards the consumer's packet toward producers when handoff occurs to achieve seamless data delivery with minimum packet loss and delay.

(b) We introduce a new forwarding strategy that combines the NDN's default forwarding and the location-based forwarding strategy.

(c) A Kalman filter method is used to predict the UAV's location coordinates after the handoff.

(d) Using a realistic scenario, we implemented and evaluated the performance of the DMMS against two well-known solutions, Kite [17] and MAP-ME [13].

This paper is structured as follows. Section 2 provides a concise overview of the NDN architecture, its forwarding pipeline, the associated mobility issues, and a review of the pertinent literature. In Section 3, the design and functionalities of DMMS are detailed. Section 4 offers an evaluation of our proposed solution, examining its performance and effectiveness. Lastly, Section 5 concludes the paper and outlines potential avenues for future research in this domain.

2. NDN Background and Related Work

2.1. Named Data Networking Architecture

The communication philosophy of NDN is fundamentally based on three types of entities: consumers, producers, and routers. When a consumer wants to request content, it sends an Interest Packet (Inpt) containing the content's name. This Inpt is routed through the network towards the data producer, which responds with a Data packet containing the requested content. Each router in the NDN maintains three crucial tables: the Pending Interest Table (PIT), Forwarding Information Base (FIB), and Content Store (CS). These tables work together to optimize content delivery in NDN by efficiently managing the Inpt and Data packet within the network. Once receiving an Inpt, a router first checks its CS (a cache that stores data requested before) to see if it already has a copy of the requested data. If a match is found in the CS, the router immediately satisfies the received Inpt. If the CS check fails to find the desired data, implying that no cached copy of the required content exists, the router then performs a lookup on the PIT (which keeps track of all the Inpts that have been forwarded but have not yet been satisfied) to verify whether it has an entry for the received Inpt. If an entry is found in the PIT, the router will add the received interface ID to the PIT entry, and the Inpt will be dropped. Otherwise, the router then performs a name-based longest prefix match lookup in its FIB (which contains the forwarding information for each content name) to determine the interface to forward the Inpt toward the producer. If a match is found for the content name in one of the FIB entries, the router then creates a new PIT entry for the Inpt using the received interface ID and Inpt's content name, and the Inpt is forwarded upstream. This process is repeated until the Inpt reaches the producer that replies with a Data packet, which follows the reverse path stored in the PIT entries across the routers, taken by the Inpt, back to the consumer. In the case when no longest prefix match is found, then the Inpt is forwarded according to the forwarding strategy used.

2.2. Mobility in Named Data Networking

In NDN, the mobility of UAVs can be classified based on their applications into two distinct types: producer and consumer mobility. The latter is more easily managed due to inherent features of the NDN architecture design, like in-network caching, multipath forwarding, and consumer-driven communication. For instance, when a consumer changes its PoA, it can resend Inpts from its new location without updating the location or establishing a new connection. This Inpt can be satisfied either by intermediate routers that have previously cached the data or by the producer. On the other hand, the NDN architecture faces difficulties in effectively supporting producer mobility. This is mainly because Inpts are forwarded based on the content's name, not the producer's location. When a producer moves to a new PoA, consumer Inpts are initially sent toward the producer's previous location due to the outdated FIB of the relevant routers. As a result, the Inpts cannot reach the newly moved producer until these routers' FIBs are updated to reflect the producer's new location. The time taken to update all relevant FIBs can cause significant delay in communication and increase packet loss, which is unacceptable in scenarios that require low-latency communication.

2.3. Related Work

Several survey studies [22–25] have been conducted to categorize the producer mobility support strategies into different approaches based on the technique used and their functions and features. However, these strategies can be generally classified into two broad distinct categories: Anchor-less and Anchor-based approaches.

2.3.1. Anchor-Less Approach

In the Anchor-less approach, managing producer mobility is achieved without the assistance of a specific node in the network. Various strategies are proposed to ensure the effective delivery of Inpts to an MP. These strategies may involve dynamically updating the FIB tables after a producer's handoff, caching the producer's content on a neighboring router before the producer's handoff, or employing an optimal broadcast to reach the MP.

Hussaini et al. [11] proposed a solution that involves the MP broadcasting a special Inpt (mobility Interest) throughout its new location. This Inpt carries the producer's location prefix information, which is used to update the FIB tables of routers within the producer's domain. Additionally, if the producer and consumers are located in different domains, the mobility Interest message is also used to update the FIB tables of routers in neighboring domains.

The scheme outlined in [12] introduces a forwarding strategy designed to support producer mobility within NDN-IoT. In this scheme, each PoA continuously sends periodic Inpts to monitor the status of all attached sensors. When the PoA does not receive a response from a sensor, it sends a Recovery packet to remove the forwarding information associated with this sensor from the routers along the path between the producer and the consumer. Upon detecting the connection of a new sensor, the PoA sends a Recovery packet with the aim of creating a new FIB entry leading to this sensor. This Recovery packet is forwarded to the consumer using the longest prefix match method.

Augé et al. [13,14] proposed "MAP-ME", a solution for supporting producer mobility in real-time communication by dynamically updating the FIB table in minimal routers in the network. In [13], when the MP changes the PoA, it sends a special update Inpt from the new PoA to the old PoA to update the router's FIB along the path between the new and old PoA. A new variant of the MAP-ME scheme was proposed in [14] to address the producer micro-mobility, which includes an additional protocol named "Notification/Discovery" to alleviate the issues of Inpt update dropping or late updating of the FIB due to network latency, which can be lead to an increase in packet loss. An Inpt notification is left by the producer at the old PoA to redirect the future consumer's Inpts through a broadcasting method until they meet the path toward the producer. A significant shortcoming of MAP-ME is it suffers from a triangular routing problem. It is worth noting that MAP-ME is related to topology-based methods.

The scheme proposed in [15] focuses on supporting producer mobility in NDN by leveraging the in-network caching feature. The main idea of this scheme is to predict the next location of an MP and proactively cache the most frequently requested data in the router nearest to the consumer before the producer's handover occurs.

The work described in [16] presents a scheme designed to reduce the delay and minimize packet loss caused by producer mobility in real-time remote health-monitoring systems based on NDN. The main idea of the proposed scheme is to register the MP in both the current PoA and the nearest PoA. In this way, the MP establishes connections and maintains a presence in the network even when moving between different PoAs. However, it is worth noting that inter-AS mobility brings a large signaling overhead since the proposed scheme uses a broadcasting forwarding strategy to locate a producer each time it crosses between autonomous systems. Furthermore, registering a large number of MPs with multiple PoAs can lead to a significant increase in the size of FIB tables at PoAs.

2.3.2. Anchor-Based Approach

To support producer mobility, a solution proposed by the NDN Research Labs [17], denoted as "Kite", involves using a Rendezvous Server (RVS) as an intermediary to facilitate communication between consumers and producers. This strategy involves publishing the producer's content under a reachable RVS prefix. When the producer changes PoA, it sends a Trace Inpt (TI) to the RVS. In response, a Trace Data Packet (TD) is sent back to the producer, which follows the same path as the TI. When intermediate routers receive the TD, they create a FIB entry for the MP using the data name extracted from the TD and the face from which the TD is received to create a hop-by-hop path between the MP and the RVS. Thus, when the consumer wants to fetch data from the producer, it sends Inpt initially to the RVS. Subsequently, the RVS forwards this Inpt towards the MP along the path that was created earlier during the handoff process. However, the Kite scheme introduces two main drawbacks. Firstly, it brings a large signaling overhead due to the necessity of maintaining the path to the MP in the Trace FIB table. Secondly, frequent handoffs by the MP can lead to the "stale path" problem, where outdated path information results in increased packet loss.

The authors of [18] proposed a redirection-based strategy to support content source mobility in Content-Centric Networking (CCN) to reduce routing overhead caused by updates to the router's FIB. When a content source moves to a new PoA, it sends a special Inpt to its home domain router, which acts as a home agent in the context of mobile IP. This Inpt contains the prefix advertised by the new PoA. Subsequently, when the home domain router receives a consumer's Inpt, it encapsulates it within a new Inpt using the prefix of the new PoA as the content name. This encapsulated Inpt is then forwarded to the content source based on the content name associated with the new PoA. However, this strategy suffers from the path stretch problem, as all Inpts must pass through the content source's home domain to reach it, potentially increasing latency. Additionally, Inpt encapsulation introduces additional complexity to the CCN protocol, making it more challenging to implement and maintain.

Kim et al. [19] presented an on-demand Anchor strategy to support producer mobility. In this strategy, when the MP switches to a new PoA, it sends an Inpt (mobility update) containing its content name to an immobile Anchor router, establishing a trace path with the Anchor. As the consumer's Inpts reach the previous PoA of the MP, rather than being dropped, they are redirected to the Anchor router. Subsequently, the Anchor router forwards the Inpts toward the producer's new location. However, this strategy does not specify Anchor placement in the network, potentially leading to path stretch due to the consumer's Inpt redirection through the Anchors.

The work described in [20] proposes a pulling-based strategy, denoted "ProPull", to support producer mobility for real-time content delivery. This strategy aims to provide the best possible quality of service for consumers by ensuring uninterrupted multi-media transmission during producer mobility while minimizing signaling overhead costs. The ProPull strategy achieves this by pulling Inpts from the producer's old PoA to the new PoA. When the producer predicts a handover, it sends a special Inpt to the current PoA to obtain its prefix. Once connected to the new PoA, the producer communicates with the old PoA by exchanging special messages called Propaganda Inpts and Propaganda Data packets. This communication establishes a temporary forwarding path between the old and new PoA, ensuring that consumer Inpts are pulled efficiently to the producer's new location. However, the ProPull strategy introduces high signaling overhead due to exchanging Propaganda Inpts and Propaganda Data packets during each producer relocation.

To reduce handover latency in real-time communication scenarios, a scheme proposed in [21] utilizes the producer's future location prediction to ensure seamless producer mobility. When a producer detects a handoff event by monitoring the signal strength from the current PoA, it sends a special Inpt to the current PoA, which then predicts the best PoA candidate the producer is expected to connect to after the handoff. Once the handoff is completed and the producer successfully connects to the new PoA, the consumer's Inpt is redirected toward the new PoA from the old PoA. However, using a redirection strategy can result in path stretching, leading to increased latency. Furthermore, when the prediction of a producer's future location goes wrong, the new PoA broadcasts the producer's reachable prefix throughout the network, resulting in significant signaling overhead and congestion.

3. Proposed Solution

In this section, we introduce the DMMS, designed to minimize the impacts of the producer's mobility in content delivery in terms of delay and packet loss using minimal signaling overhead. Our strategy relies on three components:

A distributed cooperative strategically deployed Anchor, which proactively forwards the consumers' packet toward MPs when handoff occurs to achieve seamless data delivery with minimum packet loss and delay.

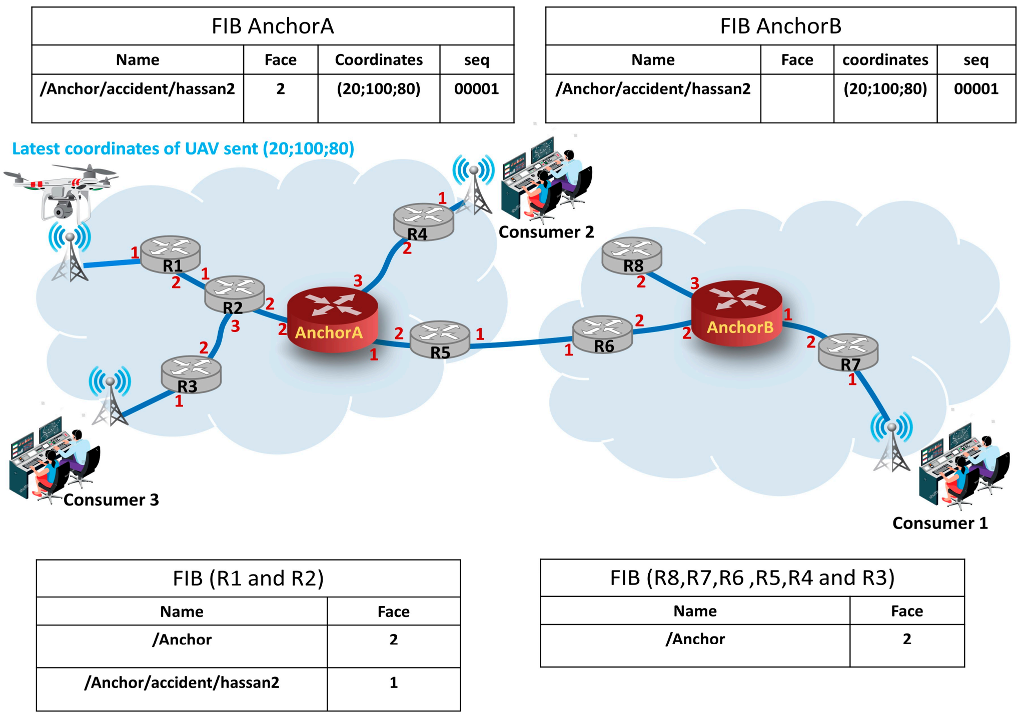
A forwarding strategy that combines the NDN's default forwarding and the location-based forwarding strategy for improving data delivery.

A location prediction technique to anticipate the producer's future location coordinates before a handoff event occurs, which allows for proactive forwarding of consumers' Inpts through the Anchor to the coordinates of the next PoA, where the producer is expected to be connected after the handoff.

We separately detail in this section the Distributed Anchor design, the forwarding strategy, and the producer's location prediction technique.

3.1. Distributed Anchor Design

In the DMMS, distributed synchronized Anchors are strategically placed throughout the network. These Anchors act as rendezvous nodes between consumers and producers, where the Anchor redirects consumers' Inpts toward the producer when a direct path is not found toward the producer with minimal path stretch. Each Anchor operates as a regular NDN router with the same functionalities and maintains the same structure with a slight modification to the FIB entries. This modification includes two additional fields, namely "coordinates" and "seq". The "coordinates" field is used to store the coordinates of the producer's location or the predicted one, while the "seq" field is a sequence number used to maintain the consistency of the producer's coordinate information among Anchors and prevent forwarding loops during producer's notification message propagation (e.g., AnchorA's FIB in Figure 1).



FIB AnchorA			
Name	**Face**	**Coordinates**	**seq**
/Anchor/accident/hassan2	2	(20;100;80)	00001

FIB AnchorB			
Name	**Face**	**coordinates**	**seq**
/Anchor/accident/hassan2		(20;100;80)	00001

FIB (R1 and R2)	
Name	**Face**
/Anchor	2
/Anchor/accident/hassan2	1

FIB (R8,R7,R6 ,R5,R4 and R3)	
Name	**Face**
/Anchor	2

Figure 1. Anchor-based distributed mobility management architecture.

Regarding the inconsistencies in the producer's coordinates among Anchors, they can be shared in real time by implementing synchronization protocols designed for distributed dataset synchronization in NDN, such as ChronoSync [26], VectorSync [27], or PSync [28]. Alternatively, a producer's handoff-driven mechanism is used to minimize unnecessary network traffic. When a producer handoff event occurs, it sends a special Inpt with a sequence number that serves as a unique identifier for the update, incremented with each handoff. This Inpt is then propagated to Anchors in the network to ensure that updates are applied in the correct order.

Like the IP anycast mechanism, the Anchors in the DMMS share and advertise the same routing prefix through the routing protocol within the network, which allows consumers to efficiently discover and connect with the nearest Anchor, regardless of their specific location within the network. Furthermore, to ensure the reachability of MP in the network, each MP is assigned a unique hierarchical name for its data, which uses the same Anchor prefix.

We present in Figure 1 a comprehensive architecture concept. After an MP connects to a new PoA, it establishes a path between itself and the nearest Anchor. To request data from this MP, the consumer sends an Inpt containing the MP data name (e.g., in Figure 1, the producer data name is "Anchor/accident/hassan2"). Once an intermediate router receives the Inpt, it checks its FIB to determine the Face with the longest prefix match. If a matched FIB entry to this Inpt is found, then the Inpt will be forwarded to the MP without the intervention of the Anchor (e.g., in Figure 1, the Inpts of consumer 3 will be forwarded along R2 and R1 without reaching the Anchor). If a matched FIB entry does not exist, then the Inpt will be forwarded to the nearest Anchor. The Anchor then checks its FIB to determine the strategy to forward the Inpt, either using the NDN's default forwarding (e.g., in Figure 1, the Inpts of consumer 2 at AnchorA will be forwarded along the path AnchorA $\longrightarrow$ R2 $\longrightarrow$ R1 $\longrightarrow$ PoA $\longrightarrow$ producer) or using the location-based forwarding strategy (e.g., in Figure 1, the Inpts of consumer 2 at AnchorB will be forwarded using location-based forwarding).

In fact, using Anchors in the network has a significant impact on increasing the path stretching between consumers and MPs compared to an Anchor-less approach. However, strategically positioned Anchors can effectively reduce the number of hops that the Inpts need to traverse to reach the MP, thereby minimizing latency and improving content delivery efficiency. The problem of selecting Anchor placements that can manage the producer mobility in a dense network can be viewed as a variant of the well-known k-center or p-median problems, which are optimization problems that involve selecting a subset of locations from a larger set of potential locations. However, solving this problem is challenging; as the network size grows, the number of possible Anchor placements increases exponentially, meaning that finding an optimal solution in polynomial time is unlikely, making it an NP-hard problem.

As a result, most researchers have focused on developing heuristic algorithms [29] and simulation tools [30] that can find near-optimal solutions in a reasonable amount of time. To determine the optimal Anchor placements that minimize the path stretch, we propose a method that involves partitioning the network into clusters based on network topology and selecting a router or Access point as an Anchor for each cluster based on its proximity to all routers and Access points within the cluster and its closeness to Anchors in other clusters to minimize intra-cluster and inter-cluster communication distances.

There are numerous clustering algorithms that can be applied to partitioning the network. Each algorithm has its own strengths and weaknesses, and some do not guarantee convergence to the optimal solution. A comprehensive survey of clustering algorithms can be found in [31]. Our proposed method uses an iterative approach based on a simple and efficient clustering algorithm, namely the K-Medoids algorithm [32]. It starts by selecting a k router as an initial cluster center router (medoids). The quality of this initial cluster is iteratively improved through a local search process based on minimizing both the distance between any cluster's router and medoid and the distance between medoids. During each iteration, center routers are replaced by non-center ones, a process that continues until the intra-cluster and inter-cluster distances are reduced. However, the quality of the final clustering solution of our method may not usually be optimal since it is sensitive to the initial selection of the center cluster. Therefore, our method is executed multiple times with different random initialization. We then choose the optimal solution as Anchor placements. It is worth noting that determining the optimal number of Anchors for a given network is a complex problem that is out of the scope of this paper.

3.2. Forwarding Strategy

In this paper, we introduce a new forwarding approach that combines the NDN's default forwarding strategy and the location-based forwarding strategy. As we have already mentioned, the data retrieval in NDN is based on content names without any location information of the device, resulting in the inability to handle producer mobility. That is why we propose integrating the location-based forwarding strategy, which can coexist with the NDN's default forwarding strategy to maintain uninterrupted content delivery to consumers. When the MP initiates a handoff, it sends a RemovePath (RP) packet to notify the nearest Anchor about the predicted coordinates of its next location where it will be present after completing the handoff. Consumers remain unaware of the MP's location change and send Inpts using NDN's default forwarding strategy. Once an intermediate router receives the Inpt, it checks its FIB to determine the Face with the longest prefix match. If a matched FIB entry does not exist, the Inpt will be forwarded until it reaches the nearest Anchor. Subsequently, the Anchor checks its FIB to determine the strategy to forward this Inpt. If the path has been created between the MP and this Anchor before the reception of the Inpt, then the Anchor and all intermediate routers along the path use the NDN's default forwarding strategy. Otherwise, the Anchor updates the X, Y, and Z coordinates in the Inpt with the producer's predicted location coordinates and forward it using the location-based forwarding strategy.

Leveraging the location information of fixed routers and Access points, each router in the location-based forwarding strategy determines the next hop for the Inpt based on the Cartesian coordinates of its neighbors, where it calculates the Euclidean distance between its neighbors' coordinates and the destination coordinates included in the Inpt and it forwards the packet to the nearest router to the destination coordinates, ensuring the shortest path to the MP.

To implement the location-based forwarding strategy without modifying the Inpt structure, we use the "GeoTag" header field within the Inpt, introduced in the second version of the NDN Link Adaptation Protocol [33]; this predefined 3-byte field is specifically designated for including some location information in Inpt. We assume that routers and Access points involved in the network are stationary and possess prior knowledge of each other's Cartesian locations.

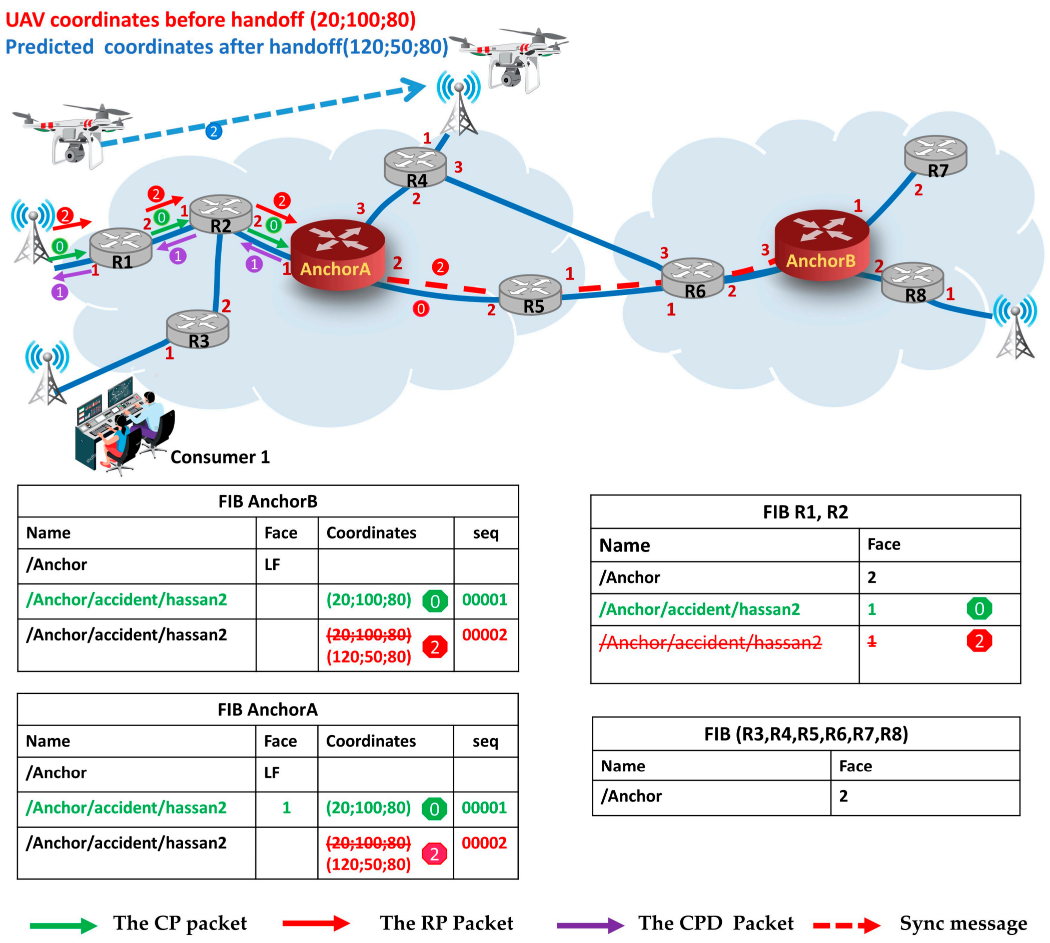
3.2.1. CreatePath and RemovePath Packet Forwarding

To efficiently implement our mobility management scheme, we use two types of Inpts:

(1) The CreatePath (CP) packet is an Inpt used to announce the producer's connection to the network, which includes a Cartesian coordinate representing the producer's location. Its purpose is to establish a path between the producer and Anchor and to notify the Anchors about the producer's location. This packet is specifically sent to the nearest Anchor in two scenarios:

When a new producer connects to the network for the first time and when the MP cannot receive the Inpts after the handoff due to the failure to predict its destination coordinates. Like a regular Inpt structure, the CP packet uses a content name in the form of "Anchorprefix/CreatePath/producerDataName", where the "Anchorprefix" refers to the prefix advertised by the Anchors, "producerDataName" represents the name of the producer's data, while "CreatePath" serves as a special identifier that allows all routers to identify the packet as a request to establish the path between producer and an Anchor and handle it accordingly. Upon receiving the CP packet (step 0 in Figure 2), an intermediate router (e.g., R2 in Figure 2) creates a FIB entry using the incoming Face through which the CP packet arrives at the router and the extracted name from the CP packet name by removing the "CreatePath" identifier. This ensures that subsequent Inpts from consumers can be routed along the established path towards the producer without the intervention of an Anchor, particularly where the path between the MP and consumer is stretched to the Anchor (Inpts of consumer 1 in Figure 2). To avoid failures in establishing the path between the MP and Anchor, a CreatePathData (CPD) packet (step 1 in Figure 2) is used as an acknowledgment from the Anchor to the MP, confirming the receipt of the CP packet.

(2) The RemovePath Packet is an Inpt that serves a dual purpose: (1) updating the FIB entry associated with the producer's data name in the Anchor with the producer's predicted location; (2) removing the outdated FIB entry in the routers along the path between the producer and the nearest Anchor that may have been created before. Like the CP packet, the RP packet uses the same content name format, "Anchor prefix/RemovePath/producerDataName", and includes the Cartesian coordinates representing the producer's predicted location where the MP will be present after handoff. When an MP starts moving from one Access point to another, it senses the degradation of the Received Signal Strength (RSS) quality in the current channel. As the MP moves out of range from the current PoA, the RSS quality continues to degrade. Once the RSS quality reaches a preset threshold (e.g., -75 dBm), before initiating the process of selecting a new PoA, the MP sends the RP packet (step 2 in Figure 2) towards the nearest Anchor which contains the predicted coordinates of the next location. Upon receiving the RP packet, each router (e.g., R2 in Figure 2) along the path deletes the FIB entry associated with the MP and forwards the packet towards the Anchor. Once the Anchor receives the RP packet, it updates the FIB entry of the MP using the predicted coordinates included in the RP packet and propagates this information to Anchors.



FIB AnchorB			
Name	Face	Coordinates	seq
/Anchor	LF		
/Anchor/accident/hassan2		(20;100;80) 0	00001
/Anchor/accident/hassan2		~~(20;100;80)~~ (120;50;80) 2	00002

FIB AnchorA			
Name	Face	Coordinates	seq
/Anchor	LF		
/Anchor/accident/hassan2	1	(20;100;80) 0	00001
/Anchor/accident/hassan2		~~(20;100;80)~~ (120;50;80) 2	00002

FIB R1, R2		
Name	Face	
/Anchor	2	
/Anchor/accident/hassan2	1	0
~~/Anchor/accident/hassan2~~	~~1~~	2

FIB (R3,R4,R5,R6,R7,R8)	
Name	Face
/Anchor	2

Figure 2. Mobility management scenario: The MP sends the CP packet when it connects to the network for the first time (step 0). The MP send the RP packet prior to handoff (step 2).

3.2.2. Consumer Interest and Producer Data Packet Forwarding

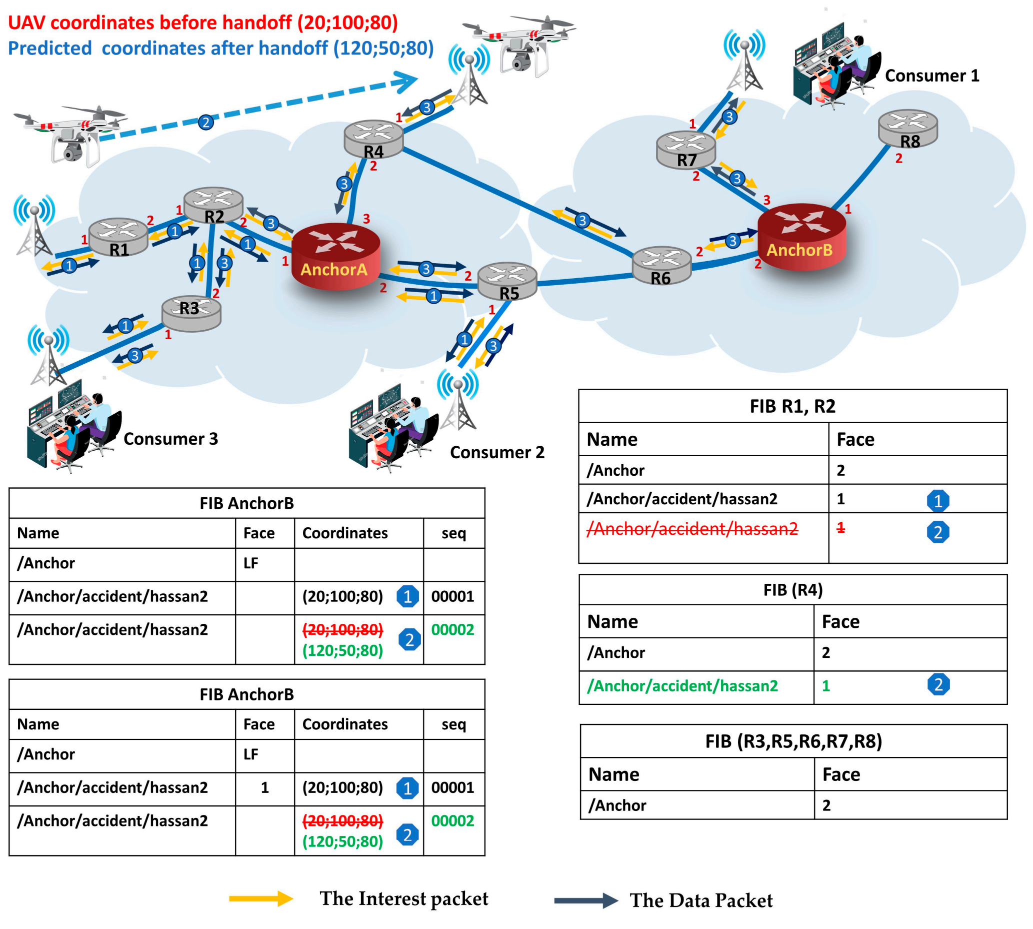
(1) Consumer Inpt Forwarding: When a consumer wants to request content, it sends an Inpt that typically includes the name of the desired content. In the case of our proposal, the Inpt also includes the X, Y, and Z coordinates, initialized with negative values by default. Upon receiving the Inpt, each intermediate router checks if the Inpt has non-negative coordinates. If so, the router forwards the Inpt using the location-based forwarding strategy. If not, the intermediate router performs a name-based longest prefix match lookup in its FIB to compare the Inpt content's name with the content names stored in the FIB entries. According to this comparison:

If a match is found for the content name in one of the FIB entries, then the router determines the next-hop interface based on the information stored in the FIB entry, allowing the Inpt to be forwarded directly towards the producer without involving the Anchor. For instance, in Figure 3, the path of the Inpts of consumer 3 (steps 1) to the producer before handoff is: consumer 3 $\longrightarrow$ PoA $\longrightarrow$ R3 $\longrightarrow$ R2 $\longrightarrow$R1 $\longrightarrow$ PoA $\longrightarrow$ producer.

If no matching is found for the content name in the FIB entries, in this case, the longest matching prefix will be the prefix of the Anchor, and the Inpt will then be forwarded toward the nearest Anchor using the appropriate interface (e.g., Inpts of consumer 2 (steps 1 and 3) in Figure 3). When the Anchor receives the consumer's Inpt, it checks its FIB entry associated with the producer's data name. This entry contains two important pieces of information: (1) producer's location: this includes the X, Y, and Z coordinates of the producer retrieved previously through the CP or the predicted location coordinates after handoff updated by the RP packets; (2) Outgoing Interfaces: this information specifies the interfaces through which the consumer's Inpt should be forwarded. If an Outgoing

Interface is found, meaning that the path between the MP and Anchor has been created before the reception of the Inpt, then the Anchor and all intermediate routers along the path use the appropriate interface. If no Outgoing Interface is found, the location-based forwarding strategy is used to determine the appropriate interface. In this case, the X, Y, and Z of the Inpt coordinates are updated with the producer's location coordinates stored in the FIB entry, and each intermediate router along the path between the Anchor and the MP uses the location-based forwarding strategy to select the next Face to forward this Inpt (e.g., in Figure 4, the path of the Inpts of consumer 1 to the producer after AnchorB is AnchorB $\longrightarrow$ R6 $\longrightarrow$ R4 $\longrightarrow$ PoA $\longrightarrow$ producer). The forwarding process of the consumer's Inpt is detailed in Algorithm 1.

(2) Producer data packet forwarding: when the producer receives the consumer's Inpt, it responds by sending a Data packet that follows the same path as the Inpt in reverse. Each intermediate router along the path uses the information stored in its PIT to forward the Data packet back to the consumer. Additionally, to reduce the computational burden of calculating distances for every Inpt in location-based forwarding, each intermediate router along the path between the producer and the Anchor creates a FIB entry upon receiving the producer Data packet. This FIB entry includes the name of the MP's data and the incoming Face from which the Data packet was received (e.g., in step 3, a FIB entry for the producer's content name is added to R4 after handoff, shown in Figure 4). By doing so, the router establishes a path toward the producer for future consumer Interest forwarding without relying on the involvement of the Anchor, particularly, in the worst-case scenarios, where the path between the MP and consumer is stretched to the Anchor. The forwarding process of the Data packet is detailed in Algorithm 2.



FIB AnchorB

Name	Face	Coordinates		seq
/Anchor	LF			
/Anchor/accident/hassan2		(20;100;80)	1	00001
/Anchor/accident/hassan2		~~(20;100;80)~~ (120;50;80)	2	00002

FIB AnchorB

Name	Face	Coordinates		seq
/Anchor	LF			
/Anchor/accident/hassan2	1	(20;100;80)	1	00001
/Anchor/accident/hassan2		~~(20;100;80)~~ (120;50;80)	2	00002

FIB R1, R2

Name	Face	
/Anchor	2	
/Anchor/accident/hassan2	1	1
~~/Anchor/accident/hassan2~~	~~1~~	2

FIB (R4)

Name	Face	
/Anchor	2	
/Anchor/accident/hassan2	1	2

FIB (R3,R5,R6,R7,R8)

Name	Face
/Anchor	2

Figure 3. Example of the forwarding of Interest and Data packets.

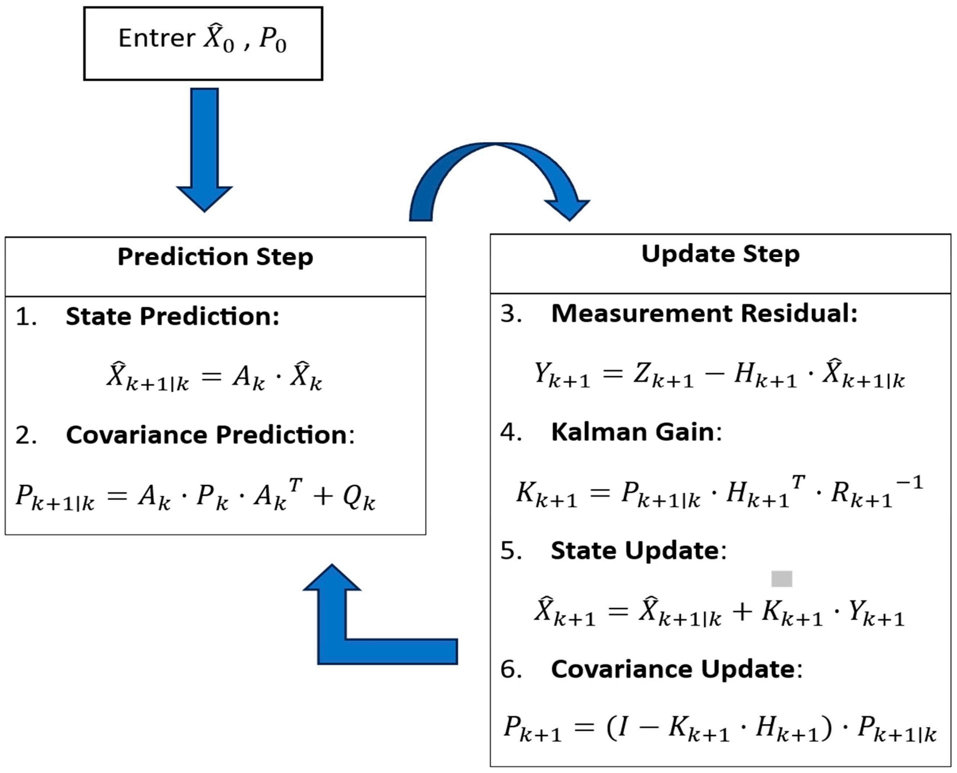

Entrer $\hat{X}_0$, P_0
Prediction Step
1. State Prediction:
$\hat{X}_{k+1|k} = A_k \cdot \hat{X}_k$
2. Covariance Prediction:
$P_{k+1|k} = A_k \cdot P_k \cdot {A_k}^T + Q_k$
Update Step
3. Measurement Residual:
$Y_{k+1} = Z_{k+1} - H_{k+1} \cdot \hat{X}_{k+1|k}$
4. Kalman Gain:
$K_{k+1} = P_{k+1|k} \cdot {H_{k+1}}^T \cdot {R_{k+1}}^{-1}$
5. State Update:
$\hat{X}_{k+1} = \hat{X}_{k+1|k} + K_{k+1} \cdot Y_{k+1}$
6. Covariance Update:
$P_{k+1} = (I - K_{k+1} \cdot H_{k+1}) \cdot P_{k+1|k}$

Figure 4. Pipeline of the prediction–update procedure of the KF.

Algorithm 1 Consumer_InterestForward(Interest *I*, incomingface *F*)

```
1: Regular CS and PIT lookup
2: if this.IsAnchor then
3:      Entry ← FIB.GetMatchedPrefix (I.name)
4:       if Entry ≠ ø and Entry. HasFace then
5:            SendToNeighbor(I, $f_i$)
6:       else
7:            AddcoordinateToInterest (I)
8:            for $N_i$ in Neighbors do
9:                 $Dist_i = \sqrt{(x_{N_i} - x_{Inter})^2 + (y_{N_i} - y_{Inter})^2 + (z_{N_i} - z_{Inter})^2}$
10:               end for
11:                    $CloseNei = min(Dist_i)$
12:                    SendToNeighbor ( $I, CloseNei.face$)
13: else
14: if I. HaspositifCoordinates and I.IsNoteCreate_Nor_RemovePath then
15:      for $N_i$ in Neighbors do
16:       if Entry ≠ ø and I. HasNegatifCoordinates then
17:                 $Dist_i = \sqrt{(x_{N_i} - x_{Inter})^2 + (y_{N_i} - y_{Inter})^2 + (z_{N_i} - z_{Inter})^2}$
18:       end for
19:                 $CloseNei = min(Dist_i)$
20:                 SendToNeighbor ( $I, CloseNei.face$)
20: else
21:    Entry ← FIB.LongestPrefixMatch(I.name)
22:    for $f_i$ in Entry. NextHop do
23:       SendToNeighbor(I, $f_i$)
24:    end for
25: return
```

Algorithm 2 OnIncomingData (Data *D*, Incomingface *F*)

```
1: CheckValidity()
2: Interestpackets=PitEntry.getInterests ($D.name$)
3: if Interestpackets $\neq ø$ then
4:    for $I_i$ in Interestpackets do
5:        if $I_i$. HaspositifCoordinates () then
6:            FIB.addentry($D.name$,$F$)
7:        InterestIncomingFace=PitEntry. getInRecords( $I_i$)
8:        // go to outgoing Data pipeline
9:        OnOutgoingData(data, InterestIncomingFace)
10:   end for
11: return
```

3.3. Producer Location Prediction

The prior knowledge of the producer's new PoA coordinates plays a crucial role in ensuring seamless mobility. As part of our approach, we propose utilizing the producer's predicted coordinates of the next location where the MP will be present when the handoff is complete as destination coordinates to receive the consumer's Inpt. These predicted coordinates represent the nearest location to the producer's new PoA.

When a producer moves out of the range of its current PoA and needs to establish a connection with a different PoA, its network interface continuously monitors the RSS in the current channel. The RSS continues to degrade as the producer moves further away from the current PoA. In our strategy, when the RSS drops below a predefined threshold, the producer predicts the future position coordinates at the time T_h, representing the time required to complete the handoff process. Subsequently, the producer sends the RP packet, which includes the producer's predicted coordinates, to notify the Anchor.

Several methods have been proposed to predict the future location of a UAV. These prediction methods aim to estimate the UAV's position at a future time based on its current location and other relevant factors. Common approaches include trajectory extrapolation [34], sensor fusion [35], Kalman filtering [36], machine learning [37], and dynamic models [38]. In this paper, a Kalman Filter (KF) [39] method is used with the aid of Global Positioning System (GPS) and Inertial Measurement Unit (IMU) sensor technologies for predicting the UAV's future position coordinate.

The KF method is a popular method for predicting the future location of UAVs. It is a recursive algorithm that uses a state space model to estimate the UAV's current state and predict its future state based on the previous state and new measurements. This makes it more robust to noise and disturbances than other prediction methods, such as trajectory extrapolation. The KF operates through a two-step process: prediction step and update step, as shown in Figure 4. In the prediction step, the KF leverages the state transition model and current state (position and velocity) to predict the UAV's state (position) at the next time step. This prediction is associated with an uncertainty represented by the error covariance matrix. In the update step, the predicted state is compared with the actual measurements (obtained from a real-time GPS and IMU sensors) to compute the measurement residual. The filter then adjusts the state estimate and error covariance based on this residual and the measurement noise covariance. To predict the position of a UAV using the KF, we need to define the KF's components as follows:

1. State vector X_k
   The state vector of a UAV at moment k consists of its position and velocity, therefore, $\hat{X}$ is a vector of 6×1.

$$\hat{X}_k = [x_k,\ y_k,\ z_k,\ vx_k,\ vy_k,\ vz_k]^T \tag{1}$$

   where $[x_k,\ y_k,\ z_k,]^T$ are the estimated position coordinates and $[vx_k, vy_k,\ vz_k]^T$ are the estimated velocities along the x-axis, y-axis, and z-axis, respectively.

2. State transition matrix (A)
   Since the position of the UAV changes over time from one time step to the next step, the new coordinate in the next time step can be approximated by Equation (2).

$$\begin{cases} x_{k+1} = x_k + vx_k \times \Delta t \\ y_{k+1} = y_k + vy_k \times \Delta t \\ z_{k+1} = z_k + vz_k \times \Delta t \end{cases} \tag{2}$$

   The state transition matrix is typically defined as follows:

$$A = \begin{bmatrix} 1 & 0 & 0 & \Delta t & 0 & 0 \\ 0 & 1 & 0 & 0 & \Delta t & 0 \\ 0 & 0 & 1 & 0 & 0 & \Delta t \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \end{bmatrix}$$

   Δt represents the time interval between two consecutive time steps k and k + 1.
3. Measurement matrix H_t
   The measurement vector Z_t is a 3×1 vector representing the position measurements (x_t, y_t, z_t) obtained from a real-time GPS sensor. Consequently, the measurement matrix H_t is a 3×6 matrix that maps the state vector X_t to a 3-dimensional measurement vector Z_t. It is typically defined as follows:

$$H_t = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

4. The initial covariance matrix P_0
   The initial covariance matrix P_0 is a critical component of the KF, it represents the uncertainty in the initial state estimate. In the context of our case, where the initial state estimate is acquired from GPS and IMU sensors with high accuracy, the diagonal elements of P_0 are usually assigned small values, indicative of a high level of confidence in these estimates. Simultaneously, the off-diagonal elements are set to zero, assuming that the initial errors in different state variables are uncorrelated. The initial values for P_0 matrices are defined as follows:

$$P_0 = \begin{bmatrix} 0.01 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.01 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.01 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.01 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.01 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.01 \end{bmatrix}$$

5. The initial values for the process noise covariance Q and the measurement noise covariance R matrices are determined as follows:

$$Q = \begin{bmatrix} 0.001 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.001 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.001 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.001 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.001 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.001 \end{bmatrix} \quad R = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

UAV Mobility Model

The movement of UAVs differs significantly from that of common mobile vehicles, owing to their exposure to diverse external factors, including wind speed, turbulence, and environmental conditions. To accurately represent real-world UAV movements, the mobility model must reasonably emulate UAV movement. For that purpose, we propose to use the Gauss–Markov (GM) mobility model [40] as it closely emulates the uncertain and dynamic nature of real-world UAV movements by introducing randomness into the UAV's motion. In the GM mobility model, the UAV's velocity and direction are updated at fixed intervals of time based on their previous values using Gaussian Equation (3). The model employs average velocity, direction, and Gaussian random noise to introduce randomness and emulate the uncertain nature of UAV motion. The degree of dependency on past velocity and direction is controlled by a tuning parameter α, which affects the persistence and smoothness of the UAV's trajectory. A higher value of α indicates a stronger dependence on past values, leading to more persistent movement patterns, while a lower value results in a more dynamic and rapidly changing trajectory. In the GM mobility model, the cycle of updates for the UAV's velocity and direction is repeated throughout the simulation. The new values for these parameters are calculated as follows:

$$\begin{cases} V_n = \alpha V_{n-1} + (1+\alpha)\overline{V} + \sqrt{(1-\alpha^2)} \ * V_{x_n-1} \\ d_n = \alpha d_{n-1} + (1+\alpha)\overline{d} + \sqrt{(1-\ \alpha^2)} \ * d_{x_n-1} \end{cases} \tag{3}$$

where α is the tuning parameter, $\overline{V}$ and $\overline{d}$ are the mean of velocity and direction, while V_{x_n-1} and d_{x_n-1} are random variables from a Gaussian distribution that give some randomness to the new velocity and direction parameters.

The parameters of the Gauss–Markov model for our scenarios are listed in Table 2.

Table 2. Simulation parameters.

Category	Parameters	Value
Network	5 × 5 grid topology	100 × 100 m cell size
	Link delay	5 ms
	Data rate	10 Mbps
	Link layer protocol	IEEE 802.11
NDN	Interest frequency	10 Inpts/s
	Packet size	1024 bytes
	Numbers of Anchors	4
	Simulation duration	100 s
Simulation	Simulated repetitions	100 for each speed
	Prediction time step	0.1 s
Mobility (Gauss–Markov)	Alpha	0.85
	Mean velocity	Variable 1–30 m/s
	Mean direction	Uniform RV (min = 0, max = 6.28)
	Normal direction	Gaussian RV (mean = 0, var = 0.2, bound = 0.4)

4. Evaluation

This section presents the performance evaluation of the DMMS. A comparative analysis is conducted with two popular schemes from the literature: MAP-ME (representing an Anchor-less proposal) and Kite with multiple rendezvous servers, denoted as multi-RV KITE (representing an Anchor-based proposal), which are designed to address the challenges of real-time content delivery in mobile network environments. We have faithfully reproduced the source code of MAP-ME and multi-RV KITE to establish reliable benchmarks. To emulate the named data networking environment, we employ the ndnSIM [41] simulator, an open-source NS-3-based simulator widely adopted for named data network research.

We implemented the clustering algorithm using Python libraries. The resulting placements of Anchors are provided as input to our scenario for the DMMS and for the multi-RV KITE solution. It is worth noting that the author of the Kite solution did not provide explicit details regarding the method used for selecting the placements of rendezvous servers.

Subsequent sections provide detailed insights into the simulation environment configuration, the selection of performance metrics for evaluating both consumer experience and network cost, and an extensive discussion of the obtained results. A summarized overview of the simulation parameters can be found in the accompanying Table 2.

4.1. Simulation Setup

In order to evaluate the efficiency of the DMMS, we have employed a real-world topology to provide a rigorous assessment. Specifically, we have chosen to adopt a Wireless Mesh Network (WMN) installation scenario located within the city of Barcelona [42], as visually depicted in Figure 5. Within this context, our scenario was limited to 25 Access points as shown in Figure 6a within a 5 × 5 grid topology where each distinct grid cell dimension measures 100 × 100 m with one Access point, as shown in Figure 6b. All Access points operate in the 2.4 GHz radio frequency range and use the IEEE 802.11n access network. Notably, each grid node acts as both an Access point and router to enhance the mesh network's operational capabilities. In terms of wireless propagation, the channel models used combine the constant speed propagation delay model and the log-distance path loss model.

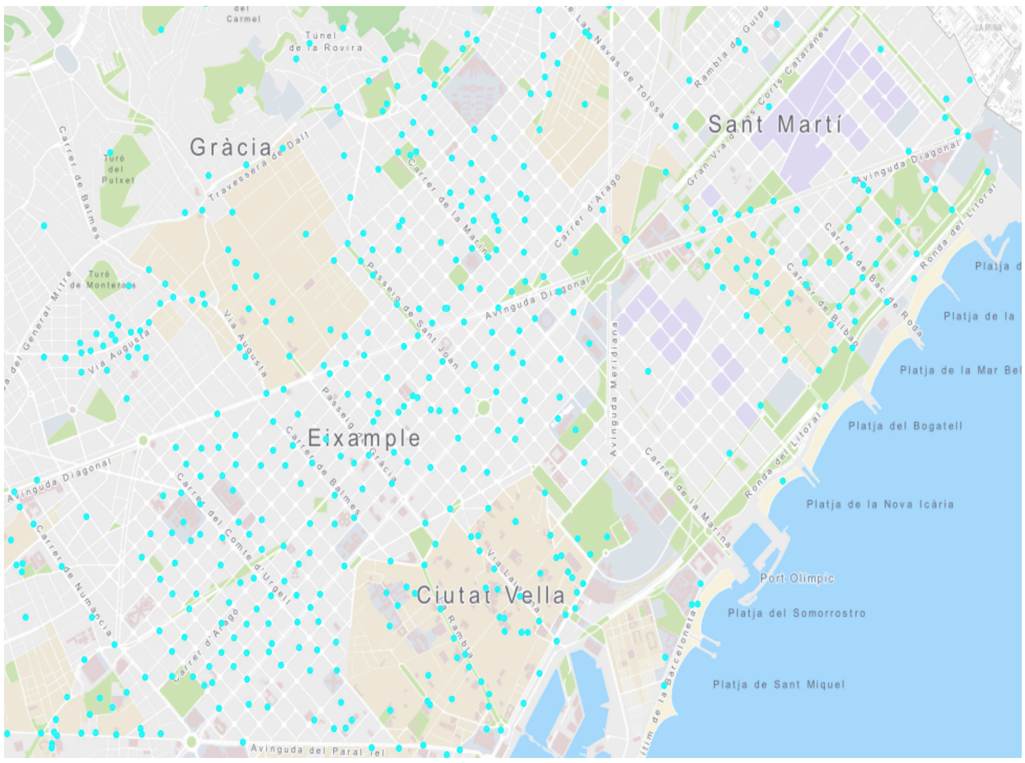


Figure 5. WMN topology of the city of Barcelona, Spain with Wi-Fi Access points (cyan dots).

In our scenario, we have on MP and one stationary consumer, with producer equipped with a streaming video application. Consumer continuously sends requests for data to an MP for 100 s with a frequency of five Inpts per second using a Constant Bit Rate (CBR) consumer application. Based on the clustering algorithm, Nodes 6, 8, 16, and 18 are designated as Anchors for the DMMS and rendezvous servers for multi-RV KITE. We set a 100 ms delay for updating the producer's coordinates (attachment information) on all Anchors in the DMMS and RVs in multi-RV KITE. The MP moves freely within the grid topology according to the GM mobility model, where it changes its velocity and direction in each epoch based on previous movements to emulate the flight pattern of a real-world UAV. We conducted 100 runs for each MP speed, with the MP's movement speed ranging from 1 m per second to 30 m per second. We implemented the KF algorithm using C++ for the producer's coordinate location prediction. The next location coordinate is predicted every ΔT = 500 ms using the same parameters as described in the previous section. Figure 7 shows the true trajectory of the MP at a speed of 30 m/s in blue and the estimated trajectory using the Kalman filter in red. The rectangle and triangle represent the start and end positions of the producer, respectively.

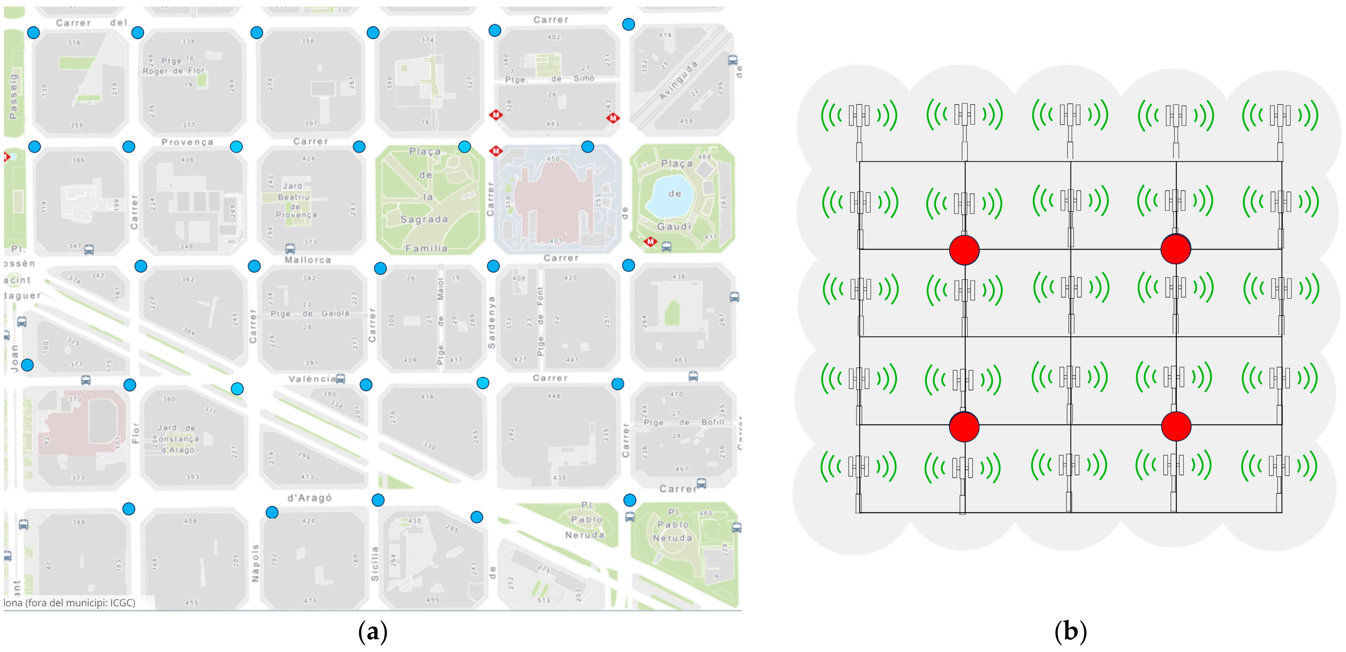


Figure 6. (**a**) An extract of Barcelona WMN topology; (**b**) the corresponding 5 × 5 topology used with rendezvous Anchor in red.

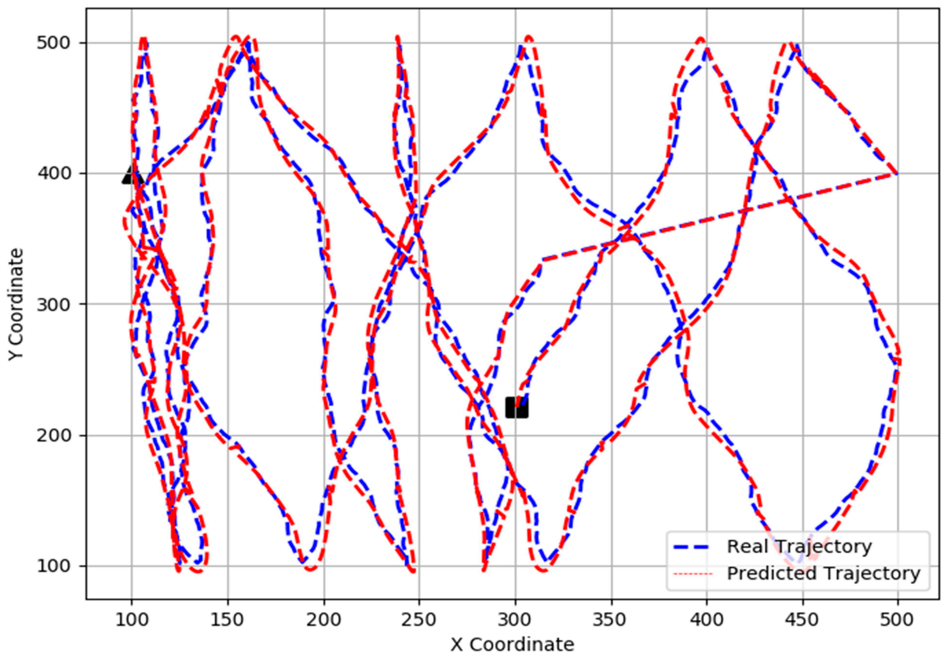


Figure 7. Comparison of the actual trajectory of the producer with the estimated trajectory using the Kalman filter.

To assess the performance of the location prediction model, we use the Root Mean Square Error (RMSE), a widely used metric for measuring the average magnitude of errors between predicted positions and actual positions. RMSE values are non-negative, with smaller values indicating better prediction accuracy.

The formula for calculating RMSE is typically:

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{k=1}^{N} (\hat{x}_k - x_k)^2 - (\hat{y}_k - y_k)^2}$$

where $(\hat{x}_k, \hat{y}_k)$ represents the predicted coordinates, (x_k, y_k) represents the real coordinates, and N is the total number of predictions.

Table 3 presents the average RMSE value of the KF model, averaged over all 100 runs for each producer speed. The KF model demonstrates accurate tracking of the producer's position. However, the RMSE value increases with the producer's speed. This is attributed to the KF's reliance on predicting the future state based on the current state. As the producer's speed rises, predicting its future position becomes more challenging due to rapid changes in location, leading to accumulated prediction errors and higher RMSE. It is noteworthy that the RMSE value of the KF is influenced by the noise in the measurements. Higher levels of measurement noise can result in an elevated RMSE value.

Table 3. RMSE of Kalman filter for various producer speeds.

Producer Speed (m/s)	1	5	10	15	20	25	30
RMSE (m)	0.23	1.52	3.17	4.41	5.94	7.72	9.46

4.2. Results and Discussion

In the subsequent section, we discuss the simulation results, focusing on the consumer Quality of Service (QoS) evaluated through four key performance indicators: average Inpt loss, average Data packet delay, average path stretch, and handoff latency. Additionally, we explore the network cost to support producer mobility in the DMMS, which is evaluated using signaling overhead performance. Furthermore, we also study how the number of Anchors in the network impacts the consumer QoS. In addition, we discuss the limitations of our strategy that can be explored for further research.

4.2.1. Interest Packet Loss

Interest packet loss is an important criterion in mobility management schemes in NDN, especially for low-latency-sensitive applications that require a negligible packet loss during the handoff to ensure that data remain relevant and useful. The Inpt loss rate is defined as the total number of Inpts sent by all the consumers minus the number of Data packets that arrived over the total number of Inpts sent. Figure 8 presents a comparison of the average packet loss rates of different mobility support schemes; the y-axis represents the average packet loss rates, while the x-axis represents the speed of the mobile producer (ranging from 1 m/s to 30 m/s). As a first observation, we notice that with the increase in the producer's speed, there is a corresponding rise in Inpt loss due to increased handoff events. Nevertheless, the result shows that the multi-RV KITE scheme has the highest packet loss rate. This is because the producer receives consumers' Inpts after TI/TD messages are exchanged to establish a new trace path, which means that the Inpts received at the time of trace path creation will be considered lost. Furthermore, multi-RV KITE suffers from the stale path problem caused by frequent handoffs that occur when the producer moves to a new PoA, and the old trace path may stay active. The DMMS and MAP-ME achieve similar packet loss rates at producer mobility speeds below 5 m/s. However, in high-speed mobility scenarios, the DMMS with high producer's location prediction accuracy exhibits significantly lower packet loss than both multi-RV KITE and MAP-ME. Even with low prediction accuracy, the DMMS maintains an acceptable packet loss ratio.

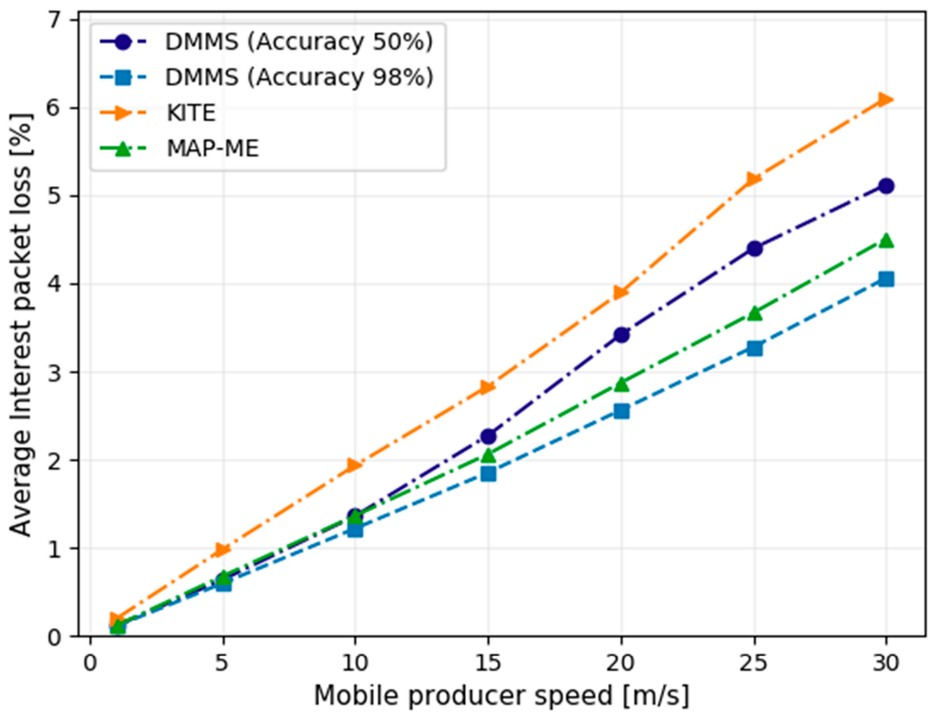


Figure 8. Average Interest packet loss rates vs. mobile producer speed.

In comparison to MAP-ME, DMMS reduces the packet loss rate by approximately 10%, and when compared to multi-RV KITE, the reduction is even more significant, at 20%. This result is attributed to the proactive mobility management that notifies the Anchor about the predicted location of the producer when handoff happens. Consequently, Inpts are redirected proactively toward the producer's new PoA.

4.2.2. Path Stretch

The path stretch metric represents the ratio of the actual forwarding path length, measured in terms of the number of hops, to the shortest possible path length between the consumer and the producer. The average hop count metric represents the average number of intermediate hops a Data packet travels from its source (producer) to its destination (consumer). Path stretch can directly impact the delay or latency experienced by Data packets as they traverse the network. When the stretch is close to 1 (i.e., the actual path is almost as short as the shortest path), Data packets reach their consumer faster, resulting in lower delay. However, if the path stretch is significantly higher, Data packets may take longer to reach their destination, leading to increased delay. An efficient mobility management strategy should ensure that the MP remains reachable using a path with minimal extension. Figure 9a,b, respectively, show the average path stretch and the hop count as a function of the producer's movement speed. As main observations, we notice that the slopes of the three schemes in the figures are not impacted much by the producer's mobility speed. We further notice that MAP-ME achieved the highest path stretch and average hop count among the three schemes; this is attributed to one of the drawbacks of the MAP-ME scheme, which tends to create a triangular path when the MP changes the PoA, as the producer sends an updated Inpt from the new PoA towards the old one to maintain a seamless producer mobility management. In contrast, the the DMMS achieves a minimal path stretch about 10% longer than the shortest possible paths, as shown in Figure 9a, and the average hop counts are 46% lower than those of the MAP-ME scheme, as shown in Figure 9b. This result is expected because of the strategic placement of Anchors within the network. By effectively positioning these Anchors at minimal distances from all nodes in the network, the additional distance traveled by packets through the Anchor remains relatively insignificant. Secondly, the location-forwarding strategy ensures that the Inpts are forwarded along the shortest path between the Anchor and the producer. It is worth noting that the multi-RV KITE scheme also has a lower path stretch due to its benefiting from rendezvous server placements, which reduces the path between the producer and consumers through the rendezvous servers.

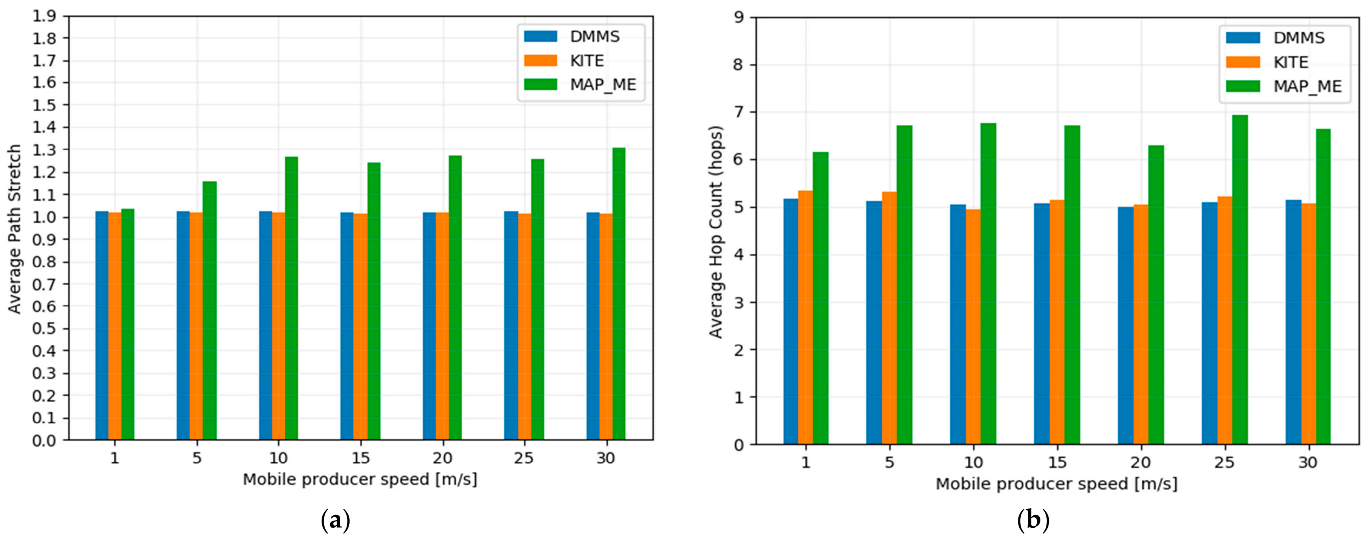


Figure 9. (**a**) Average path stretch vs. mobile producer speed, (**b**) average hop count vs. mobile producer speed.

4.2.3. Data Packet Delay

To meet the low-delay requirements of latency-sensitive applications, a suitable mobility management scheme must ensure that data can be delivered quickly and with minimal delays. The Data packet delay refers to the time it takes for a consumer to successfully receive a requested Data packet after expressing an Inpt, including the time of Inpt retransmissions. The average Data packet delay is calculated by dividing the sum of delays for each Interest–Data pair by the total number of received Data packets. Figure 10 demonstrates the average Data packet delay (represented on the vertical *y*-axis) as a function of the producer's movement speed (depicted on the horizontal *x*-axis). We can see that the DMMS and multi-RV KITE achieved a slightly lower Data packet retrieval delay. In contrast, MAP-ME achieved the highest delay among the three schemes due to the significant effect of path stretching associated with each handoff event.

4.2.4. Signaling Overhead

Signaling overhead is an important aspect to consider when designing a mobility support scheme. There is a tradeoff between signaling overhead and user experience. In fact, minimizing signaling messages can help reduce network congestion and conserve resources, leading to a more efficient use of the network infrastructure. On the other hand, increasing signaling messages for frequent updates on the network's condition can enhance user experience by maintaining a consistent quality of service. This tradeoff is a fundamental challenge in designing a mobility support scheme that provides a high-quality user experience while efficiently minimizing unnecessary signaling traffic. Signaling overhead is defined as the number of packets needed to ensure that the consumer's Inpt can reach the MP when handoff occurs. We compared the signaling overhead generated by the three schemes while varying the producer speed, as shown in Figure 11. Multi-RV KITE generates a higher signaling overhead compared to MAP-ME and DMMS. This is because of the periodic sending (every 2 s) of TI/TD packets to refresh the trace path between producer and rendezvous servers. In contrast, both the DMMS and MAP-ME schemes generate considerably less signaling overhead. The DMMS uses at least one Inpt packet to remove the old path, notify the Anchors about the handoff event, and inform Anchors about the predicted location of the MP after a handoff. Meanwhile, MAP-ME uses a single Inpt packet to update the path between the new and old producer's PoA.

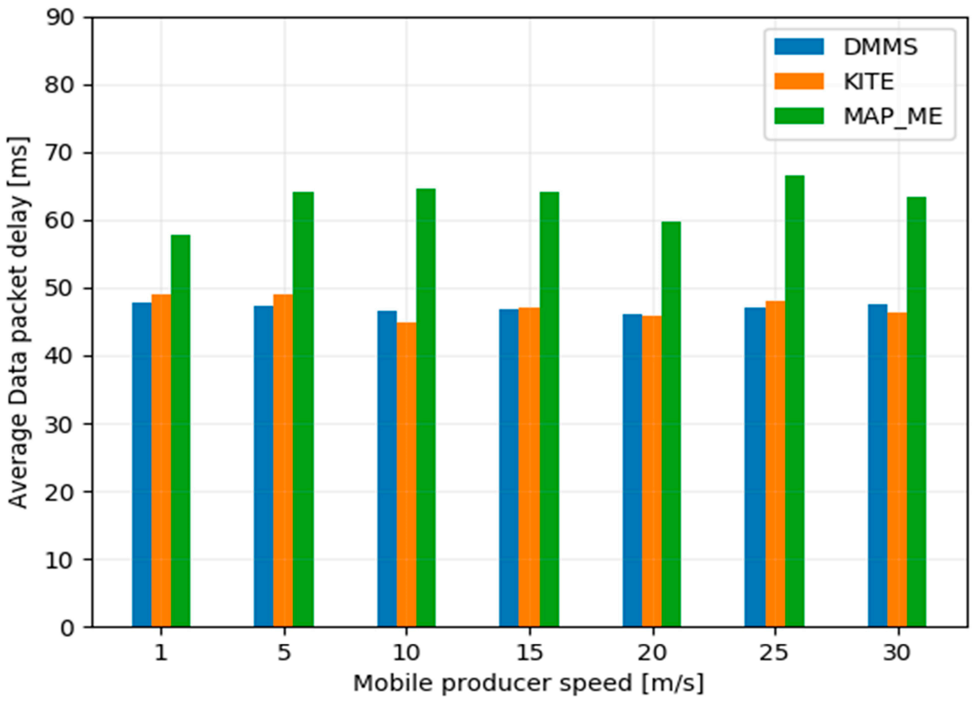


Figure 10. Average data packet delay vs. mobile producer speed.

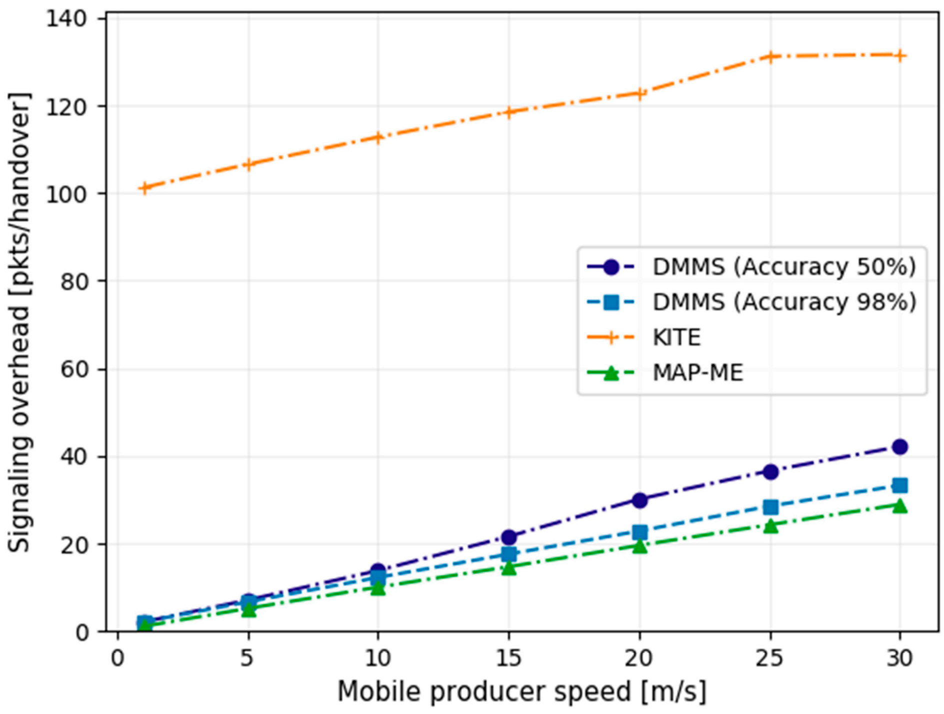


Figure 11. Signaling overheads vs. producer speed.

It is worth noting that DMMS is affected by the prediction accuracy of the producer's next location, causing an additional signaling overhead.

4.2.5. Handoff Latency

Handoff latency is defined as the period between the time when the MP loses the connection with its current PoA and the time when the MP receives the first Inpt from the new PoA. Handoff latency comprises two types of latency: (i) layer 2 latency: this delay represents the time required to complete the handoff, which is set at 100 ms in our scenario; (ii) mobility management latency: this delay is defined as the time required for the MP receive the Inpt after the handoff is completed. Figure 12 shows the handoff latency measured in milliseconds for the MAP-ME, multi-RV KITE, and DMMS schemes against the producer speed. As we can see, the highest handoff latency was for the multi-RV KITE scheme. This increase in latency was primarily attributed to the mobility management delay resulting from the exchange of TI/TD messages between the MP and the RV server, which is necessary for creating the trace path, in addition to the time required to update the MP's information on all RV servers. MAP-ME, on the other hand, achieved a lower handoff latency, where the Inpt only needs to wait for the update of the intermediate routers and reach the old PoA, which is nearly in close proximity to the new PoA in our scenario. Therefore, the handoff latency in MAP-ME is dependent on the distance between the old PoA and the new PoA.

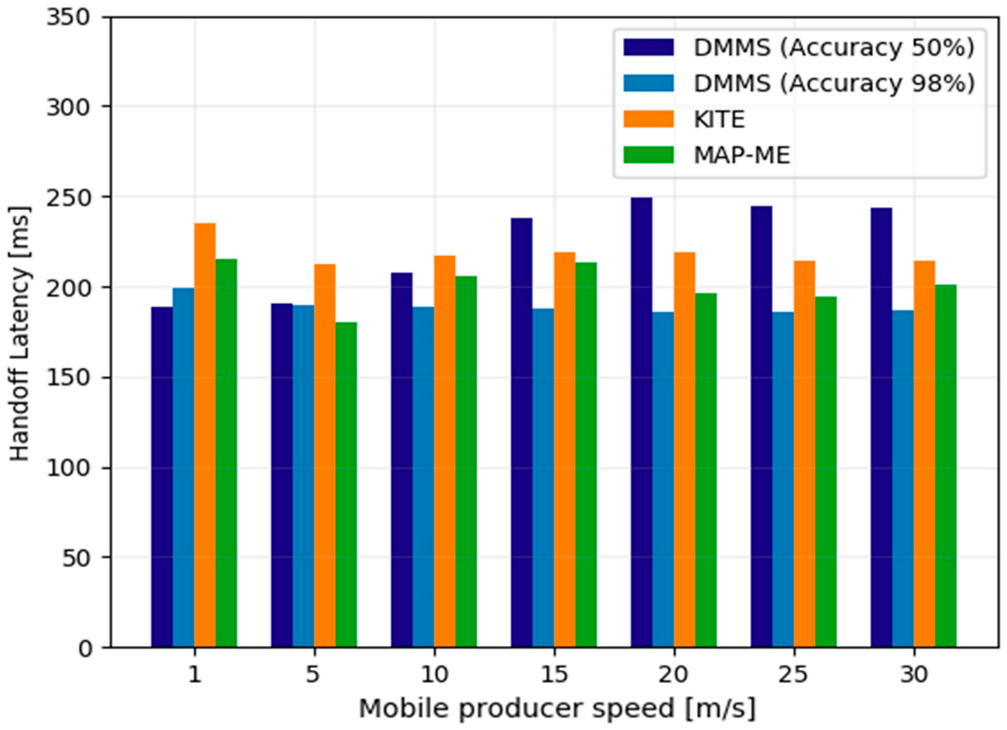
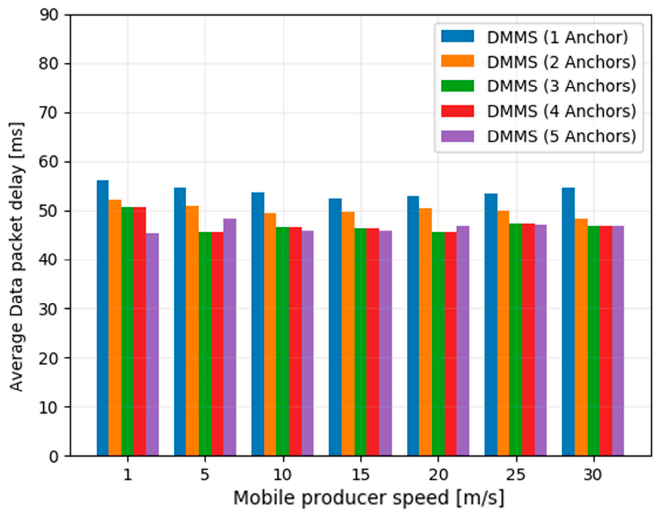


Figure 12. Handoff latency vs. producer speed.

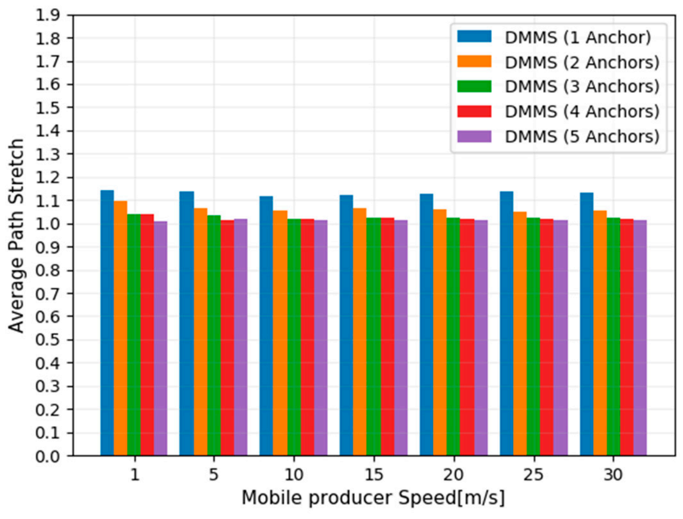
The DMMS with high producer's location prediction accuracy achieves the lowest handoff latency thanks to its proactive mechanism, which informs the Anchor about the expected location of the MP as soon as the handoff process is initiated. In other words, the DMMS reduces the handoff latency by overlapping the mobility management latency with link-layer latency, providing the quickest response to the producer's handoff event.

4.2.6. The Impact of the Number of Anchors on Consumer QoS

To quantify the impact of the number of Anchors on our proposed strategy's performance, we particularly focus on two metrics impacted by the Anchor-based approach: the path stretches and Data packet retrieval delay. We tested these metrics across varying the number of Anchors from one to five using the K-Medoids-based Anchor placement algorithm. Figure 13 depicts the path stretch and the average Data packet retrieval delay of the DMMS as a function of the MP speed using a number of Anchors varying from one to five. As the main observations, we notice that the number of Anchors significantly affects both path stretch and Data packet retrieval delay. Increasing the number of Anchors from $N = 1$ to $N = 4$ resulted in a 10% reduction in path stretch and a 16% reduction in delay. This is because an increasing number of Anchors leads to smaller clusters formed in the network, which decreases the distance between the consumer, producer, and Anchor. Consequently, it mitigates path stretch, especially in the worst scenarios where the consumer and producer are located in the same cluster and the path between them is stretched to the Anchor. We also observed that adding more than four Anchors in our scenario did not result in any additional reduction in either path stretch or Data packet retrieval delay. This is expected because increasing the number of Anchors results in shorter distances between Anchors. However, the clusters formed will become increasingly small, leading to the creation of clusters with few routers with no additional impact.



(**a**)



(**b**)

Figure 13. (**a**) Average delay vs. mobile producer speed, (**b**) average path stretch vs. mobile producer speed.

4.2.7. Limitations of the Proposed Approach

Our strategy has three main limitations. First, to support producer mobility we use the location-based forwarding strategy to fill the gap resulting from the lack of the information on the producer's new location after handoff. However, location-based forwarding cannot guarantee 100% delivery in mobile and ad hoc networks due to the local minimum issue. In a connected graph, a local minimum occurs when a node, denoted as x, is closer to the destination y than any of its neighbors, even though y is not directly connected to x. In this situation, the node x faces difficulty in deciding the next hop, resulting in a failure in the forwarding process. Second, the effectiveness of location prediction depends on the accuracy of the prediction algorithm and the quality of the input data. If the predictions are not accurate, it may result in inefficient forwarding decisions, leading to increased latency and packet loss. Third, the limitation arising from the overhead generated by the synchronization of producer location coordinates among Anchors during frequent handoffs of producers poses a significant challenge. As producers move in the network with high speed, the necessity to synchronize data introduces a notable overhead, impacting the system's performance and efficiency. The continuous synchronization process consumes communication and processing resources, leading to increased network congestion, latency, and potential resource contention among Anchors.

5. Conclusions

In this paper, we propose a distributed mobility management scheme in NDN designed to support producer mobility and ensure low-latency content delivery for UAVs as a use case. Our scheme is based on three components: (1) a new forwarding approach that combines default and location-based forwarding strategies; (2) a location prediction strategy to predict the location of the MP prior to handoff; (3) distributed Anchors that proactively track and forward the consumer Interest packets toward the MP when handoff events occur. Through simulation, we evaluated the performance of the proposed strategy using a real scenario and compared it with two popular schemes, which represent both Anchor-less and Anchor-based approaches. The results obtained showed that the proposed scheme can effectively minimize communication delay and Inpt loss with low overhead signaling under producer mobility. These promising results make the proposed strategy the best-suited producer mobility management for seamless data delivery.

Our future work includes: (1) further improving the distribution of Anchors by finding the optimal number and optimal placements of Anchors able to handle the network traffic; (2) improving the prediction accuracy by leveraging deep learning techniques by focusing on the producer mobility's related features, which we expect to present even better prediction results; and (3) conducting further experiments for different network topologies and environments.

Author Contributions: Conceptualization, M.B., N.N. and A.J.; methodology, M.B.; software, M.B.; validation, M.B., N.N. and A.J.; formal analysis, M.B.; investigation, M.B.; resources, M.B.; writing—original draft preparation, M.B.; writing—review and editing, M.B.; visualization, M.B.; supervision, N.N.; project administration, N.N. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The data can be shared up on request. The data are not publicly available due to private issues.

Acknowledgments: The authors would like to acknowledge Badr Bellaj and Abdellatif Moussaid for the funding help.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Lippi, G.; Mattiuzzi, C. Biological samples transportation by drones: Ready for prime time? *Ann. Transl. Med.* **2016**, *4*, 92. [CrossRef] [PubMed]
2. Lee, S.; Go, E.; Kwon, Y. UAV Pilot Status Identification Algorithm Using Image Recognition and Biosignals. *JAIT* **2022**, *13*, 477–485. [CrossRef]
3. Pujolle, G. An autonomic-oriented architecture for the internet of things. In Proceedings of the IEEE John Vincent Atanasoff 2006 International Symposium on Modern Computing, Sofia, Bulgaria, 3–6 October 2006. [CrossRef]
4. Benmoussa, A.; Mastorakis, S. Inter-UAV Communication Over Future Internet Architectures. In *Internet of Unmanned Things (IoUT) and Mission-Based Networking*; Kerrache, C.A., Calafate, C., Lakas, A., Lahby, M., Eds.; Springer International Publishing: Cham, Switzerland, 2023; pp. 155–172. [CrossRef]
5. Heer, T.; Garcia-Morchon, O.; Hummen, R.; Keoh, S.L.; Kumar, S.S.; Wehrle, K. Security Challenges in the IP-based Internet of Things. *Wirel. Person. Commun.* **2011**, *61*, 527–542. [CrossRef]
6. Zhang, L.; Afanasyev, A.; Burke, J.; Jacobson, V.; Claffy, K.C.; Crowley, P.; Papadopoulos, C.; Wang, L.; Zhang, B. Named Data Networking. *ACM SIGCOMM Comput. Commun. Rev.* **2014**, *44*, 66–73. [CrossRef]
7. Bellaj, M.; Naja, N.; Jamali, A. On-demand Geographic Forwarding for producer mobility support in Named Data Networking. In Proceedings of the NISS'23. ACM, Larache, Morocco, 24–26 May 2023; Volume 33, pp. 1–7. [CrossRef]
8. Cai, J.; Li, Z.; Wu, K. Performance Analysis of a Novel UAV Networks via Named Data Networking. In *Communications, Signal Processing, and Systems*; Springer: Singapore, 2019; pp. 563–568. [CrossRef]
9. Barka, E.; Kerrache, C.; Hussain, R.; Lagraa, N.; Lakas, A.; Bouk, S. A Trusted Lightweight Communication Strategy for Flying Named Data Networking. *Sensors* **2018**, *18*, 2683. [CrossRef] [PubMed]
10. Rehman, M.A.U.; Ullah, R.; Kim, B.S. NINQ: Name-integrated query framework for named-data networking of things. *Sensors* **2019**, *19*, 2906. [CrossRef] [PubMed]

11. Hussaini, M.; Naeem, M.A.; Kim, B.-S. OPMSS: Optimal Producer Mobility Support Solution for Named Data Networking. *Appl. Sci.* **2021**, *11*, 4064. [CrossRef]
12. Meddeb, M.; Dhraief, A.; Belghith, A.; Monteil, T.; Drira, K.; Gannouni, S. AFIRM: Adaptive forwarding based link recovery for mobility support in NDN/IoT networks. *Future Gener. Comput. Syst.* **2018**, *87*, 351–363. [CrossRef]
13. Augé, J.; Carofiglio, G.; Grassi, G.; Muscariello, L.; Pau, G.; Zeng, X. Anchor-less Producer Mobility in ICN. In Proceedings of the 2nd ACM Conference on Information-Centric Networking, San Francisco, CA, USA, 30 September–2 October 2015; ACM: New York, NY, USA, 2015. [CrossRef]
14. Augé, J.; Carofiglio, G.; Grassi, G.; Muscariello, L.; Pau, G.; Zeng, X. Map-me: Managing anchor-less producer mobility in content-centric networks. *IEEE Trans. Netw. Serv. Manag.* **2018**, *15*, 596–610. [CrossRef]
15. Farahat, H.; Hassanein, H.S. Proactive caching for Producer mobility management in Named Data Networks. In Proceedings of the 13th International Wireless Communications and Mobile Computing Conference (IWCMC), Valencia, Spain, 26–30 June 2017; Institute of Electrical and Electronics Engineers (IEEE): Piscataway, NJ, USA, 2017; pp. 171–176. [CrossRef]
16. Kar, P.; Chen, R.; Qian, Y. An efficient producer mobility management technique for real-time communication in NDN-based Remote Health Monitoring systems. *Smart Health* **2022**, *26*, 100309. [CrossRef]
17. Zhang, Y.; Xia, Z.; Mastorakis, S.; Zhang, L. KITE. In Proceedings of the 5th ACM Conference on Information-Centric Networking, Boston, MA, USA, 21–23 September 2018; pp. 125–136. [CrossRef]
18. Lee, J.; Cho, S.; Kim, D. Device mobility management in content-centric networking. *IEEE Commun. Mag.* **2012**, *50*, 28–34. [CrossRef]
19. Kim, D.; Ko, Y. On-demand anchor-based mobility support method for named data networking. In Proceedings of the 2017 19th International Conference on Advanced Communication Technology (ICACT), PyeongChang, Republic of Korea, 19–22 February 2017; pp. 19–22. [CrossRef]
20. Rui, L.; Yang, S.; Huang, H. A producer mobility support scheme for real-time multimedia delivery in named data networking. *Multimed. Tools Appl.* **2018**, *77*, 4811–4826. [CrossRef]
21. Ali, I.; Lim, H. Anchor-Less Producer Mobility Management in Named Data Networking for Real-Time Multimedia. *Mob. Inf. Syst.* **2019**, *2019*, 3531567. [CrossRef]
22. Zhang, Y.; Afanasyev, A.; Burke, J.; Zhang, L. A survey of mobility support in Named Data Networking. In Proceedings of the 2016 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), San Francisco, CA, USA, 10–14 April 2016; pp. 83–88. [CrossRef]
23. Azamuddin, W.M.H.; Aman, A.H.M.; Hassan, R.; Mansor, N. Comparison of Named Data Networking Mobility Methodology in a Merged Cloud Internet of Things and Artificial Intelligence Environment. *Sensors* **2022**, *22*, 6668. [CrossRef] [PubMed]
24. Hussaini, M.; Nor, S.A.; Ahmad, A. Producer mobility support schemes for named data networking: A survey. *Int. J. Electr. Comput. Eng.* **2018**, *8*, 31–42. [CrossRef]
25. Meddeb, M.; Dhraief, A.; Belghith, A.; Monteil, T.; Drira, K. Producer mobility support in named data Internet of Things network. *Procedia Comput. Sci.* **2017**, *109*, 1067–1073. [CrossRef]
26. Zhu, Z.; Afanasyev, A. Let's ChronoSync: Decentralized dataset state synchronization in Named Data Networking. In Proceedings of the 2013 21st IEEE International Conference on Network Protocols (ICNP), Goettingen, Germany, 7–10 October 2013. [CrossRef]
27. Shang, W.; Afanasyev, A.; Zhang, L. VctorSync: Distributed dataset synchronization over named data networking. In Proceedings of the 4th ACM Conference on Information-Centric Networking (ICN'17), Berlin, Germany, 26–28 September 2017; Association for Computing Machinery: New York, NY, USA, 2017. [CrossRef]
28. Zhang, M.; Lehman, V.; Wang, L. Scalable name-based data synchronization for named data networking. In Proceedings of the IEEE INFOCOM 2017–IEEE Conference on Computer Communications, Atlanta, GA, USA, 1–4 May 2017. [CrossRef]
29. Arya, V.; Garg, N.; Khandekar, R.; Meyerson, A.; Munagala, K.; Pandit, V. Local search heuristic for k-median and facility location problems. In Proceedings of the Thirty-Third Annual ACM Symposium on Theory of Computing (STOC'01), Hersonissos, Greece, 6 July 2001; Association for Computing Machinery: New York, NY, USA, 2001. [CrossRef]
30. Bellaj, M.; Najib, N.; Jamali, A. Distributed Rendezvous placement for producer mobility support in NDN-IoT. In Proceedings of the ITM Web Conference, Casablanca, Morocco, 8–9 March 2023. [CrossRef]
31. Xu, D.; Tian, Y. A comprehensive survey of clustering algorithms. *Ann. Data Sci.* **2015**, *2*, 165–193. [CrossRef]
32. Han, J.; Pei, J.; Kamber, M. *Data Mining: Concepts and Techniques*; Elsevier: Amsterdam, The Netherlands, 2011.
33. NDNLPv2-NFD. Available online: https://redmine.named-data.net/projects/nfd/wiki/NDNLPv2 (accessed on 31 May 2022).
34. Zhong, G.; Zhang, H.; Zhou, J.; Zhou, J.; Liu, H. Short-Term 4D Trajectory Prediction for UAV Based on Spatio-Temporal Trajectory Clustering. *IEEE Access* **2022**, *10*, 93362–93380. [CrossRef]
35. Peng, J.; Zhang, P.; Zheng, L.; Tan, J. UAV Positioning Based on Multi-Sensor Fusion. *IEEE Access* **2020**, *8*, 34455–34467. [CrossRef]
36. Luo, C.; McClean, S.I.; Parr, G.; Teacy, L.; De Nardi, R. UAV position estimation and collision avoidance using the extended Kalman filter. *IEEE Trans. Veh. Technol.* **2013**, *62*, 2749–2762. [CrossRef]
37. Yang, K.; Bi, M.; Liu, Y.; Zhang, Y. LSTM-based deep learning model for civil aircraft position and attitude prediction approach. In Proceedings of the 2019 Chinese Control Conference (CCC), Guangzhou, China, 27–30 July 2019. [CrossRef]
38. Khaghani, M.; Skaloud, J. Autonomous Vehicle Dynamic Model-Based Navigation for Small UAVs. *J. Inst. Navig.* **2016**, *63*, 345–358. [CrossRef]
39. Welch, G.; Bishop, G. *An Introduction to the Kalman Filter*; University of North Carolina at Chapel Hill: Chapel Hill, NC, USA, 1995.

40. Liang, B.; Haas, Z. Predictive distance-based mobility management for PCS networks. In Proceedings of the IEEE INFOCOM'99. Conference on Computer Communications. Proceedings. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies, New York, NY, USA, 21–25 March 1999; Volume 3, pp. 1377–1384. [CrossRef]
41. Mastorakis, S.; Afanasyev, A.; Zhang, L. On the evolution of ndnSIM: An open-source simulator for NDN experimentation. *ACM SIGCOMM Comp. Commun. Rev.* **2017**, *47*, 19–33. [CrossRef]
42. Marrero, D.; Macías, E.; Suárez, Á.; Santana, J.A.; Mena, V. Energy Saving in Smart City Wireless Backbone Network for Environment Sensors. *Mob. Netw. Appl.* **2019**, *24*, 700–711. [CrossRef]