Supplying Partners Suite of Protocols for P2P 3D Streaming Over Thin Mobile Devices

by

Haifa Raja Maamar

Thesis submitted to the
Faculty of Graduate and Postdoctoral Studies
In partial fulfillment of the requirements
For the Ph.D. degree in
Electrical and Computer Engineering

School of Electrical Engineering and Computer Science
Faculty of Engineering
University of Ottawa

To my loving parents

Abstract

The recent advances in mobile computing devices and wireless networking produced the technical platform for multimedia services over thin mobile devices. Nowadays, we are witnessing an important growth in applications using thin mobile devices, such as social networks, virtual walkthrough, media streaming, and augmented reality (AR), just to mention a few. Most of these applications are based on the client-server architecture, however several studies showed that the client-server architecture suffers from various issues, such as the server bottleneck, latency and the lack of scalability. This led most of the systems to switch to the peer-to-peer (P2P)-like environment for its scalability and potential cost saving.

P2P multimedia streaming over thin mobile devices-based classes of applications has known a significant growth during the last years. Although P2P video streaming over thin mobile devices received a great deal of attention, the application of 3D streaming over mobile devices was challenging mainly due to the limited mobile resources and capabilities, as well as the wireless medium limitations. Having 3D streaming over Mobile Ad hoc Networks (MANET) is considered more challenging given that the 3D streaming-based system has to deal with a dynamic environment resulting from nodes mobility, which may lead to route breakages and connection loss. Therefore, one of the major difficulties in 3D streaming over MANET is related to the supplying partner's strategy that aims at determining the most suitable source holding the required 3D data to stream it quickly and efficiently to the requesters.

In this thesis, we propose our P2P based 3D streaming system which we refer to as MOSAIC as well as a suite of supplying partner strategy protocols for P2P 3D streaming over thin mobile devices. Our proposed suite of protocols selects the potential sources that have the relevant 3D data, based on a set of criteria such as the source location, the mobile device's available resources as well as its residual energy. We also proposed a multihop supplying partner selection protocol that takes into account the signal strength and the nodes mobility when streaming the relevant 3D data. The performance evaluation obtained to evaluate our MOSAIC system as well as our suite of protocols using an extensive set of NS2 simulation experiments, is then reported.

List of Publications Related to the Thesis

Refereed Journal Papers

- Haifa Maamar, Graciela Roman, Azzedine Boukerche, Emil Petriu: Energy management control for supplying partner selection protocol in mobile P2P 3D streaming, Journal of Concurrency and Computation: Practice and experience, DOI: 10.1002/cpe.1814, (2011)

- Haifa Maamar, Azzedine Boukerche, Emil Petriu: Streaming 3D Meshes over thin mobile devices, Wireless Communications Magazine (2012)(In press).

- Haifa Maamar, Azzedine Boukerche, Emil Petriu: 3D Streaming supplying partner protocols for mobile collaborative exergaming for health. IEEE Transactions on Information Technology in Biomedicine, 10.1109/TITB.2012.2206116, (2012) (In press).

- Haifa Maamar, Azzedine Boukerche, Emil Petriu: An Efficient Multihop Supplying Partner Protocol for 3D Streaming based Systems over MANET. Concurrency and Computation: Practice and Experience (Accepted).

Refereed Conference papers

- Haifa Maamar, Azzedine Boukerche, Emil Petriu: A Multihop Supplying Partner Protocol for 3D Streaming Systems over Thin Mobile Devices, Second ACM International Symposium on Design and Analysis of Intelligent Vehicular Networks and Applications (DIVANet12), October 2012

- Haifa Maamar, Graciela Roman, Azzedine Boukerche, Emil Petriu: Two-Level Energy-Based Supplying Partner Selection Protocol for Mobile P2P 3D Streaming, IEEE ICC Communication Software Services and Multimedia Applications Symposium ('ICC'12 CSSMA'), June 10th- 15th, 2012

- Haifa Maamar, Graciela Roman, Azzedine Boukerche, Emil Petriu: Energy-Aware analysis for supplying partner selection in mobile P2P 3D streaming, The 16th IEEE International Symposium on Computers and Communications (IEEE ISCC11), June 28th- July 1st pages:74-79 (2011)

- Haifa Maamar, Azzedine Boukerche, Emil Petriu: Load Balancing model for mobile peer-to-peer networks-Based 3D Streaming, The IEEE/ACM International Symposium on Haptic Audio-Visual Environments and Games (HAVE10) pages:51-56 (2010)

- Haifa Maamar, Azzedine Boukerche, Emil Petriu: MOSAIC–A Mobile Peer-To-Peer Networks-Based 3D Streaming Supplying Partner Protocol, The 14th IEEE/ACM International Symposium on Distributed Simulation and Real Time Applications (DSRT 10), 8 pages, October 17-20, pages: 61-68 (2010)

- Haifa Maamar, Richard Pazzi, Azzedine Boukerche, Emil Petriu: A supplying partner strategy for mobile networks-based 3D streaming - Proof of concept, IEEE International Parallel and Distributed Processing Symposium, the 9th International Workshop on Performance Modeling, Evaluation, and Optimisation of Ubiquitous Computing and Networked Systems (PMEO-UCNS10), pages: 1-6 (2010)

Acknowledgements

I am immensely grateful to my supervisors Dr. Azzedine Boukerche and Dr. Emil Petriu for their professionalism, guidance, encouragement, moral and financial support throughout the work on my PhD. I couldnt have done it without the motivation that I received from my supervisors.

I also would like to thank Dr Graciela Roman Alonso for her help and guidance during my PhD. Her pertinent comments helped improving the performance evaluations of the protocols proposed in this thesis.

I would like to thank my friends and colleagues at our PARADISE Research laboratory, especially Maram Bani Younes, Robson De Grande and Cristiano Gato. They were always supportive and understanding and made every day of work at the lab a pleasant experience. My special thank you goes to my ex-colleague and best friend Anahit Martirosyan for her continuous support, good mood and love.

Last but not least, I am very thankful and grateful to my family - my parents Moncef and Souad, and my brothers Waleed and Sofien for their continuous support and encouragement, for filling my life with love, and especially for always being patient and understanding. This thesis is dedicated to them because I couldn't have done it without their sacrifices and patience.

Glossary of Terms

ACK Acknowledgment message.

ANG$_{mvt}$ Angle of movement.

AOI Area of Interest.

AR Augmented Reality.

Area$_{Intersect}$ Area of intersection.

Buffer$_{Overload}$ Buffer of overloaded sources.

CL Cluster Leader.

Consumed$_{Energy}$ Source consumed energy.

CPM Compressed Progressive Meshes.

CPU Central Processing Unit.

CVE Collaborative Virtual Environment.

DHT Distributed Hash Table.

Dif$_{Load}$ Difference load between the source and the minimum declared load.

DSR Dynamic Source Routing.

DVE Distributed Virtual Environment.

FLoD Flow Level of Detail.

FOV Field Of View.

GPS Global Positioning System.

HMD Head Mounted Devices.

IBR Image Based Rendering.

ID$_{ObjFirst}$ Sequence number of the first packet in the main buffer.

ID$_{ObjLast}$ Sequence number of the first packet in the main buffer.

IM Interest Mamangement.

ISP Internet Service Provider.

$L_{closeNeighbor}$ Set of nearby peers.

$\mathbf{L}_{collision}$ List of colliding neighbors.

$\mathbf{L}_{Direction}$ List of peers located in the movement direction of the requester.

$\mathbf{L}_{Neigh}$ List of Neighbors.

$\mathbf{L}_{Overload}$ List of overloaded sources.

$\mathbf{L}_{Source}$ List of sources.

$\mathbf{Load}_{Min}$ Minimum load.

LOD Level of Detail.

LODDT Level of Detail Description Tree.

$\mathbf{Main}_{Buffer}$ Main Buffer.

MANET Mobile Ad hoc Networks.

MAR Mobile Augmented Reality.

MCPM Modified Compressed Progressive Meshes.

MDC Multiple Description Coding.

MMOG Massively Multiplayer Online Games.

MPEG Moving Picture Experts Group.

NACK Negative Acknowledgment message.

NS2 Network Simulator 2.

NVE Networked Virtual Environment.

PDA Personal Digital Assistants.

$\mathbf{Percentage}_{Intersect}$ Percentage of intersection.

PM Progressive Meshes.

P_r Received Power.

P_t Transmitted Power.

P2P Peer-to-Peer.

QoS Quality Of Service.

QTVR QuickTime Virtual Reality.

R_{AOI} Radius of AOI.

RDMM Random Direction Mobility Model.

Ren_n Source residual energy.

RM Region Master.

ROI Region Of Interest.

RPGM Reference Point Group Mobility Model.

RR% Realization ratio.

$RR\%_{max}$ Threshold value for realization ratio.

RSS Received Signal Strength.

RTT Round Trip Time.

RWPM Random Waypoint Mobility Model.

SAP Streaming Access Point.

SRR Server Request Ratio.

TTL Time To Live.

TTS Time To Serve.

VE Virtual Environment.

VNC Virtual Network Computing.

2D Two Dimentional.

3D Three Dimentional.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

In this chapter, we shall introduce the basic idea of 3D streaming, explain the difference that exists among 3D streaming and video streaming and present our research objectives.

1.1 What is 3D streaming?

3D streaming consists of gradually downloading and rendering 3D content, such as meshes and textures to permit the interaction of the user with its virtual world without a full download or a pre-installation [2, 4].

Due to the significant advances in the area of communications and networking, the development of networked virtual environment (NVE) applications is increasing considerably. These applications include massively multiplayer online games (MMOGs) [8, 11], virtual walkthrough [13] and augmented reality (AR) applications [20], just to mention a few. The implementation of most of these applications requires a very large 3D content where the users must store the applications locally way in advance in order to be able to use it in the future. Consistency among all users, in that case, is maintained by broadcasting events related to the game state, across the network. Given that many users participate in the application and that the content is dynamic and has to be frequently updated, it becomes extremely hard for users to keep up with the updates. Moreover, as the content of the virtual environment (VE) grows, pre-installation is no longer a viable approach. Furthermore, having VE based class of applications on thin mobile devices such as personal digital assistant (PDA), or iPhones, is very hard to realize if a full download of the VE is required. This is mainly due to the limited capabilities and memory storage of the mobile devices. Therefore, in order to decrease their communication and

computation overhead, it is preferable to only get the content of interest through 3D streaming techniques.

1.2 3D Streaming VS Video Streaming ?

Although P2P streaming has been very popular for file downloading, video streaming and 3D streaming based systems have different characteristics, and they require different design decisions due to the data type to be streamed which is a 3D mesh for 3D streaming and image based in video streaming.

Moreover, one of the next major differences relies upon the object manipulation that is considered easy in 3D streaming when compared to video streaming. In other words, once the user gets the 3D mesh, he is able to manipulate it, i.e., rotate it, have a closer view of it, zoom in and zoom out, without needing to stream extra data. This is not possible to realize in video streaming where everytime the user needs to manipulate the object, additional streaming of images is required.

The content fragmentation and access pattern are another difference between the 3D streaming and the video streaming. In video streaming, the video is fragmented into sequentially ordered frames based on the time. Its access pattern is therefore considered linear and frame prediction can be applied in this case; Whereas in 3D streaming, the 3D content can not be ordered and has to be fragmented based on the user's behavior and his viewing angles. The 3D content access pattern in 3D streaming is thus considered non linear and prediction in this case has been proven to be very hard to accomplish [2], given that the user can have unpredictable movements.

When streaming is based on the P2P delivery, the selection of the media provider, also called supplying partner, differs depending on whether 3D streaming or video streaming is being used. When video streaming is used, similarly to file sharing, the supplying partner is chosen based on the data availability. However, in 3D streaming, the supplying partner is selected based on the data locality, i.e., peers who are nearby in the VE have nearly the same view of the VE and may therefore need the same 3D content. 3D streaming is therefore considered different from video streaming, presenting thereby different issues and challenges. In the next section, we shall present the challenges and issues relative to 3D streaming.

1.3 Challenges and Issues

Several 3D streaming techniques have been proposed in the literature [23, 33, 55, 69]. However, several network challenges need to be addressed before 3D streaming technology becomes a commodity. Moreover, the use of 3D streaming across wireless networks and/or mobile ad hoc networks (MANET) induces extra challenges that need also to be taken into consideration. Therefore, a given 3D streaming solution is considered efficient if it takes into account the following challenges:

- *Mobile device limitations*: In this thesis, we focus upon thin mobile devices which consists of mobile devices with limited capabilities in terms of low processing power, limited storage capacity, limited graphics' hardware and graphics' accelerator [1], just to mention a few. These limitations make it very diificult for thin mobile devices to render and process large and complex 3D scenes. Example of the targeted thin mobile devices are Personal Digital Assistant (PDA), iPhones, iPADs.

- *Wireless network bandwidth limitation*: where the wireless medium access is constantly exposed to background noise, multipath fading, shadowing and interferences, which makes the bandwidth variant over the time, and leads to link disruptions and thereby resulting in high error rates and packet loss [51].

- *Density*: has a significant impact on the quality of the streaming where a high node density may lead to a significant communication overhead, resulting into packet collision, while a low node density induces a decreased signal strength resulting into a high packet drop.

- *Nodes mobility*: dynamic networks creates a new challenge for 3D streaming based systems. This is mainly due to the dynamic and frequent route changes.

- *Streaming performance*: such as latency, network congestion, long data acquisition times, and invalid requests just to mention a few, may have a devastating effect on 3D streaming over mobile networks.

- *Scalability*: considered as one of the most important requirements in NVE applications [8, 11], may be hard to obtain on client-server based models as opposed to P2P networks, and therefore need to be carefully investigated.

1.4 Problem Statement and Research Objectives

3D streaming based class of applications are considered very challenging to achieve given that they are considered network demanding. In order to provide a satisfactory streaming quality to the user, the application must satisfy a stringent set of requirements at both ends of the application, i.e., the sender and the receiver. Therefore, by identifying the relevant requirements and satisfying them, the application ensures a satisfactory streaming quality. Several requirements need to be satisfied and they include the network bandwidth, delay, and congestion just to mention a few. By all means, the architecture of the network application can also play an important role for meeting and supporting the requirements.

Several 3D streaming applications [2,23,33,40] have been deployed. However, most of the already developed 3D streaming applications are based on the client-server architecture where the server holds the entire VE and is responsible for streaming the required 3D objects to interested clients based on their positions. The clients are only responsible for rendering locally the received 3D object. However, 3D streaming applications based on the client-server have significant drawbacks such as the server bottleneck due to the sole supplier issue, and the request contention that occurs when several data requests are sent and need to be served at the same time. Both the server bottleneck and the request contention induce significant delay which may degrade the system's performance. Another important drawback that the client-server architecture suffers from is the lack of scalability that occurs due to the server's limited resources. This issue made the client-server architecture inappropriate for large scale applications.

With the advances of wireless communication and mobile computing, several applications [12–14, 35, 40, 45] started to take advantage of 3D streaming over thin mobile devices such as PDA, iPhones, and iPads. However, this was not an easy task since thin mobile devices can not render large and complex 3D scenes, and have limited 3D resources and capabilities. Researchers worked extensively to solve this issue and several approaches have been presented in the literature [13, 14, 45]. Most of the adopted approaches are based on remote visualization using a client-server based architecture that uses an image-based technique as opposed to sending a 3D mesh. Some obvious drawbacks for remote visualization include: server bottleneck, significant delay, lack of scalability, and communication overhead that occurs during the manipulation of the 3D object and where extra images need to be streamed.

Due to all of the above mentioned restraints, several applications have opted for the

P2P delivery, where each user[1] can be both *requester* and *supplier* for other users. Using P2P approach has not only addressed the server bottleneck issue but also improved the system's scalability. However, on the downside, P2P-based solutions have to face the regular P2P issues, in terms of extra load on the peers. P2P-based solutions have adopted the localization approach by having replicates of the system on each peer and sending updates to each peer in order to keep the consistency of the scene as viewed by the users. The key idea behind P2P-based 3D streaming is that users need only a portion of the VE in order to perform actions. Therefore, the user needs only to interact with other users located within its Area of Interest (AOI). The AOI of a given user is determined based on its viewing angle, its field of view (FOV). Recall that peers located within a user's AOI are called AOI neighbors [4].

AOI neighbors have similar views of the VE due to their overlapped visibility and may need the same 3D data. Therefore, if a given user needs 3D data, it only requests it from its AOI neighbors. However, since users are mobile within the VE, the list of AOI neighbors may change. Therefore, there is a strong need in implementing a set of protocols that allow to quickly discover the suitable peer that possesses the relevant data and that has enough bandwidth to stream the needed data quickly and efficiently to requesters. We refer to this class of protocols as *supplying partner selection* protocols. Moreover, 3D streaming is still considered a challenging problem when dealing with thin mobile devices, since frequent 3D streaming overuses the device's resources. Hence, extra factors such as the bandwidth, or the mobile device's resources need to be taken into consideration during the supplying partner selection process.

The supplying partner selection search may spread on multiple hops, and the selected supplying partner may be far (located at multiple hops) from the requester. Sending the relevant 3D data packet using best effort, may lead to a packet drop due to the large size of the packet and to the changing quality wireless medium. This clearly affects the streaming performance of the system. In order to avoid this from happening, a content delivery method adapted for MANET based 3D streaming applications is in strong need.

Several researchers [2, 4, 23] have pointed out the challenges of choosing the suitable source to supply the required 3D data to requesters. However, most of the already designed schemes proposed a supplying partner technique for *wired* networks-based 3D streaming. To the best of our knowledge, the solutions found in the literature regarding this topic do not address the supplying partner strategy for MANET-based 3D streamingpaperspapers and/or mobile P2P netorks.

[1]The terms *user*, *node*, *client* and *peer* will be used interchangeably from now on.

In this thesis, we focus upon a 3D streaming over thin mobile devices, and we propose a P2P 3D streaming system which we refer to as MOSAIC as well as a suite of supplying partner protocols for P2P 3D streaming . Our suite of protocols are based on the P2P architecture in order to avoid the drawbacks of the client-server models. We concentrate the research objectives on 3D streaming over thin mobile devices as follows:

- *Acquisition time*: In our protocols, as opposed to previously proposed techniques, we choose to select *multiple* sources taking into account several criteria such as the physical location and the mobile device's available resources. Once a list of potential sources is computed, the streaming is distributed among them, which leads to a faster acquisition time.

- *Mobile device limitation*: Our proposed approach takes into account the available resources on each mobile device during the object selection process. We recall that rendering multiple 3D objects is considered resource demanding and frequent 3D streaming overuses the mobile device' resources and dissipates its energy. In our work, we address the mobile device limitations, by reducing the number of 3D objects that need to be streamed.

- *Network bandwidth limitation*: This issue is very challenging and may greatly affect the system performance. For this reason, our proposed protocols try to cope with the network bandwidth limitation in order to lessen the bandwidth consumption. To accomplish this, our proposed supplying partner selection protocols use an interest management technique assisted by a location-based filtering of messages in order to reduce the number of messages transmitted accross the network.

- *Scalability*: All of our proposed protocols are based on the P2P network overlay. By having multiple users participating in the streaming process, the server bottleneck is avoided thereby improving the scalability of 3D streaming over P2P networks.

- *Streaming performance*: Overloaded suppliers badly affect the system performance since requests queue up at the sources which results in significant delay and high response time. Overloaded supplier's energy is also dissipated faster, which shortens the source's energy lifetime and consequently affects the system's P2P behavior. This leads to the system becoming centralized where requests are served by the server, which could result in a server bottleneck. Our supplying partner selection protocols improve the streaming performance by taking into account the available

resources on each mobile device. Our proposed protocol aims at relieving the loaded sources and balancing the load among the peers.

1.5 Contributions

This thesis comprises four main contributions which are presented in the following:

- *MOSAIC*: is a P2P based 3D streaming system over thin mobile devices which we refer to as MOSAIC. Our proposed system is intended for augmented reality based class of applications using thin mobile devices where the real world is mapped into the virtual world and thin mobile devices are used to navigate within the VE.

- *Supplying Partner Selection Protocols*: which comprises two protocols namely the Location based Supplying Partner Protocol (LB3D) and the Resources based Supplying Partner Protocol (RB3D). The former aims at selecting sources based on their location and proximity to the requester; while the latter aims at relieving the overloaded sources by taking the resources of the supplier into consideration. Both protocols shall discover multiple supplying partners while minimizing the peer's 3D data acquisition time.

- *Energy Management Control Mechanisms*: which comprises two protocols namely, the One-Level Energy based Supplying Partner Protocol (1L-EB) and Two-Level Energy based Supplying Partner Protocol (2L-EB). In the former, the requester is responsible for managing the energy consumtpion of the system by minimizing the load on the sources and selecting the least loaded sources; whereas in the latter, the requester and the supplier play important and complementary roles in the system energy management.

- *MULTIPLY*: which is a multihop supplying partner selection protocol aiming at selecting sources located at multiple hops from the requester. During the source discovery phase, only peers located in the movement direction of the requester and at given distance are examined. Sources are selected based on the 3D data availability. The protocol is based on an efficient content delivery technique where the signal strength between peers is measured before sending the 3D data packet.

The remainder of this thesis is organized as follow. Chapter 2 presents the state of the art of 3D streaming techniques used to address the challenges of 3D streaming over mobile

devices. In Chapter 3, we present our supplying partner selection suite of protocols for P2P 3D streaming over thin mobile devices which we refer to as MOSAIC. Chapter 4 presents our energy management control for supplying partner selection protocol, while Chapter 5 introduces our multihop supplying partner selection protocol. Chapter 6 concludes our thesis followed by our future work.

Chapter 2

Related Work

Nowadays, we are witnessing a significant growing interest in the use of VE based class of applications, such as augmented reality AR [35], virtual walkthrough [13], multiplayer online games [8, 44], CVE [20] just to mention a few. Most of the earlier implemented applications are based on the client-server architecture with a copy stored locally on the client. However, it was proven that it is not always efficient to have the entire VE stored on the client, due to memory constraints. To overcome this problem, most of the applications applied 3D streaming that consisted of gradually downloading and rendering 3D content such as meshes and textures to permit the interaction of the user with its virtual world without a full download or a pre-installation. In this chapter, we will present a state of the art of the 3D streaming based solutions.

2.1 3D Streaming Approaches

So far a significant body of work [2, 16, 23] has been dedicated to the implementation of an effective 3D streaming technique. Based on the nature of the application, several solutions have been designed and implemented [12, 32, 69, 70]. All of the proposed techniques can be grouped into four categories namely: *Geometry replication*, *progressive meshes*, *impostors*, and *Image Based model*. Table 2.1 illustrates a comparison of these techniques stating their advantages and disadvantages.

2.1.1 Geometry Replication

The geometry replication technique [90, 91] consists of having a copy of the 3D models geometry stored on the client and rendered by its local hardware. The copy of the 3D

Table 2.1: Comparison of 3D Streaming Techniques

Approach	***Technique***	***Advantages***	***Limitations***
Geometry Replication	3D model stored on the client	Scalability	Not applied on thin mobile devices
Progressive Mesh	3D objects rendered and transmitted progressively	3D model visualized in lower quality and refined	Long times for mesh generation
IBR	Reference images, Panoramas	Light, Applied on thin mobile devices	Display problems: pixel loss, distortion
Impostors	3D model rendered to an image and texture mapped to a shape	Faster rendering times and lower bandwidth usage	Inadequate for objects close to users; New impostor for every user rotation

model can either be acquired from the client's hard disks or optical drivers, as it is done by computer games, or downloaded from the server. In Massively Multiplayer Online Games (MMOG) for example, the VE is stored locally on each client who performs all 3D rendering. In order to maintain the scene view consistency among all clients, events related to the game state are broadcasted across the network. However, this technique presents major disadvantages when dealing with wireless networks using thin mobile devices. Firstly, the size of the VE is extremely large and it takes a long time to download. Secondly, due to the limited mobile resources and capabilities, (i.e., low processing power, limited storage capacity, limited graphics hardware and graphics accelerator) mobile clients are unable to render the received 3D models in the same quality as rendered by the server, and consequently unable to process large and complex 3D scenes. Therefore, this technique is not suitable for applications running on thin mobile devices.

2.1.2 Progressive Meshes (PM) Techniques

Progressive meshes (PM) [55] technique consists of having virtual objects rendered and transmitted progressively. With the PM technique, the client is able to visualize the 3D model in lower quality and then progressively refine it until it obtains its original quality. To do so, first, PM generates and streams a basic low polygon shape version of the original 3D model. The simplest representation of the 3D model is called the *base mesh*. Then, details to generate more complex representations, called *refinement layers*, are generated and progressively streamed as discussed by [46, 49, 52, 53]. The Progressive Meshes techniques are based on the edge collapse [46] and edge split [46] mechanisms. The former aims at reducing the resolution of the object, whereas the latter aims at applying the inverse process, i.e. increasing the resolution of the object [49]. Figure 2.1

illustrates the simplest representation of a teapot, while Figure 2.2 shows a more refined version of the same teapot and finally a high refined version of the teapot is represented in Figure 2.3.

Several derivatives [49, 55, 63, 67] of the PM technique exist. Isenburg and Lindstrom [55] [55] proposed the streaming meshes technique, where a mesh is stored into a fixed size buffer, and triangles and vertices are either added or removed from the mesh in order to reconstruct it. Kircher and Garland [49] proposed an algorithm that provides a multi-resolution representation for deforming objects with a high quality approximation. The multilevel mesh proposed by the authors aims at having iterative edge contraction, and uses less space since it stores progressive representation (mesh connectivity at each level) instead of the entire hierarchy. However, edge contraction makes the mapping false since vertices forming the children can move. Fang and Tian [61] also proposed a technique for mesh simplification based on the triangle contraction simplification.

Compressed progressive meshes [63] (CPM) approach aims at improving the PM technique by focusing on the overhead and latency engendered by the PM and trying to get rid of it. For this matter, CPM uses the implant sprays technique to refine the mesh by assembling the vertex splits into batches. In consequence, CPM occupies 50% less storage than PM model. Modified Compressed Progressive Meshes (MCPM) technique [67] improves CPM by including a decision module that selects the most suitable transport protocol for each geometric sub-layer taking into account the network bandwidth and the loss ratio.

PM technique has several advantages, however, on the downside, it is considered complex since mesh generation takes long times, and it is considered not efficient in terms of compression ratio. Moreover, progressive refinement induces a considerable overhead specially when the entire mesh has to be downloaded. And finally, PM techniques is not the best solution to use when dealing with thin mobile devices, given that large and complex 3D models are difficult to stream due to the mobile device's limitation in terms of memory capacity.

2.1.3 Impostor Technique

In the impostor technique [69, 70] illustrated in Figure 2.4, the server, based on the client's orientation, is responsible for rendering the complex 3D model into an image and sending it to the client; while the client is responsible for texture mapping the obtained image to a simple shape such as a plane or a box [69, 70]. This technique has faster

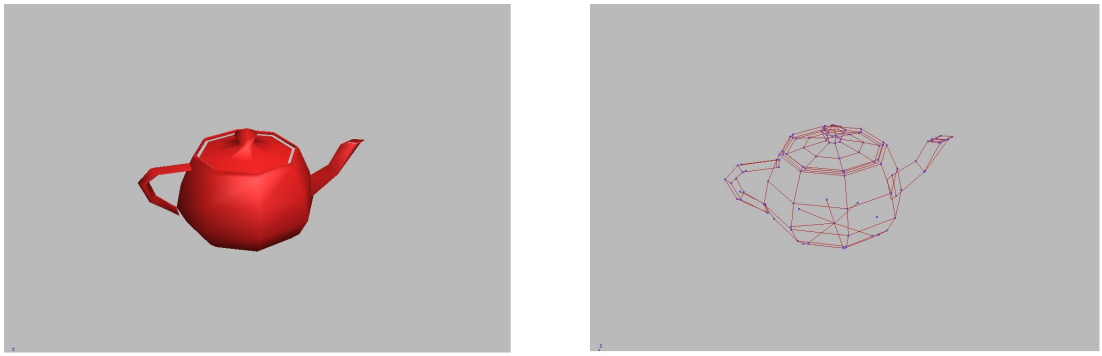

Figure 2.1: Low Level of Details of a Teapot

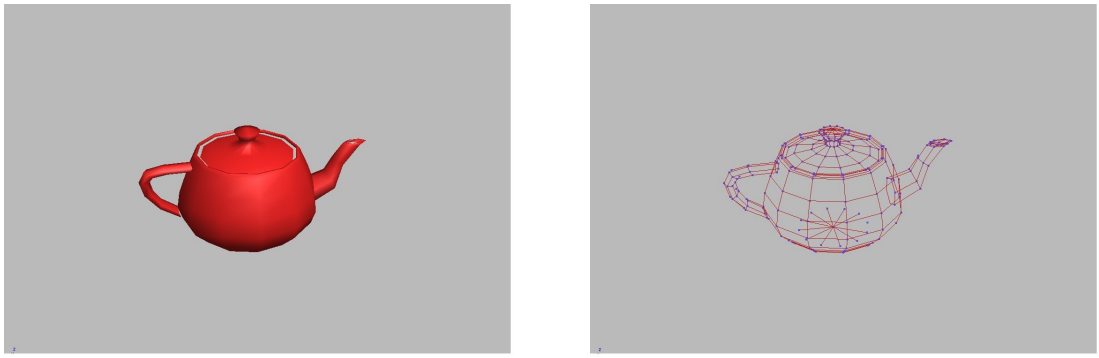

Figure 2.2: Medium Level of Details of a Teapot

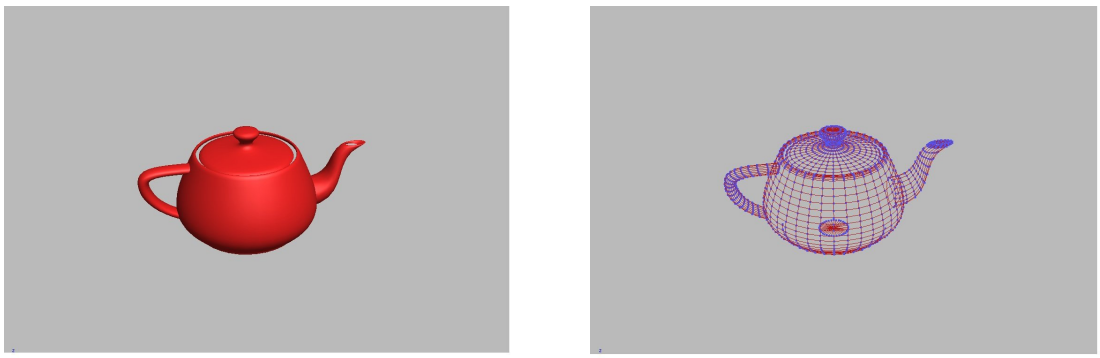

Figure 2.3: High Level of Details of a Teapot

Figure 2.4: A case of impostors

rendering times and lower bandwidth usage, since impostors are simple shapes with smaller sizes compared to the actual 3D model, which makes them easy to render and transmit. Impostors are usually used to render objects that are far from the users sight. However, this technique is not considered a good choice for objects that are closer to the user, since the latter can notice the lack of image depth. Moreover, this technique induces significant overhead given that new impostors have to be streamed every time the user moves and changes its orientation.

2.1.4 Image Based Model

Rendering 3D models requires powerful devices with high resources and capabilities since 3D models are large in size, which makes them ineligible for low bandwidth networks, and impossible to use in light-weight mobile devices. The image-based model, also called image based rendering (IBR), overcomes the beforehand mentioned difficulties by representing the VE using images instead of geometry. In IBR, the server is responsible for the rendering tasks since it is equipped with powerful rendering hardware, while the clients are simply responsible for displaying the received images [72]. This concept is called Remote Rendering and is considered the best choice among the 3D streaming techniques when dealing with thin mobile devices. IBR techniques can be applied using two methods: Reference images [78] and Panoramas techniques [14, 16, 91].

Reference Images

In order to avoid display delays and long waiting times to acquire the 3D data, several IBR techniques are applied on the clients by using a set of reference images [78] to construct intermediate views, new images or portion of them. These techniques are based on the plenoptic function [31] that states that the world can be perceived as a set

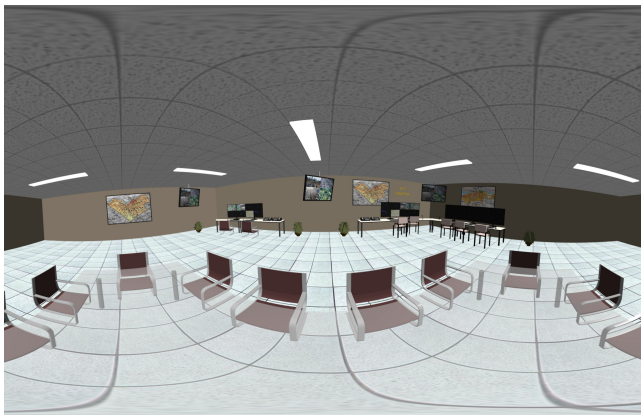

Figure 2.5: Panoramic Images

of light rays filling the space that can be seen by eyes or cameras. The plenotpic function represents real world views, where the user can be at any position, look anywhere, at any time. However, this is not an easy task when dealing with VE, given that the user experience is more restrained, depending on the application and the available hardware.

Panoramas Technique

Nowadays, most of the 3D IBR representations use panoramas [14] that give to the user the illusion of seeing new views. Panoramas consist of a series of panoramic images obtained by capturing different images of the visible views in all directions and projecting them on a 3D shape such as spheres, cylinders, and/or cubes just to mention a few [91]. Figure 2.5 illustrates a panaorama projected on a cylinder. Spherical panorama allows the user to look anywhere from his position, since it covers 360 degrees horizontally and vertically. However, this type of panoramas produces image distortions and involves multiple rendering passes which makes them inefficient. Cubic panorama, also covers 360 degrees horizontally and vertically, and projects the result on a cube's six faces that has each a 90*90 field of view. The cubical projected images represent usually the image source that several spherical panorama viewers are based on. Cubic panoramas generation is more efficient when compared to spherical panoramas and is supported by most of today's standard graphic cards [16]. Google Street View [37] and QuickTime VR (QTVR) [38] are two applications based on panoramas.

2.2 3D Streaming-based Applications Architectures

3D streaming has been extensively studied in recent years. So far, several research works have been dedicated to the design of 3D streaming systems and frameworks [2,73,74,80]. In this section, we shall present the 3D streaming works that have been implemented, classifying them into two groups based on the architecture used, i.e.,: *remote rendering*, and *P2P Streaming*. In the former group, i.e., remote rendering, a client-server architecture has been used, where the server is responsible for maintaining the VE, generating the 3D model, and sending it to the clients; while the clients are responsible for rendering the received 3D data locally. The server, in this case, is considered the solely 3D data supplying source. However in the second group, i.e., P2P 3D streaming, a given peer can be both supplier and requester. The VE data is distributed among the participating peers, and the requester's neighbors can be considered supplying partners and can be responsible for streaming the 3D data to requesters. In the following, we shall discuss further both the remote rendering and P2P streaming techniques.

2.2.1 Remote Rendering

Remote rendering is the solution adapted by several frameworks due to the client's limited storage capacity. When dealing with wireless networks using thin mobile devices, 3D streaming becomes difficult to realize since thin mobile devices can not render large and complex 3D scenes. Furthermore, they have limited 3D acceleration support, and storage capacity. Several researchers have extensively investigated to solve this issue and several approaches have been presented. Most of the adopted approaches are based on remote visualization. Depending on the user's position and its viewing angle, the server renders the scene, and then sends the required images to the client for display. We identify two types of remote rendering [34]: the server side and the hybrid side. In the former, the remote server hosts the 3D environment, and is responsible for rendering the complex 3D scenes and sending only images to the mobile user according to its view. In the latter, the server is responsible for rendering some parts of the 3D environment such as high resolution of the 3D environment, while the client is responsible for rendering the remaining parts such as the residual error images and low resolution geometry [34]. Several research works [12, 32, 73, 74] have applied remote rendering and they can be classified in two approaches, namely: remote rendering for desktop-based applications and remote rendering on thin mobile devices.

Remote rendering for desktop-based applications

Several remote rendering desktop-based frameworks have been designed in the literature [73, 74, 80], such as Virtual Network Computing (VNC) [73], or Chromium [74]. In VNC [73], the server computes resulting images, and sends them back to the remote clients after recieving their events that have been initiated using a mouse and a keyboard for instance. As for Chromium [74], which uses a modified implementation of OpenGL [39], it sends the views for rendering to a graph of processing nodes, and the resulting images are sent back to the remote clients for display. Yoon and Neumann [85] have proposed an approach where the client uses a web browser to display the remote 3D VE. Their obtained results indicate that their scheme exhibits a better frame rates using IBR methods when compared to previous studies.

Remote rendering on thin mobile devices

Due to the recent advances in mobile computing devices and wireless networking, remote rendering based applications over thin mobile devices are increasing incredibly. Lamberti et al. [32] proposed a three tier architecture where the server is able to render complex 3D models, encodes them into MPEG video streams and sends them to the mobile clients for visualization. To accomplish this, Lamberti et al. [32] used a cluster of personal computers to render the VE scenes and to provide a good resolution. The rendering procedure is done using the Chromium framework that divides the 3D set into N parts and distributes them among the servers for parallel processing. Each station is responsible for executing the OpenGl directives and generating a portion of the image. All portions are then assembled to construct the final image, encoded into MPEG packets and streamed to the mobile clients via MPEG Transport Streaming (MPEG-TS). The scene rendering is applied on demand, every time the client changes his orientation. However, the proposed technique does not take into account the user navigation to predict and buffer the images to be streamed.

Boukerche et al., [12, 16], proposed a buffer management mechanism that uses a two-level buffer in order to handle the rendering and communication tasks at the same time. For this purpose, two buffers are used: texture buffer and communication buffer. The former holds the reference image for texture information, while the latter holds reference images that will be passed to the texture buffer when needed. The authors also proposed a rate control protocol that informs the server about the network state, so it can adapt its rate and avoids congestion and bandwidth over utilization [12, 16]. Similarly, Boukerche

et al., [13], proposed a scheduling and buffering mechanism that improves the streaming performance and avoids the network jitter. Their scheme consists of streaming in advance images that are close to the user's position in the VE and that have a high probability to be displayed in the close future. As a conclusion, the client device is able to render the user's viewpoint locally avoiding network jitter. However, this research work limits the view rotation of the user to one axis, which is not realistic in 3D environments. Chang and Ger [7] used IBR to accomplish 3D graphics capability on mobile devices. They designed a client-server system for mobile devices equipped with IEEE 802.11b based wireless interface, where the clients (thin mobile devices) request images, and the server (3D graphics programs running on a desktop computer),renders the 3D environment and sends reference images and depth maps. By applying McMillans image warping algorithm [30], and upon receiving the reference images, the clients generate intermediate views. Brachtl et al [26], the VE is rendered as a movie on the server and is transferred then to mobile clients. However, users in this case are unable to interact with the environment.

Diepstraten and Ertl [80] combined the processing powers of both the clients and the server and implemented a remote line rendering for mobile devices. To do so, the server is responsible for executing most of the tasks, while the client is only responsible for rendering 2D line primitives that have been produced on the server and derived from arbitrary 3D scenes. Although this technique reduced the load on the client, it presents several drawbacks such as the low resolution of the rendered images that are neither colored nor shaded.

Mobile augmented reality

The last years have shown an increasing growth demands of multimedia services over mobile devices, especially in AR based classes of applications. In mobile augmented reality (MAR) [22], the real world is mapped to the virtual world and mobile devices are used to navigate the VE. Most of the MAR approaches have a camera attached to the device, and images are taken, sent to the server so it can render them and send them back to the client for display. Early applications such as the Touring Machine [5], Mars [22], and Archeoguide [21], just to mention a few, are based on the use of wearable devices, like effective laptops, that are not considered a good choice for single-handed applications since they are large, expensive and heavy. To overcome these drawbacks, MAR applications started using thin mobile devices, i.e. PDA and HMD that greatly enhanced the user capability to take advantage of MAR applications. Most of the MAR

applications [40,45] are based on a centralized client-server approach, where the server is responsible for streaming 3D meshes to the clients. MobiTree [35], Batportal project [43], AR-PDA project [45] are examples of MAR applications that use thin mobile devices. Since PDAs and cellular phones have limited capabilities in terms of storage, they can not have the entire VE installed on the device. Therefore, streaming can be used.

Wagner et al. [42] designed a stand-alone tracking mobile AR system that consists of a combination of self-tracking thin client and a tracking computation off-loaded to the server. Depending on the accessibility of the server and on the client location, the system changes dynamically and transparently at the application runtime by either having the client fully self-contained or by offloading the tracking computation to the server. Doran et al. [35] designed the Mobitree system that aims at progressively streaming and rendering 3D tree models on thin mobile devices. The server is responsible for the generation of the 3D tree model, its rendering and its streaming to the mobile client. The AR-PDA project developed by Gausemeier et al. [45] uses mobile thin clients, such as PDAs, to provide AR based daily services. It is based on a client-server architecture and is an image based object recognition and tracking method for mobile devices. The AR-PDA includes a camera integrated in it that takes live-video images. The latter are sent to the server that examines them, recognizes the object, adds relevant context sensitive information and streams them back to the AR-PDA for display. Krosche et al., [41], designed the MobiDENK system that is a location-aware information system that guides tourists to historic sites of interest and provides related information. It uses thin mobile devices (PDA) that have a GPS system incorporated. Depending on the tourist position, the server streams multimedia information related to the monument they are viewing.

Limitations of remote rendering

The remote rendering techniques [13, 16, 32] have shown a significant load reduction on the clients' devices and can overcome the device's limitations; however, the server-based supplying techniques suffer from several obvious drawbacks. First, this approach implies a single point of failure given that the entire VE is centralized at the server. Second, remote rendering suffers from server bottleneck and network congestion that occur if several clients join the session and several clients' requests have to be answered at the same time. This obviously induces significant delay [24, 25] and degrades consequently the system's streaming performance. Third, a client-server based technique suffers from a lack of scalability, given that the server has a limited capacity. Despite the fact that the server's limited capacity could be overcame using multiple servers, the scalability

remains hard to achieve.

2.2.2 Peer-to-Peer Streaming

P2P based streaming techniques have been used as an approach to avoid the client-server based limitations. Therefore, instead of asking the server, peers can be both providers and requesters for each others minimizing delay and remote access costs. Several research works [2, 17, 33] have demonstrated the benefit of using P2P architecture in order to minimize the server load and to increase the user satisfaction with the multimedia service. However, on the downside, P2P networks are dynamic, where nodes can move freely, join and leave the network at any time. Thus, P2P-based applications have to face the supplying partner strategy challenge which consists of selecting the suitable source that has the relevant data to stream it to the requesters. When using P2P streaming over thin mobile devices, additional challenges, such as the mobile device limitations and the networks impairments, need to be taken into consideration. In the following section, we shall present the challenges of streaming 3D objects over thin mobile devices.

2.3 Challenges of 3D Streaming over Mobile Devices

Geometry based-3D streaming requires a large amount of data and high resolution textures to be transmitted in real time over the network. This obviously creates network bottlenecks, overuses the network resources and affects the streaming performance. When applied in wireless networks using thin mobile devices, 3D streaming becomes more difficult to achieve due to the devices' limited capabilities. Therefore, 3D streaming over thin mobile devices is not an easy task due to the number of challenges that need to be faced. These challenges can be related to the mobile devices limited capabilities, the network impairments such as network bandwidth over-utilization, latency, network congestion, scalability, just to mention a few, or related to the architecture of the application.

In this section, we shall discuss the vulnerabilities to which 3D streaming applications are exposed to and explain how the 3D streaming techniques have dealt with these vulnerabilities. Table 2.2 presents an overview of the challenges that the 3D streaming face and the possible solutions that can be applied to address these issues. In what follows, we will discuss some of the challenges in more details.

Table 2.2: 3D Streaming Challenges and Solutions

Challenges	***Solution***
Device Limitations	1-Remote Visualization 2-Remote line rendering
Bandwidth over-utilization	1- IM techniques (AOI) 2- Zoning techniques
Network Congestion	1- Rate control mechanisms 2- P2P based streaming
Latency	Dead Reckoning techniques
Scalability	P2P based 3D streaming
Wireless bandwidth limitation	1- View dependent simplification 2- LOD technique
Node mobility	Efficient routing protocols
3D data retrieval	Supplying partner strategies

2.3.1 Bandwidth Over-Utilization

Several research works [8, 16, 86] have pointed out the challenge of 3D streaming characterized by bandwidth over-utilization. Most of the proposed solutions, suggested to use Interest Mamangement (IM) [8], or zoning techniques that aim at filtering the data to be processed in order to reduce the network load and avoid the bandwidth over-utilization. In the following we shall describe the different techniques that have been implemented to reduce the network load and to cope with the bandwidth limitation. Table 2.3 illustrates a comparison of these techniques stating their advantages and limitations.

1. **Interest Management (IM)**:

   One of the most used techniques to address the VE network load issue is the IM that aims at minimizing the transmission of unnecessary messages. Indeed, broadcasting updates to all users within the VE, knowing that each user has a limited visibility, is not efficient. When 3D applications are used on mobile devices, this issue, i.e., the transmission of unnecessary messages, has a higher impact and may be a major cause of streaming performance degradation. In fact, all of the mobile nodes will be broadcasting every update message received, which leads to the flooding problem, network congestion and to the bandwidth over-utilization. To overcome this problem, several approaches [8, 86] agreed upon designing an IM technique to filter the messages and to send only updates that are of interest to each user [8].

Most of the IM techniques [8] are based on the user's perception (i.e., what the user can see) that illustrates its Area of interest (AOI) also known as region of interest (ROI). Most of the time, the user's AOI is represented by a fixed size circle having the user at the center. Only objects that are within the user's AOI are processed and streamed to the client, reducing, therefore, a significant amount of processing time and network load. As for objects that are far from the user's AOI, they are attributed lower priorities for rendering and streaming. Although the user-centric AOI has been a very popular approach being used [8, 11], it suffers from object popping discontinuities [86], as a given large object will not be rendered by the user if it is located outside the user's AOI; however as the user moves forward, the large 3D object becomes visible suddenly on the user display, which may disrupts the users navigational experience [86]. In order to overcome this problem, i.e., the object popping problem, Seo and Zimmerman [87] proposed a novel visibility algorithm called Object-initiated view model, where each object has its own AOI. Unlike the traditional user AOI model that has a circular shape, the authors [87] used a square shaped AOI given that this shape is very simple and more efficient to be indexed in a grid-based virtual world.

2. **Zoning based techniques**:

   Another set of filtering mechanisms that have been discussed in the literature [2,62] make use of the *zoning techniques*. When dealing with 3D environments and in order to restrict the interest of a user, proximity is manifestly the criteria to take into account, since entities close to the clients will have nearly the same view of the VE, and will likely share the same interest. Therefore, filtering is accomplished by dividing the VE into regions. Each user will only receive data and updates related to objects within its region, or adjacent regions. Different shapes were adopted during the zoning, such as honeycomb regions [8], or voronoi diagram [2]. However, one of the significant drawbacks of the zoning-based techniques is that the amount of data filtered is highly dependent on the size of the zone.

   Yu et al. [8] suggested dividing the VE into honeycomb regions where the user's AOI has a radius of a hexagon cell. Each user only sends updates to those that are covered by its AOI. Boulanger et al. [62] claimed that messages can be filtered and minimized by taking into account obstacles. They compared a variety of IM algorithms using different zonings while integrating various visibility levels such as: Euclidean distance, square tile, hexagonal tile, and ray visibility algorithms. Their

Table 2.3: Comparison of 3D Streaming Approaches for Network Load Reduction

Approach	***Technique***	***Advantage***	**Limitations**
Interest Management (IM)	Only relevant is sent	Minimization of the amount of data processed	Highly dependent on the zone size
View Dependent Simplification	Adaptation of the 3D model resolution to the factors' change	Network load reduction	3D model stored in memory; Increase of the 3D model size
Level of Details (LOD)	Different representation of each object	Taking into account the user's bandwidth	Data redundancy
Dead Reckoning	Object's state prediction	Latency and updates reduction	Sudden changes in the user's movements
Remote Line Rendering	2D line primitive rendered locally on the client	Network load reduction Load balancing between C/S	Inefficient with fully shaded colored objects

obtained results indicate that ray visibility is the best algorithm to use in terms of message filtering when compared to Euclidean distance algorithm, square tile algorithm, and hexagonal tile algorithm.

2.3.2 Latency Issue

Latency may be the result of different network issues such as a network congestion, invalid requests, and long times to acquire 3D data. All of these issues greatly affect and considerably degrade the quality of the streamed 3D data. Network congestion for example can take place if many requests have to be served at the same time. This case happens specially in a client-server architecture, where the server becomes a bottleneck since it is the only supplier for clients' requests. To address this issue, two sets of idea have been proposed: (1) a rate control mechanism [16] that allows the sender to adapt its transfer rate based on the information returned from the requesters, and (2) P2P streaming, where systems are designed based on the P2P architecture.

Long times to acquire data affects the streaming performance of the system since the clients experience delay to display the received 3D data. In order to eliminate this delay, dead reckoning [16] is the technique that have been applied. Dead reckoning is based on prediction, where using the behavioral model of a given object, and its previous states, the user can predict the actual remote object's state. Dead reckoning parameters include the object's speed, its position and direction [66]. Several studies [16, 19] used the dead reckoning especially in predicting where the user is probably going and which images need to be streamed. By determining the movement direction of the user, the

server can stream in advance the images that have a high probability to be displayed in the close future decreasing therefore the time to acquire the 3D data. Wu et al. [19] used the dead reckoning techniques to predict the application workload and to assign thus the required CPU cycles. According to the authors, the mobile workload changes (increases) when there is a change in the scene level of detail (LOD) or in the complexity of the animations. These changes lead to an increase of the mobile device energy consumption. By displaying the lowest LOD, the system minimizes the energy consumption. Using these assumptions, the authors proposed their Energy efficient real time rendering (EARR) heuristic technique that minimizes energy consumption by comparing the predicted frame rate with the actual frame rate and adapting future predictions.

2.3.3 Wireless Medium Limitation

Streaming 3D meshes over wireless networks is very challenging due to the wireless medium limitation that is constantly exposed to background noise, multipath fading, shadowing and interferences. This makes the bandwidth variant over the time, and leads to link disruptions resulting in high error rates and packet loss, which obviously degrades the quality of the 3D data streamed. In order to overcome this issue, researchers [51] opted for increasing the wavelength of the signal carrier. However, this solution reduces the amount of available bandwidth for the nodes and is considered inadequate given that it does not satisfy the stringent requirements of the 3D streaming based applications. Other researchers opted to reduce the size of the 3D mesh by using techniques such as LOD, PM [46] and view dependent simplification [56, 57, 60] just to mention a few.

1. **View Dependent Simplification**: The AOI techniques are usually tightly used with another set of techniques called multi-resolution or view dependent simplification [79]. The view dependent polygonal simplification methods [46, 84] are based on a data structure englobing different representations of a given 3D object while changing its resolution. The resolution of the 3D model and its level of detail, as illustrated in [79] are adapted to the change of several factors such as the user's location, and illumination, just to mention a few. The object's resolution is determined based on the user's depth of sight and on the distance that separates the user from the object. An object is considered visible to a given user, if it is within the users AOI or, if the users scope and the object's scope overlap. Despite the fact that this technique reduces the bandwidth over-utilization, the view dependent simplification technique does increase the size of the 3D model. Moreover,

it requires the entire 3D model to be stored in the main memory. To address the last issue, especially when dealing with mobile clients, external memory, also called out-of-core [57], was proposed to store the VE on the disk. However, the mobile device performance is seriously affected and degraded since disk access speed is very slow when compared to CPU speed.

2. **Levels of Detail (LOD)** Most of the 3D streaming solutions assign different importance levels to the objects located within the user's field of view and their display resolutions are therefore adapted to the amount of perceived details. LOD techniques [23] take advantage of this paradigm, and provides different representation for each object, by reducing or increasing the object's complexity and changing the number of polygons and/or the texture resolution among other parameters. Although, the LOD technique takes into account the user's perception and available bandwidth, on the downside, it produces, for each 3D object, multiple models at different resolutions to be sent over the network. This unfortunately may lead to data redundancy and computation overhead.

Another significant challenge related to the wireless medium is the time varying link that may be instable due to the node mobility [51]. This may greatly affect the 3D streaming quality due to an increase in the packet drop which unfortunately increases the 3D data acquisition time. Efficient routing protocols are therefore needed in order to establish new routes and ensure a reliable delivery of the 3D data.

2.3.4 Scalability Issue

In order to insure the scalability requirement, researchers agreed upon using P2P networks. A considerable number of solutions [2, 3, 23] used P2P streaming over wired networks and several frameworks such as FLoD [2], aiming at providing scalability in NVE, have been implemented. In wireless networks, we notice that existing research works [17, 18] focused on video streaming and proposed P2P based protocols for video streaming over thin mobile devices. To the best of our knowledge, none of the existing research works implemented a solution for 3D streaming based applications over thin mobile devices.

2.3.5 3D Data Retrieval

The 3D data retrieval consists of finding the relevant 3D data and the peer that holds it in an unstructured P2P networks. This issue is especially important for P2P based 3D streaming applications, where there is no entity that holds the entire VE. The supplying partner selection strategy consists of selecting the suitable source that has the relevant 3D data to stream it to the requesters. In P2P based applications, peers collaborate together and participate to provide the relevant 3D data. Therefore peers can be both requesters and suppliers for other users. In general, the supplying partner chosen is selected among the adjacent peers [2]; however, this peer may not have enough bandwidth to transfer the relevant data. Moreover, streaming 3D data using mobile devices, overuses the mobile device's resources in terms of energy and memory. Therefore, extra measures need to be taken in consideration when selecting the supplying partner. factors such as the mobile device's residual energy, need to be taken into consideration during the selection of the supplying partner.

2.4 Visibility Determination Techniques

Due to the mobile device's limited capabilities, several visibility determination techniques aiming at selecting the 3D objects required for the user's scene have been developed and presented in the literature [84, 87, 92]. The most commom object selection approach is used based on the distance separating the user from the 3D object. This technique was first proposed by Clark [92] in 1976 who suggested that different resolutions (LOD) of a given 3D object should be created and the selection of the corresponding LOD has to be performed based on the distance separating the user from the object.

In [27], the authors proposed a DVE that uses a multiresolution caching technique. This caching mechanism tries to have a minimum resolution of the virtual objects. The object's resolution is determined based on the user's depth of sight and on the distance separating the user from the object. An object is considered visible to a given user, if it is within the users AOI or, if the users scope and the object's scope overlap. The authors also highlighted the importance of the object's size. Very small objects, even within the AOI, should not be rendered, which reduces the network load.

Zhu et al., [89] proposed a peer-assisted rendering, where each user has a frustum projected from its position delimiting the visible objects to the user. The authors classified the objects into three categories, far, static near, and dynamic based on the position

of the user, its view direction, and its viewing angle. Seo and Zummermann [87] proposed an object-initiated view model and a user-initiated view model. In their work, each object has an AOI centered at the object location. A given object is considered visible to the user if the object's AOI covers the location of the user. The author then proposed an enhanced versions of the user and object visibility algorithms by taking into consideration the parameters of the visual acuity model including the width of the target objects, the distance between the user and the target object, and the angle subtended at the users eye by the target object [86].

Li et al., [68] prioritized the selected objects by dividing the user's AOI into three regions, based on the angular distances between the user and the objects: visible region, potential visible region and the prefetching region. In the former, i..e, the visible region, the objects are considered visible to the user and have the highest priority for transmission. As for the potential visible region, it includes the objects that are not instantaneously visible to the user, but those that will become visible if the user simply moves forward or changes its direction. Therefore, these objects may be transmitted to the user, once all of the objects in the visible region have been transmitted. And finally the prefetching region includes the non visible objects. These objects would be prefetched to the user if and only if objects in the former regions are transmitted and an extra network bandwidth is available.

Hu et al., [2, 3] proposed Flow level of Detail (FLoD), a solution for exchanging 3D content between peers in an arranged 2D P2P networks. In their work, it is the responsibility of the user to determine the list of 3D objects that need to be acquired. To do so, upon receiving the scene description from its neighbors, the user determines the list of 3D objects covered by its AOI and acquires the relevant objects in a P2P manner. FloD improved the scalability, however, it is considered inefficient when it comes to dynamic VE, given that it is difficult to update the scene descriptions.

Cheng et al. [79] used progressive meshes and aimed at addressing how a receiver can determine which vertices are visible before it receives the complete mesh. For this matter, they supposed that parents of visible vertices are visible. The parent of a vertex is the one from which the vertex is split. The authors assigned a unique ID to each vertex split and organized them in a hierarchical way forming a forest of binary trees.

Botev et al., [88] proposed the HyperVerse protocol to provide 3D web based on federated and torrent concepts. In this work, multiple servers are constructed in a static structure overlay. These servers are responsible for keeping track of objects and users positions in the VE. It is the responsibility of the servers to notify each user of its relevant

objects and neighbors based on its field of interest.

Cavagna et al., [23, 33] proposed the LOD description tree (LODDT) protocol designed to support 3D streaming on the urban cityscapes large environment. The VE is partitioned into specific progressive and hierarchical tree object data structures called pBtree, where each node in the pBtree has an associated aura that determines the LOD and colours of the object it represents. In order to acquire a given object, the pBtree is traversed in a breadth first manner using some culling rule. However, the LODDT overuses the network bandwidth due to the redundant information.

Rahimi et al., [90] proposed an activity centric based technique to select 3D objects in MMOGs over mobile devices. Their proposed approach used a context-aware method, where the objects to stream are selected based on the activity undertaken by the player. Activity centric approach relays on the game designer in assigning an importance factor for each object in the VE according to all possible activities. However, the authors used LOD as a 3D streaming technique, which is considered inefficient when dealing with thin mobile devices.

2.5 Supplying Partner Techniques for Multimedia Streaming

Multimedia streaming in P2P alike environments is receiving a great deal of attention in recent years for its scalability and potential cost savings. Multimedia streaming can be classified into three categories: *video*, *real-time* and *3D streaming* techniques. Although, P2P based multimedia streaming has several advantages, several challenges remain to be resolved related to the supplying partner selection. Several supplying partner selection strategies have been proposed in the literature [2, 17, 23, 36] and are summarized in Table 2.4. In this section, we shall review the supplying partner selection strategies, classifying them based on the nature of the multimedia.

2.5.1 Selection Techniques for Real-Time Streaming Systems

P2P live streaming consists of streaming and disseminating a live video content to all users [81]. The playbacks of the video are synchronized, and users do not have the flexibility to start watching the video from the point they choose. Several live streaming systems have been deployed. Ciullo et al. [28] designed a P2P-TV system taking into account the ISP resources utilization and trying to optimize them while minimizing the

Table 2.4: Supplying partner strategies for multimedia streaming

Protocol	***Approach***	***Mobile***	***Media Type***	***Limitations***
Hu et al., 2008 [2]	Naive query response method	X	3D	Latency, communication overhead
Cavagna et al., 2006 [23]	Source with the smallest TTL	X	3D	Latency, Request contention
Royan et al., 2007 [33]	TTL, available data	X	3D	Client's resources not considered
Sung et al., 2008 [4]	Multi-level AOI	X	3D	Long data acquisition time
Peltotalo et al., 2009 [36]	Proximity based selection	√	Real-time	Computation overhead
Wu et al., 2009 [17]	Dynamic chaining technique	√	Video	Performance degradation for unstable moving patterns
Wu et al., 2010 [18]	Preference scores	√	Video	Long download time
Chow et al., 2007 [6]	Naive query response method with MDC	√	Video	Control overhead and routing load
Cheng et al., 2009 [79]	Sources grouped based on the hierarchical structure of the mesh	X	3D	Control overhead and latency
Chien et al., 2010 [24]	Bandwidth availability on sources	X	3D	Inefficient due to the random mobility

delay. To do so, the authors [28] proposed a two-step technique. First, the streaming sources are selected based on their proximity to the requesters, which minimizes the delay since the source is physically close to the requester. Second, the streaming path length is minimized and information about the network status are taken into consideration. Peltotalo et al. [36] designed a real-time P2P streaming system for mobile network environments. The system groups the peers into clusters based on their RTT values, and uses the partial RTP stream concept as media delivery. The peers clustering can be according to the peers' proximity or to their interest level for a given piece of data. Each cluster has a cluster leader (CL) that is responsible for managing the peers within the cluster and for selecting the supplier for a given requester. CLs are selected based on their capabilities: high throughput, large memory and CPU power and a long lasting battery life. Although this approach insures scalability, it suffers from significant computation overhead. The technique applied for the CL selection designates the data source as a CL for a given cluster. This obviously overuses the resources of the source.

2.5.2 Video Streaming Selection Techniques for Wireless P2P Networks

P2P Video streaming on mobile devices has been quite popular in the recent years. Several P2P video streaming based class of applications have been implemented [17, 18, 59]. They rely on the collaborative behavior of peers to assist the source in delivering multimedia content, reducing therefore costs and increasing the scalability of video distribution. Nevertheless, finding the suitable peer to provide the relevant data remains a constant challenge. Researchers worked extensively to solve the supplying partner strategy issue and several approaches have been presented. These research works can be classified into 2 groups: one-hop and multihop supplying partner strategies.

One-Hop Supplying Partner Techniques

Wu et al. [17] proposed a two-level framework that provides effective data management for video streaming in mobile networks. Their mechanism is based on the headlight prefetching and the dynamic chaining. In order to overcome the wireless connection instability and the uncertainty of clients' movements, the authors used headlight prefetching where media segments are prefetched and streamed based on the virtual illuminance of the headlight zone [17]. The dynamic chaining is applied as a supplying partner technique used to alleviate the load on the server and to maximize the user's cache utilization. This supplying partner technique is based on location proximity, movement pattern and load balancing [17]. When a mobile user is in need of information, it requests its cell's streaming access point (SAP) that looks for possible supplying partners located in the same cell. The source chosen is one of the requester's neighbors. It has to be lightly loaded and heading towards the same direction as the requesting user. If a partner is found, the mobile user is chained to it to receive the required information and can be considered as a supplying partner for other mobile users who need the same information at around the same time. Simulation results of this mechanism show that the streaming disruptions and bandwidth utilization are minimized, and the response time improved. However, the performance of the protocol is degraded under fast changes and highly unstable moving patterns.

Similarly, Wu et al. [18] proposed an adaptive P2P video streaming, where a set of supplying partners is computed and video frames are downloaded from the source in a round robin fashion. Yang et al. [54] proposed the ViewCast model used to design a multi-party/multi-stream system that, based on the user interest, decreases the resources'

consumption and provides minimum quality guarantees. The authors applied a source diversification technique to diversify the supplying sources in order to reduce the load on sources.

Multihop Supplying Partner Techniques

Several research works presented multihop supplying partner techniques for P2P based video streaming. Shiang et al. [59] proposed a distributed video streaming approach over multihop wireless networks taking into account timely information feedback related to the network status and expected delays. Based on this information, the authors adapt the packet scheduling and routing strategies to avoid the risk of packets' loss.

Chow et al. [6] proposed a source diversity protocol for video streaming applications over mobile ad hoc networks (MANET). Their protocol is based on the use of multiple description coding (MDC) scheme, where each user sends a description of the video. It is the responsibility of the requester to specify the number of sources and which video description coded with MDC they should send. As for the routing, the authors proposed an extension of the dynamic source routing (DSR) protocol in order to support multipoint-to-point video transmission [6]. Their routing extension consists of comparing the routes and favoring the routes with minimal shared nodes. However, this approach suffers from significant control overhead given that the requester is fully responsible for determining the optimal disjoint routes for each source. Moreover, this approach induces an important routing load due to the creation of backup routes.

Xu et al. [75] designed a peer selection technique based on mobile agent and a super peer selection technique based on trust aiming at improving the QoS in P2P media streaming systems and at decreasing the client's startup delay. For the peer selection technique, each client in the system creates a control mobile agent (CA) that is responsible for answering the client's requests using a search agent (SA). As for the super-peer selection technique, the authors calculated a trust value for each client and chose the one that has the highest value.

2.5.3 3D Streaming Selection Techniques for Wired P2P Networks

Several supplying partner protocols [2, 4, 23, 24, 33] have been implemented. However to the best of our knowledge, none of the already proposed protocols is intended for 3D streaming over thin mobile devices.

For 3D streaming supplying partner techniques, one may think of an obvious solution using the distributed hash table (DHT), where a list of peers and the 3D objects held by each peer, is maintained. However, DHT is considered very expensive to maintain and produces significant control overhead [79]. Hu et al. [2] proposed a naive query-response method to request 3D content, and chose randomly the source from the neighbors that replied positively. Only few query messages are sent in order to avoid the flooding. This approach is considered effcient given that it provides a local load balancing due to the sources random selection. However, this same advantage, i.e., random selection of sources, causes latency since the users have to piggy-back before requesting the data [4], which also leads to communication overhead.

Sung et al. [4] proposed a source selection strategy based on an active periodic exchange of messages to inform about incremental content availability and a multi-level AOI request in order to reduce the request contention. Each AOI is divided into several concentric levels where each level that is close to the center has higher request priority. When data is needed, peers at high levels are requested first, and if more than one peer is available, then one supplying partner is chosen randomly. By adapting this technique, Sung et al. [4] try to make data requests close to requesters, leading to the request contention avoidance. As for the information exchange messages related to the content availability, they allow requesters to quickly locate sources and to prevent therefore long time requests. However, this technique has several drawbacks. First, the information exchange messages, sent to inform about incremental content availability, lead to communication overhead. In order to minimize the number of messages sent, the protocol avoids informing about data cache removal. Nevertheless, this may lead to invalid requests. Second, this approach induces long data acquisition times given that requesters have few supplying partners in their high level AOIs. And third, this technique does not take into consideration the source's available bandwidth [24].

Cheng et al. [79] proposed making a super peer responsible for selecting a potential source for a given requester. Potential sources are grouped according to the hierarchical structure of the 3D mesh, and one peer, called the super peer, is responsible for maintaining the hierarchy. When data is needed, the requester contacts the super peer who chooses a source. This technique obviously induces significant communication overhead. Moreover, the hierarchical P2P lookup suffers from control overhead and latency.

Chien et al. [24] proposed BAPS, a 3D streaming supplying partner strategy based on the bandwidth availability of sources. The main objective of their work is to improve the streaming quality by ameliorating the bandwidth utilization and avoiding peer over-

loading and request contention. In their work, Chien et al. [24] do not limit the list of potential sources to only AOI neighbors, but also include the bandwidth factor during the selection process. The list of potential sources includes peers that have sufficient bandwidth to stream 3D data. Requesters select their sources by giving priority to those that have a larger upload bandwidth. One of the main disadvantages of this technique lies in the fact that the list is not updated, which is considered inefficient due to the random mobility of the client. Moreover, the list of available sources is only maintained at the server, which leads to the server bottleneck or makes the system inefficient if the server brakes down.

Cavagna et al. [23] proposed a P2P based 3D streaming scheme for urban scenes which they refer to as level of detail description tree (LODDT). Each peer has an LOD description of an area and a self adaptation mechanism that aims at adapting its upload and download bandwidth. The root peer holds the entire environment with its roughest details. As the levels increase, more details are added to the view. Peers periodically send their Time to Serve (TTS) in order to make the supplying partner selection easy. When a given peer needs data, a list of its neighbors, with their TTS, is returned. The peer estimates the content held by its neighbors based on their coordinates and selects the one with the least TTS. However, the selected supplying partner may be new, and may thereby not have the required information. This may cause latency due to invalid requests, or may cause the request contention problem if many peers choose the same supplying partner [4]. Royan et al. [33] addressed the latter issue, i.e., unserved requests. They proposed a streaming technique, called the Progressive building tree (PBTree) based on a tree structured model description where each tree node contains the 2D 1/2 model of an object. Each tree node is responsible for selecting its supplying partner based on several criteria including the number of peers currently served, or the realization ratio which is the percentage of successfully served requests.

Chapter 3

Location and Resources based Supplying Partner Selection Protocols

In this chapter, we shall describe our proposed P2P based 3D streaming system ovr thin mobile devices which we refer to as MOSAIC. It comprises the following two supplying partner protocols: the location based supplying partner protocol and the resource based supplying partner protocol. Both protocols shall discover multiple supplying partners while minimizing the peer's 3D data acquisition time. Before we proceed further, we wish to describe in details first our system model, then discuss the main components of our MOSAIC system.

3.1 System Model

MOSAIC system is intended for augmented reality (AR) applications using thin mobile devices for an emergency preparedness and responses class of applications. In such environment, the real world is mapped into the virtual world. Therefore, if objects are proximate in the real world, they are also considered proximate in the virtual environment (VE)[1]. Each person in the real world holds a thin mobile device, that is used to navigate the VE. These persons are represented by mobile peers in the VE and use the information streamed to the mobile device to navigate wihin the VE.

The mobile devices that can be employed in our scenario are PDAs, and iPhones

[1]In the rest of the paper, we refer to proximity in the VE as proximity for short.

among others. They are homogeneous devices with the same transmission range. These devices have the ability to be location-aware, i.e., capable of detecting their 3D co-ordinates in the real world, which are then mapped to determine the corresponding coordinates X, Y and Z in the VE.

Since we are dealing with thin mobile devices, bandwidth is an issue to take into consideration, due to the limited capabilities of these devices. In order to lessen the bandwidth consumption, our design reduces the propagation of unnecessary messages using an interest management technique assisted by a location-based filtering of messages. We divide the VE into fixed, well defined regions, whose sizes are determined at the beginning of the implementation. Each region has a server which we refer to as region master (RM), responsible for it. It holds a complete view of the VE part in that region and is responsible for (1) keeping track of mobile peers in its area, (2) providing the required data to requesters when needed including the list of objects within their area, and (3) maintaining important state updates. The RM also holds the scene description of its area and is responsible for sending the description to the peers when they first join the scene.

Each region is then subdivided into well-defined, fixed sub-regions. The size of the sub-region is also defined at the beginning of the implementation. Each sub-region constitutes a scene composed of 3D objects represented by 3D meshes and their associated textures. The second subdivision was made in order to reduce further the number of messages processed by each mobile clients. Figure 3.1 illustrates the system model of MOSAIC.

Given that each region is associated to an RM, it is the responsibility of the RM to deal with the mobile peers join and leave operations. Moreover, when a given mobile peer gets close to the borders of an adjacent region, it is also the responsibility of the RM to provide the required information. In this case, and based on the mobile peer's coordinates and direction, the corresponding RM contacts the new region's RM asking for potential relevant 3D data. The RM keeps the data in its buffer to stream it to the mobile peer if its AOI starts exceeding the borders of the actual region. The transfer of region is smoothly done. Once the mobile peer moves to a new region, it automatically stops receiving and processing updates from the previous region.

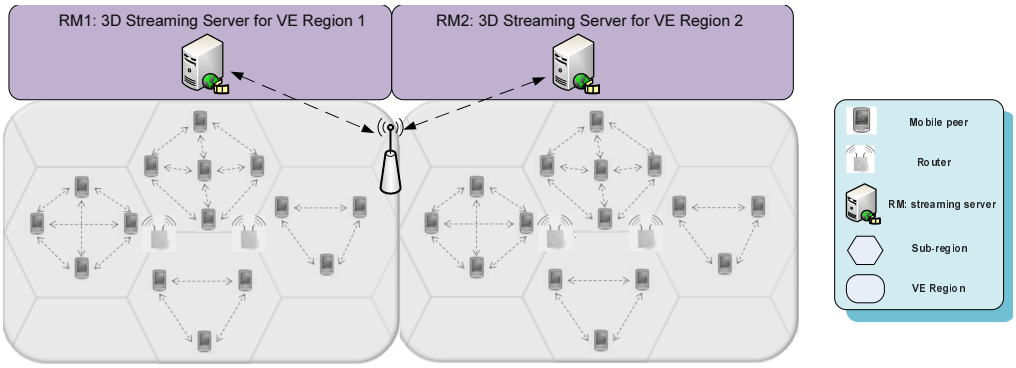

Figure 3.1: System model of MOSAIC

3.1.1 Adopted Interest Management (IM) Technique

Beside the double partitioning technique that has been applied in MOSAIC, a second IM technique, AOI, is adapted at the peer's level. Given that the VE is large, the mobile user does not need to acquire all of the 3D objects. In our design, each mobile peer has an AOI determined based on its visibility and that specifies its data of interest. Therefore, only objects that are within the peer's AOI are processed and streamed, reducing, therefore, a significant amount of processing time and network load. We propose to have the mobile peer's AOI with a spherical shape, where the mobile client is the center of the sphere. We chose this AOI shape, in order to allow the mobile client to have a 3D view of the VE and not to be limited to a planar view.

The AOI's radius which we refer to as R_{AOI}, should be dynamic and proportional to the available resources of the mobile device. We believe that a fixed high value may result in the streaming of more data to the mobile clients, which leads to the resources over-utilization. In the opposite case, when R_{AOI} receives a small value, the user will have a very narrowed view of the VE, given that its AOI does not cover a large VE surface. In our work, we propose to have the R_{AOI} expressed in terms of the mobile devices *residual energy* and its *available memory*. We believe that these two factors are the most affected by the streaming activity. In fact, while navigating the VE, the mobile device's energy is greatly consumed since it has to process, render and display the 3D objects. Moreover, the mobile device's memory is also decreased given that the user has to keep the last acquired 3D objects in its buffer to stream them to requesters when needed. Therefore, we believe that the AOI's radius R_{AOI} is highly dependent and proportional to the device's residual energy $Ener_{Device}$ and its available memory

Mem_{Device} as illustrated in (3.1). It increases when more resources are available and decreases otherwise.

$$R_{AOI} = \xi Mem_{Device} + \sigma Ener_{Device} \quad 0 < \xi, \sigma < 1 \tag{3.1}$$

In MOSAIC, we do not focus on object selection strategies. We assume that, based on the list provided by the RM and the coordinates of the 3D objects, the mobile peer knows which 3D objects are covered by its AOI and that need therefore to be acquired.

Each mobile device contains two buffers: A $Main_{Buffer}$ used to keep a copy of the 3D objects included within the peer's AOI, and a secondary buffer, which we refer to as $Hold_{Buffer}$, used to keep 3D objects for a very short period of time. We set the size of the mobile device's $Main_{Buffer}$ to 50 MB [58], given that our design deals with thin mobile devices such as PDAs. Two indices, which we refer to as $ID_{ObjFirst}$ and $ID_{ObjLast}$, represent the sequence numbers of the first and last objects in the $Main_{Buffer}$ respectively. The 3D objects kept in the $Main_{Buffer}$, are ordered based on their sequence number. This assumption has been added in order to facilitate the content availability verification during the supplying partner selection.

Each mobile peer tries to keep the 3D objects in its buffer, in order to stream them to requesters when needed, and to avoid therefore requesting the RM server. When the storage amount in the buffer decreases, a random number of packets, starting from the oldest packets in the buffer, are selected and erased, in order to make space for new packets.

3.1.2 Dead Reckoning

Given that we are dealing with thin mobile device over wireless networks, network bandwidth limitation is a constant challenge to take into consideration. In order not to overuse the network, one could minimize the number of messages transmitted across the network. Several techniques, such as the IM techniques, can be applied for this purpose. Another technique that greatly minimizes the bandwidth utilization is the dead reckoning technique that aims at predicting the future state of a peer given its actual state. In our design, we define the state used by the dead reckoning as the position. Therefore, in MOSAIC, every mobile peer tries to estimate the future positions of its mobile neighbors based on their direction and actual coordinates. This technique has been applied in order to minimise the frequency of updates transmitted, which consequently, lessens the number of messages sent and therefore leads to less bandwidth consumption and to the congestion avoidance.

3.2 System and User Requirements

3.2.1 The System's Requirements

From the system's perspective, one of the main requirements is the system's *efficiency* that can be expressed in terms of the *network bandwidth utilization*. Given that wireless networks are used, the network bandwidth is very important. Sending a high number of messages over the network, or flooding it, overuses the network bandwidth. While minimizing the number of messages sent over the network, preserves the network bandwidth and makes the system more efficient. Our objective is therefore to optimize the efficiency of the system by avoiding the bandwidth over-utilization.

Another important requirement to take into consideration is the system's *performance* that can be captured using concepts such as *response time*, *data delivery*, *load balancing* and *scalability*. From the data delivery perspective, the main objective is to favor the P2P delivery over the RM (server) delivery. This avoids the server bottleneck, since data requests are distributed among users who will be responsible for transmitting the relevant 3D data. Once the server bottlenecks are avoided, peers acquire data faster which greatly improves the system's response time as well as the system's performance. Moreover, the P2P delivery improves the system's scalability allowing the system to have a high number of users while keeping a good performance. As for the load balancing, it denotes the amount of streaming work performed by each peer. Our system should balance the load among users and avoid having overused sources.

The main goal of MOSAIC is to provide efficient techniques for the selection of the best source holding the relevant data. This process includes the *source discovery* and the *source selection*. The former denotes the technique used to discover the mobile users with the same interests. Most of the time, these users are AOI neighbors that have a similar view of the VE and that have a high probability to hold the required 3D data. As for the source selection, it designates the mechanism used to select the best source and the factors taken into consideration during the selection. These factors can include the 3D content availability, and the mobile device's resources, just to mention a few.

A requirement that the system has to take into consideration is the *coverage*. This expresses how to maintain serving a giving peer when it is traveling from one area to another. The main objective is to have the system's coverage efficient, smooth and transparent, i.e., the user does not notice any change while traveling from one area to another.

3.2.2 The User's Requirements

From the user's perspective, the most important requirement to take into consideration is the *acquisition time* that denotes "*how fast*" the mobile user acquires data. The acquisition time is closely related to the system's streaming quality: A high acquisition time indicates latency which degrades the system's streaming quality since the 3D object takes a long time to be displayed on the user's mobile device. The latency can be caused by many factors such as network congestion, request contention, invalid requests, just to mention a few. One of the main objectives of our design is to optimize the streaming quality, as well as minimizing the data acquisition time, i.e., minimizing the latency.

3.3 MOSAIC System

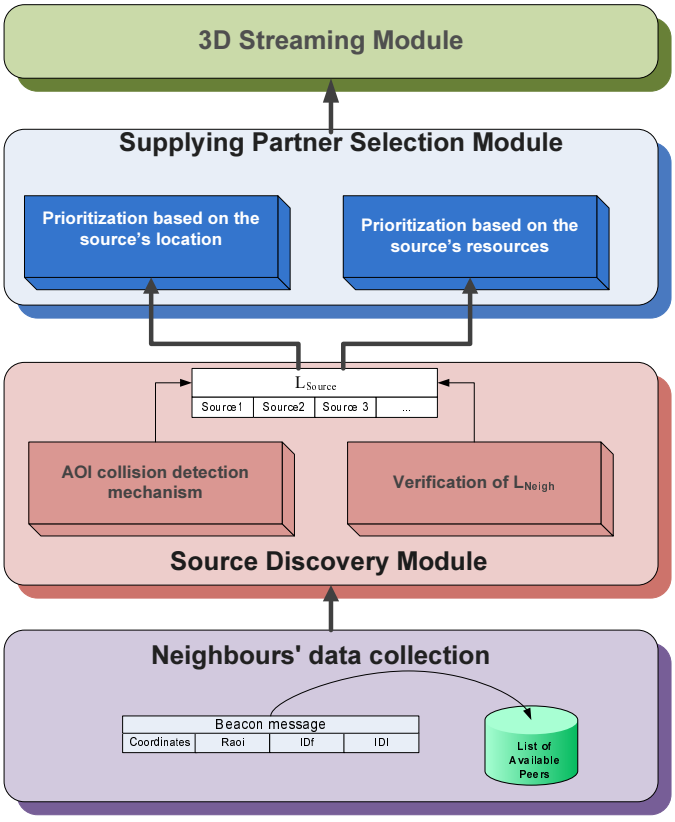

Figure 3.2: Architecture of MOSAIC

In Figure 3.2, we illustrate the main components of MOSAIC's architecture. MOSAIC is constituted of these following modules:

1. *The neighbors data collection module* aims at creating a list of available neighbors (L_{Neigh}) in the area. L_{Neigh} is updated every interval of time refered to as T_{phase} and is computed using the beacon messages exchanged among peers.

2. *The source discovery module* allows a given requester to discover potential supplying partners. This module is based on the execution of two mechanisms: the AOI collision detection mechanism and the List of neighbors verification mechanism.

3. *The supplying partner selection module* allows a given requester the selection of multiple supplying partners thereby providing a fast 3D data acquisition time.

4. *The 3D streaming module* allows each peet to stream 3D data to its neighbors when required.

Before we proceed further, let us first present the packet's header of the beacon message that has been modified in order to include additional fields. Figure 3.3 illustrates the beacon message packet format used in our MOSAIC system.

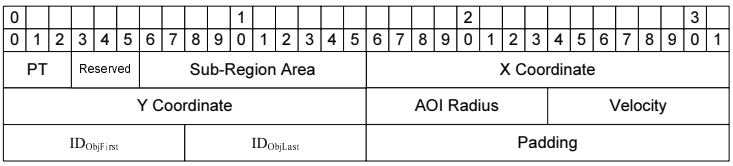

Figure 3.3: Beacon message packet format

- *Type*: indicates the type of message received and processed. In our design, we have 6 possible values for the *Type* field, namely: 1 for a beacon packet, 2 for an area packet, 3 for a data packet, 4 for a request packet, 5 for an acknowledgment (ACK) packet and 6 for a NACK packet.

- *Area*: indicates the VE region of the mobile peer. As discussed, the VE is partitioned into regions, in order to minimize the computation overhead. Therefore, upon receiving a beacon message, each mobile peer examines the Area field. The packet is discarded, if the field indicates an area different from the mobile peer's own area, and is processed otherwise.

- *Coordinates*: These fields represent the X and Y coordinates of the mobile peer. In our design, each mobile peer is aware of its position in the VE and includes its

actual X and Y coordinates in the broadcasted beacon message. This information is used by neighboring mobile peers for their source discovery mechanism.

- *AOI radius* R_{AOI}: designates the AOI radius of the mobile peer. As specified earlier, R_{AOI} is dynamic and is proportional to the mobile device's resources (memory and residual energy).
- *Velocity*: specifies the actual speed of the mobile peer.
- $ID_{ObjFirst}$: indicates the sequence number of the first packet in the mobile device's $Main_{Buffer}$. As mentioned earlier, all 3D data packets are kept in an ordered way in the $Main_{Buffer}$. Therefore $ID_{ObjFirst}$ represents the smallest sequence number.
- $ID_{ObjLast}$: This field indicates the sequence number of the last packet in the mobile device's $Main_{Buffer}$. It represents the highest packet's sequence number in the $Main_{Buffer}$.

Let us now present the modules of MOSAIC in details.

3.3.1 Neighbors Data Collection Module:

The neighbors data collection module allows a given requester to generate its set of available neighbors in the area. For this purpose, each mobile peer broadcasts a beacon message to the mobile peers within its radio range. Only the neighboring peers, that are within its region, process the received beacon messages. All of the remaining peers discard the update message. Beacon messages are sent periodically every T_{phase}, in order to minimize the number of messages sent through the network, and thereby minimize the communication overhead. Based on the information contained within the beacon messages received, each mobile peer keeps track of its neighbors, i.e., those located within its region, and creates its own list of neighbors which we refer to as L_{Neigh}. Each beacon message is also routed to the area's relative region manager (RM) in order to allow the latter to monitor the positions of mobile peers within its region.

3.3.2 Source Discovery Module:

The source discovery module allows a given mobile peer to discover potential 3D supplying partners and to create a list of sources which we refer to as L_{Source}. The source discovery module is based on the execution of the following mechanisms: (i) the AOI

Table 3.1: Parameters of the AOI Collision Detection Mechanism

Parameter	Description
P	Requester mobile peer
L_{Neigh}(P)	List of neighbors for the mobile peer P
N_i	Neighbor of P and element of L_{Neigh}(P)
$L_{collision}$(P)	List of colliding neighbors for peer P
(X1, Y1)	Coordinates of P
(Xi, Yi)	Coordinates of N_i
R_{AOI1}	AOI radius of P
R_{AOIi}	AOI radius of N_i
dist(P, N_i)	The distance that separates P and N_i

collision detection mechanism, (ii) the measurement of the area of intersection of AOIs, and (iii) the verification of L_{Neigh}. Note that (ii) and (iii) are run concurrently.

(i)The AOI Collision Detection Mechanism

At the beginning of the source discovery phase, each mobile peer applies the AOI collision detection mechanism that determines which of its neighbors' AOIs collide with its own AOI. The AOI collision detection mechanism is based on the information contained in the beacon messages received, and more specifically, the X,Y coordinates and AOI radius R_{AOI} of the neighbors (Refer to Table 3.1). The mobile peer computes the distance separating him from its neighbor using Equation(3.2) and determines wheter a collision exists or not based on Equation(3.3). Upon executing the AOI collision detection mechanism, each mobile peer creates a set of colliding neighbors which we refer to as $L_{collision}$. For a given mobile peer, $L_{collision}$ contains all neighboring peers whose AOI spheres overlap with the requester's AOI and that have a high probability to possess the relevant 3D data as illustrated in Figure 3.4. Indeed, in VE, neighboring peers can be suppliers for each others because they have a similar view of the VE and may thus hold the relevant 3D data.

$$dist(P, N_i) = \sqrt{(X1 - Xi)^2 + (Y1 - Yi)^2} \tag{3.2}$$

$$N_i = \begin{cases} \in L_{collision} & \text{if } dist(P, Ni) < R_{AOI1} + R_{AOIi} \\ \notin L_{collision} & \text{if } dist(P, Ni) > R_{AOI1} + R_{AOIi} \end{cases} \tag{3.3}$$

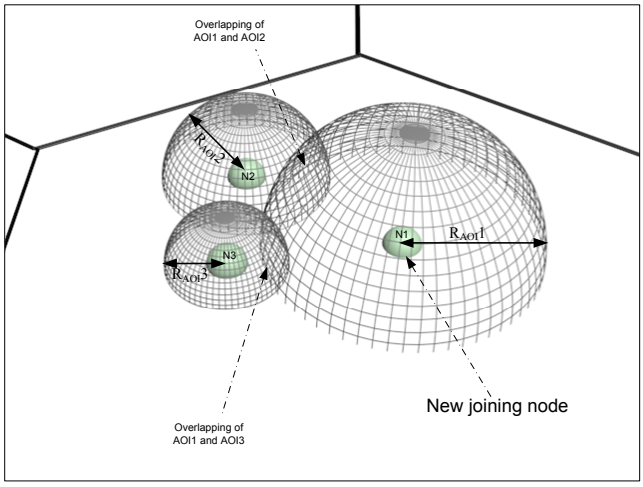

Figure 3.4: Example of Overlapping AOIs

In Figure 3.4, AOI of Node 1 overlaps with Node 2 and 3 AOIs. Therefore, Node 2 and Node 3 are added to the $L_{collision}$ of Node 1.

(ii) Generation of the List of Sources

The list of sources (L_{Source}) is generated based upon the estimation of the area of intersection of AOIs and the verification of L_{Neigh}. For each element in $L_{collision}$, the requester calculates first the area of intersection $Area_{Intersect}$ that is estimated based on Equation (3.4), and second the percentage of intersection wich we refer to as $Percentage_{Intersect}$ estimated based on Equation (3.5). All of the mobile neighbors with whom $Percentage_{Intersect}$ is greater than a predefined threshold, are added to the list of potential sources L_{Source}.

Note that the $L_{collision}$ list may not contain enough peers whose AOIs collide with the user's AOI, which may limit the sources' choice of a given requester. In this case, the requester would ask its RM in order to acquire the relevant 3D data. To address this issue, the requester will check its list of neighbors before requesting its RM. Some neighbors may indeed have the relevant 3D data, however they may not have been selected in the set of colliding peers ($L_{collision}$). This may be due to the fact that their AOIs do not collide with the requester's AOI. Therefore, in this step, the peers whose main buffer information (sequence number of the first and last 3D packets in the main buffer) indicate that they hold the required 3D data are added to the list of sources.

$$Area_{Intersect}\{O1,O2\} = \frac{1}{2}\,\widehat{CO_2D}\,{R_2}^2 - \frac{1}{2}\,{R_2}^2\sin(\widehat{CO_2D}) + + \frac{1}{2}\,\widehat{CO_1D}\,{R_1}^2 - \frac{1}{2}\,{R_1}^2\sin(\widehat{CO_1D}) \quad (3.4)$$

$$Percentage\{O1,O2\} = Area_{Intersect}\{O1,O2\}/AOI_{O2} \quad (3.5)$$

3.3.3 Supplying Partner Module:

In this phase, we aim at selecting the supplying partners, i.e., the peers that own the relevant 3D data needed. As opposed to previous studies [2, 4, 33], in our design, we do not rely on a sole supplying partner. The requester has the possibility to select multiple supplying partners that can deliver the requested data they might have, which may result in a faster data acquisition time when compared to the single supplying partner providing the data. The supplying partner selection phase is based on the selection of sources among the L_{Source}. We propose two supplying partner protocols: the location-based and resources-based supplying partner protocols. Each protocol takes into account different factors during the source selection.

3.4 Location-based Supplying Partner Protocol

In this section, we shall describe our proposed location based-supplying partner selection strategy which we refer to as LB3D. When dealing with 3D environments, proximity is manifestly the criterion to take into account, given that close peers will have nearly the same view of the VE, and will likely need the same 3D data. We believe that if two peers' AOIs overlap, they are then proximate and they have a similar view of the VE. Both peers can therefore be considered as suppliers for each other and stream 3D content to each other when needed. Hence, in our proposed location-based supplying protocol, supplying partners are selected based on their locations in the VE and their proximity to the requester. To accomplish this, we prioritize sources in the L_{Source} list based on their proximity to the requester first and the amount of data they hold second. In other words, a two-level prioritization approach is used. First, sources are ordered based on how close they are to the requester, i.e. whether their AOIs overlap or not. Then a second prioritization is applied to sort the sources depending on the amount of required

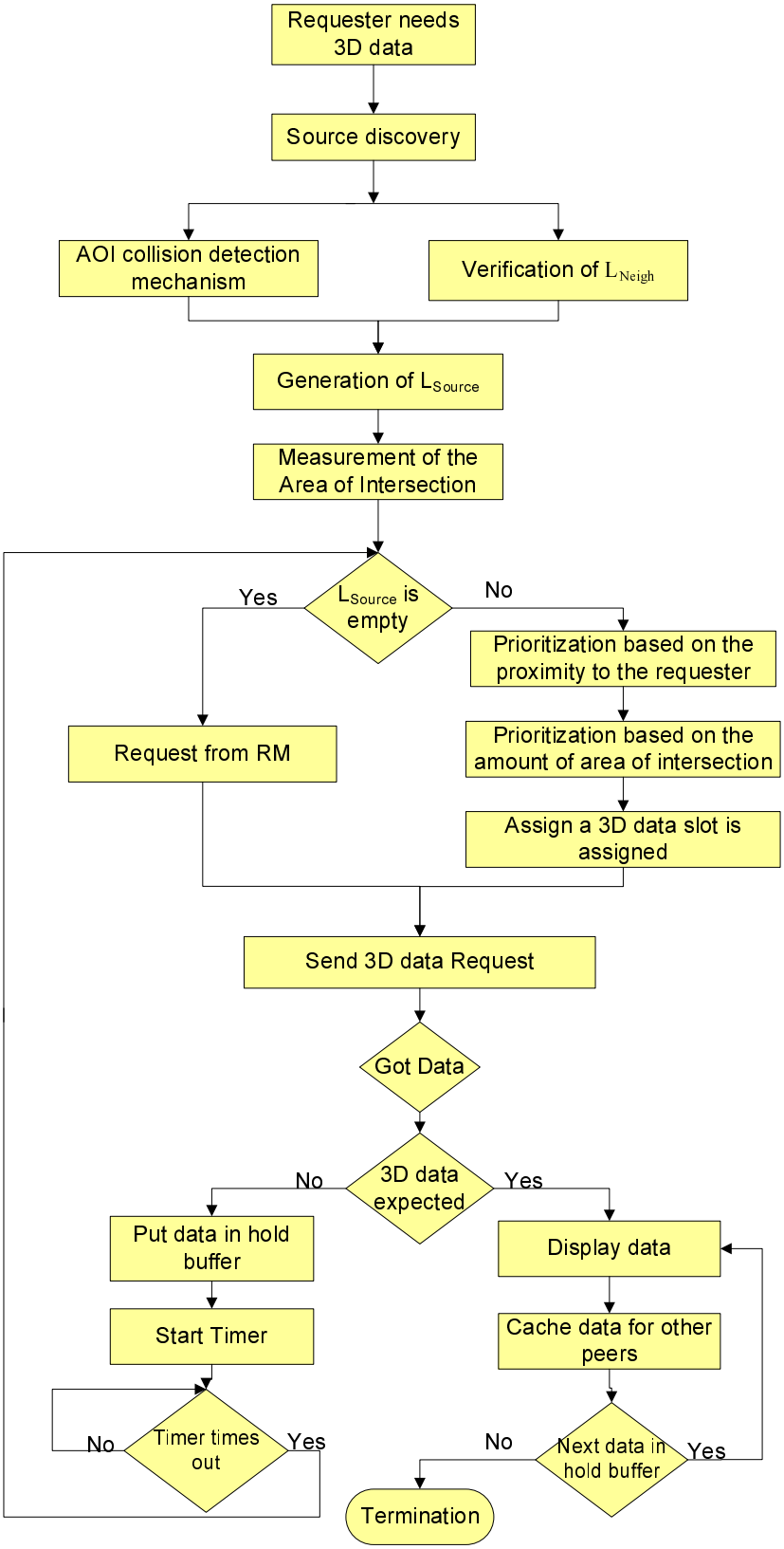

Figure 3.5: LB3D's flow diagram

3D data and its availability. Once the L_{Source} list is sorted, the requester distributes its requests among the supplying partners. In the case where the list of sources L_{Source} has more than one peer, then depending on the amount of required 3D data and its availability on each source, the requester tries to distribute its data requests among the sources asking each source to send a part of the relevant 3D data. Upon receiving the required 3D data, the requester displays it on its mobile device and keeps it in its buffer to be streamed to other neighbors when needed. An illustration of the required steps for the supplying partner phase is presented in Figure 3.5.

The detection of the lost packets is basically the responsibility of each requester. This is accomplished by finding an out of order packet. Once the out of order packet is received, the requester starts a timer and sends a NACK message with the missing 3D data packet's sequence number to the corresponding supplying partner. When the corresponding supplying partner fails to resend the required 3D data, the requester examines the 3D data availability on the remaining peers of L_{Source} and requests it from them. The RM acts as the final 3D data supplier in case the peers are unable to fulfill the data requests.

Our proposed LB3D protocol has several features: (1) stable and continuous 3D streaming, (2) remote access cost avoidance, (3) invalid requests avoidance and (4) quick 3D data acquisition time. In our design, we aim at ensuring a stable and continous 3D streaming service with a good streaming quality. This is achieved by avoiding long waiting times and congestion due to the sole supplier bottleneck and the request contention. Moreover, we aim at avoiding the remote access cost, by having the requester obtaining 3D data from its AOI neighbors instead of asking its region master. The latter will only be responsible for providing the required data, if neighboring peers are unable to fulfill the mobile peer's requests. Furthermore, given that the requester has the ability to select multiple supplying partners, the 3D data acquisition time is significantly minimized.

One apparent issue that needs to be taken into consideration during the supplying partner selection is the frequency of source selection where a given source can be frequently selected. This will obviously consume the peer's resources and dissipate its energy. In evidence, when dealing with thin mobile devices with limited capabilities, this issue becomes critical since it is considered a major cause for performance degradation. Therefore, a resource-based supplying partner protocol is proposed and will be discussed in the next section.

3.5 Resources-based Supplying Partner Protocol

Resources utilization is a crucial issue that needs to be taken into consideration when dealing with 3D streaming over thin mobile devices. Recall that thin mobile devices, such as PDAs or smart phones, have limited resources and processing capabilities. If resources' consumption is not taken into account during the supplying partner selection, it will result in some peers that are holding the mobile device, being frequently selected as suppliers. This overloads the selected sources and overuses their resources. In order to prevent this from happening, we propose a resources-based supplying partner protocol which we refer to as RB3D. Our proposed solution is to take into account the available resources on each supplier, while relieving the overloaded sources and balancing the load among others.

3.5.1 Load Estimation

Our proposed protocol, as illustrated in Algorithm 1, aims at reducing the load on the supplying partners, especially those that are very active, i.e., frequently selected to stream data to requesters. We define the load as the amount of 3D streaming that has been performed. The load is obviously affected by the average number of served data requests. In the RB3D protocol, we express the load of a given source in terms of its realization ratio (RR%) which denotes the percentage of data requests successfully served. RR% is estimated at the end of each phase T_{phase} and gives an idea about the load that a given supplying partner received during the precedent phase. At the beginning of T_{phase}, each peer sends a beacon message including its information (coordinates, velocity, AOI radius, etc.). We modified the packet header to add the $RR\%$ field which indicates the amount of load received during the previous T_{phase}. Upon receiving the beacon messages, a given requester discovers its sources at the sources discovery phase, generates its list of sources L_{Source} and analyzes the load on each source before distributing its 3D data requests. A source is considered overloaded when it has a high percentage of served requests, i.e., a high RR%. Overloaded suppliers badly affect the system performance since requests queue up at the sources which results in delay and high response time. Therefore, the RB3D protocol avoids the selection of sources that served a large percentage of requests while favoring the response time.

Algorithm 1 Load Balancing Algorithm

Require: $L_{Source}(P)$
for Each element S_i IN $L_{Source}(P)$ **do**
  if $RR\%(S_i) > RR\%_{max}$ **then**
    $Pr(S_i) \leftarrow 4$
  else
    if $State_{Busy}$ $(S_i) == 1$ **then**
      if S_i IN $L_{Served}(P)$ **then**
        $Pr(S_i) \leftarrow 2$
      else
        $Pr(S_i) \leftarrow 3$
      end if
    else
      $Pr(S_i) \leftarrow 1$
    end if
  end if
end for
Distribute streaming among sources with Priority 1
if P still needs data **then**
  Distribute streaming among sources with Priority 2
  if P still needs data **then**
    Distribute streaming among sources with Priority 3
    if P still needs data **then**
      Distribute streaming among sources with Priority 4
      if P still needs data **then**
        Send request to RM
      end if
    end if
  end if
end if

Table 3.2: Parameters for the Load Balancing Analysis

Parameter	Description
P	The requester mobile peer
L_{Source}(P)	List of potential supplying partners for P
S_i	Potential supplying partner of P and element of L_{Source}(P)
$Size_{Source}$	Number of elements in L_{Source}(P)
L_{Served}(P)	List of sources that already served P before
RR%(S_i)	Realization ratio for each source S_i
$State_{Busy}$	Flag set to 1 when the source S_i is *Busy*
$RR\%_{max}$	Threshold value for realization ratio

3.5.2 Selection Procedure

Our proposed resources-based supplying partner protocol intends to distribute the load among the sources and to relieve the overloaded ones. Therefore, it is applied during the selection of supplying partners, more specifically after the source discovery phase and once the list of potential supplying partners L_{Source} is computed. In the location-based supplying partner protocol (LB3D), as described in Section 3.4, sources are prioritized based on their proximity to the requester, which can lead to the re-selection of the same sources. In the RB3D protocol, we propose to prioritize the potential supplying partners, and evenly distribute the load among them based on a set of rules which we will discuss later. To accomplish this, a set of parameters, as illustrated in Table 3.2, have to be taken into account during the decision making and the prioritization process.

Equation 3.6 explains how a given source is considered overloaded or not. We define $RR\%_{max}$ as a predefined threshold value for realization ratio. A source S_i is considered overloaded, when its realization ratio $RR\%$, exceeds $RR\%_{max}$.

$$State(S_i) = \begin{cases} Overloaded & if\ RR\%(S_i) >= RR\%_{max} \\ Not\ overloaded & if\ RR\%(S_i) < RR\%_{max} \end{cases} \tag{3.6}$$

In our protocol, we consider two main factors: the $RR\%$ of a given source S_i, and its *state*, denoted by $State_{Busy}$. The former illustrates the percentage of data requests successfully served and informs the requester if a given supplier is overloaded. While the latter, i.e., the state of the mobile peer $State_{Busy}$, denotes whether the source is at the time busy serving other requesters. We bring your attention to the fact that a busy source can still serve other requesters, if its $RR\%$ does not exceed the predefined threshold $RR\%_{max}$). Let us consider a mobile peer P that requires 3D data. Once the source discovery phase finished, P would have collected by then its list of potential supplying partners L_{Source}(P). In RB3D protocol, P assigns a priority (Pr) to each source

S_i of its L_{Source}(P), taking into consideration the parameters mentioned above. We chose to apply the following rules when assigning priorities:

1. Initially L_{Source} = { S_i | i = 1..Size$_{Source}$ }
2. If ∃ S_i, $RR\% > RR\%_{max}$, then $Pr = 4$ // Rule 1
3. If ∃ S_i, S_i is $State_{Busy}$ and $S_i \notin L_{Served}$(P),then Pr=3 // Rule 2
4. If ∃ S_i, S_i is $State_{Busy}$ and $S_i \in L_{Served}$(P),then Pr=2 // Rule 3
5. All remaing S_i, $Pr = 1$ // Rule 4

Each element S_i in the list L_{Source}(P) receives a priority (Pr) that indicates its turn to stream 3D data. Sources are then prioritized based on their assigned priority (Pr). In our design, the supplying partner selection is performed in the descending order of priority with priority $Pr = 1$ as the highest priority and priority $Pr = 4$ as the lowest one. In other words, suppliers that receive priority 1 are pulled first, and the streaming is distributed among them by assigning a 3D data slot to each of them. Once all suppliers with priority = 1 have been selected and the requester still needs 3D data, the suppliers with the next highest priority are selected, until all needed 3D data has been streamed. Obviously, the RM acts as the final 3D data supplier in case the peers are unable to fulfill the data requests.

Our proposed RB3D protocol shall first identify the overloaded sources. According to the first rule, sources that have a high realization ratio $RR\%$, exceeding the threshold value $RR\%_{max}$, receive the lowest priority: $Pr = 4$. These sources are considered overloaded, and RB3D, in this case, intends to relieve them by making them the last ones to be selected. Once the overloaded sources are identified, the remaining potential sources are considered able to stream data and to be selected as supplying partners without significantly overusing their resources. Nevertheless, priorities are still allocated based on few rules.

A priority $Pr = 3$ is assigned to suppliers who are busy serving other peers and who are not part of the list of sources that have already served the requester before, i.e., L_{Served}(P). These sources have their realization ratio (RR%) inferior to $RR\%_{max}$ and can therefore stream data to the requester. However, they are assigned the second lowest priority, because we believe that there is a low probability that these sources hold the required 3D data. Indeed, since these sources have not served P before, there is a high probability that they do not have the same interests as the requester P and may therefore not hold the required data.

Table 3.3: Simulation Parameters

Parameter	Value
World dimension	400m * 400m
Cell size	100m *100m
AOI radius	10.5m (max)
Node speed	1-20 m/s
Buffer size	50 packets
MM packet size	10000 Bytes
Simulation Time	2000 (s)
Radio Propagation Model	Two Ray Ground
Mac Protocol	IEEE 802.11
Antenna Type	Omni Antenna
Wireless bandwidth link	11Mbps
Mobile node RXThresh	70m
RM and Routers RXThresh	250m

In the third rule, busy sources that have served the requester P before, receive the second highest priority $Pr = 2$. We believe that if a supplying partner S_i had served the requester P before, it means that most probably both peers, S_i and P, have the same interests. Therefore, there is a high probability that the source S_i still have in its $Main_{Buffer}$ most of the 3D data required.

Finally, all of the remaining sources that were not assigned any priority, receive the highest priority, i.e., $Pr = 1$. These sources do not have a high $RR\%$, and are not busy serving other peers' requests. Therefore, they are considered the best candidate to stream the relevant 3D data, without affecting their resources. Selecting these peers as supplying partners for the requester P, will not overload them, and will guarantee a fast processing and a quick streaming.

In summary, RB3D aims at achieving a balanced distribution of 3D content, while minimizing the energy consumption, leading to the streaming performance improvement. Moreover, our mechanism should be able to guarantee, as we shall see in our explanations, long waiting times' avoidance and a fast processing given that the resources of the mobile device are not overused.

3.6 Experimental Results

In this section we shall present the performance evaluation of our proposed supplying partner protocols LB3D and RB3D. An extensive set of simulation experiments using NS2 simulator [76] with the settings presented in Table 3.3 has been carried out. Each simulation run has been repeated several times with a confidence of interval of 95%. In

our simulation experiments, nodes, who represent the users in the VE, have their initial position randomly processed and their speed newly generated after each phase T_{phase}.

The performance evaluations of both protocols will be as follow. First we introduce the evaluation of each protocol by itself highlighting their advantages. Then, a comparison of LB3D and RB3D is presented.

3.6.1 Performance Evaluation of LB3D

In this section, the performance evaluation of LB3D is presented. Recall that existing streaming approaches [2, 23, 33] are designed for P2P 3D streaming on wired networks, and do not support the requirements of 3D streaming over wireless networks-based applications. As such, they are not compatible nor comparable with our proposed approach. To the best of our knowledge, all of the existing 3D streaming approaches [16, 91] over wireless networks are based on the client-server architecture. Therefore, in order to show the benefits of LB3D, we compared it to a server-based supplying partner strategy for wireless networks such as the one designed in [12, 16].

In the first experiment, the performance of LB3D is evaluated by analyzing the streaming quality. This metric is important because it represents the quality of the 3D data displayed to the users. The streaming quality can be quantified using the data acquisition time metric indicating how fast the 3D object can be displayed on the mobile device. Figure 3.6 shows the data acquisition time as a function of the number of data request messages. As illustrated in Figure 3.6, we notice that the data acquisition time for LB3D is much lower than the server-based approach. This is mainly due to the fact that at the beginning of the simulation, most of the mobile peers still do not have enough 3D data in their $Main_{Buffer}$. Therefore, when a given peer requires 3D data, only few sources can provide it, which leads to the requests queuing up at the sources, and to a large data acquisition time. As the number of data requests increases, the data acquisition time for both approaches decreases. LB3D provides 40% of improvement in terms of data acquisition time, as compared to the server-based approach. This reduction in the data acquisition time is due to the increase of the number of supplying partners as the simulation runs, which gives more choices for requesters. The obtained results confirm our theoretical study regarding the fact that LB3D provides faster data acquisition since the requests are served locally by neighboring users instead of requesting it from the remote server.

The second set of simulation experiments aimed to evaluate the performance of LB3D

in terms of resources' consumption. As discussed, LB3D uses an interest management technique assisted by a location-based filtering of messages, in order to minimize the computation overhead. We compared LB3D with its two interest management techniques to a derivative of LB3D where no zoning technique is used and where each mobile node processes every update message received. We refer to the derivative used as NZM. Figure 3.7 illustrates the computation overhead when the simulation time varies. The computation overhead can be expressed in terms of the amount of processed messages. As shown in Figure 3.7, we can see a clear reduction in the computation overhead when applying LB3D as compared to NZM. When the simulation time is small, the computation overhead is very comparable when applying both protocols LB3D and NZM. As the simulation time progresses and increases, we can observe that LB3D reduces by 20% the number of processed messages as compared to NZM. Therefore by applying LB3D along with its interest management technique assisted by the location-based filtering of messages, LB3D succeeds in minimizing the computation overhead and hence the mobile device's resources consumption.

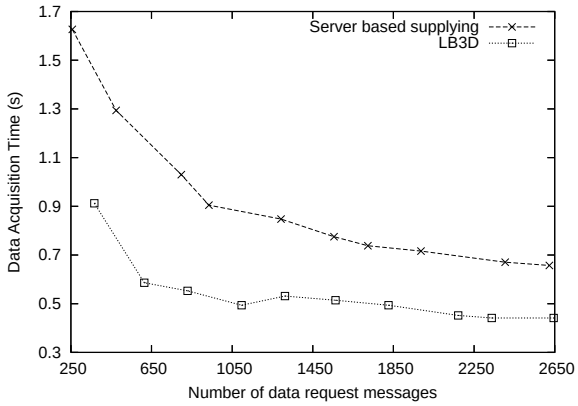

Figure 3.6: Data Acquisition Time

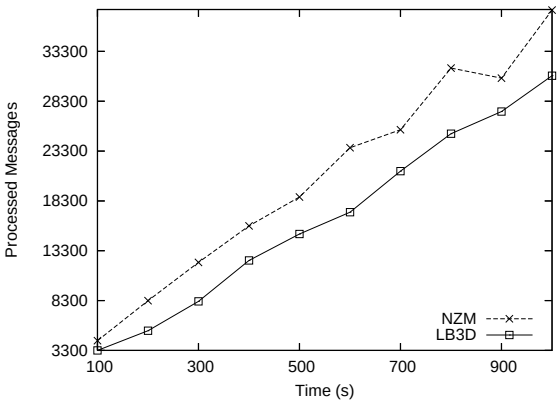

Figure 3.7: Computation overhead.

In the third set of simulation experiments, we focused on demonstrating the effect of T_{phase} on the protocol performance. To study the influence of T_{phase}, we tested the efficiency of LB3D in terms of bandwidth utilization. The latter can be expressed using the transmitted messages. As mentioned earlier, LB3D reduces the unnecessary messages' propagation, such as updates, by applying a dead reckoning technique, where updates are sent every interval of time T_{phase} and nodes' future locations are estimated based on the actual information. By minimizing the frequency of updates, the number of messages sent over the network is lessened, which leads to a decrease in the communication over-

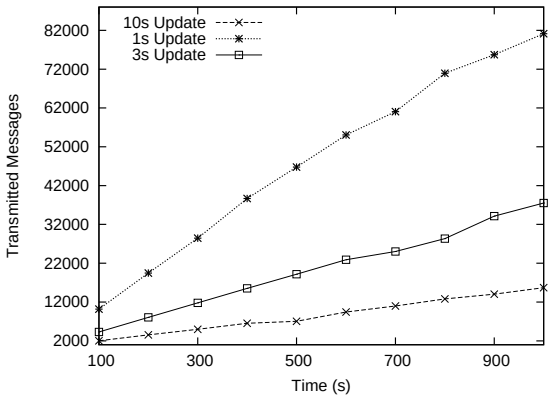

Figure 3.8: Communication Overhead as T_{phase} varies.

head and bandwidth consumption. In our experiment, we analyzed the behavior of LB3D when varying T_{phase} and we drew the communication overhead when the simulation time varies. We chose to illustrate only three values of T_{phase} which are 1sec, 3sec, and 10sec. For T_{phase} equal to 1 sec, update messages are sent every 1 sec and the dead reckoning mechanism is therefore removed. When T_{phase} is equal to 3sec, the dead reckoning technique works only for 2 sec; while the dead reckoning mechanism works for 9 sec when T_{phase} is equal to 10 sec. Figure 3.8 illustrates the simulation results of our experiment. As we can see in Figure 3.8, LB3D behaves very badly when the update messages are sent frequently (1 sec and 3 sec). The results obtained when T_{phase} is small illustrate a significant communication overhead. LB3D gives the best results when T_{phase} is large and equal to 10sec. It has a significant improvement of 80% and 60% respectively, in terms of transmitted messages, when compared to the results obtained when T_{phase} is 1 sec or 3 sec. These results confirm our assumption regarding the dead reckoning module that gives a clear reduction of bandwidth consumption and communication overhead and an evident congestion avoidance.

3.6.2 Performance Evaluation of RB3D

In this section, we shall present the performance evaluation of RB3D. We recall that RB3D aims at distributing the load among suppliers while taking into account their resources availability.

In the first set of experiments, the performance of RB3D in terms of load balancing is analyzed. RB3D protocol aims at evenly balancing the load among sources and ensuring that no sources are overloaded as compared to others. In other words, the load on sources

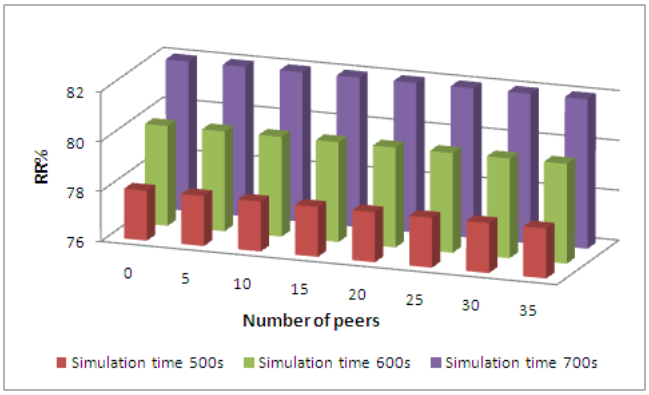

Figure 3.9: Histogram of RR%

should be comparable. In order to show the benefits of this feature, we measured the load per groups of sources. To accomplish this, we analyzed the average RR% for every group of 5 sources. We compared the results obtained for three simulation times: 500s, 600s and 700s. As illustrated in the histogram of Figure 3.9, the average of RR% obtained for each group of 5 sources for the simulation time of 500s turns around 78%.The average of RR% obtained for each group of 5 sources for the simulation time of 600s turns around 80% requests; while the average of RR% obtained for each group of 5 sources for the simulation time of 700s turns around 82% requests. Results obtained for the different simulation times are very comparable, which confirms that our proposed RB3D protocol succeeds in evenly distributing the load among peers.

In the next simulation experiments, we evaluated the performance of RB3D in terms of load distribution by examining how RB3D lightens the load on overloaded sources. For this matter, we measured the average number of requests served per priority. We recall that overloaded sources receive the lowest priority (Pr = 4), to stream data, since their $RR\%$ exceeds the predefined threshold $RR\%_{max}$. In Figure 3.10, we present the distribution of 3D data request messages per priority. As illustrated in Figure 3.10, sources with priority Pr = 1, are the ones that served the highest number of requests, whereas the overloaded sources have a considerably very low number of served requests. Therefore, most of the requests are assigned to suppliers with priority Pr =1, and only a few number of requests is sent to busy or overloaded sources. This confirms our thoughts regarding RB3D and its success to relieve the overloaded sources.

3.6.3 LB3D vs RB3D: A Comparative Study

In this section, we shall compare LB3D and RB3D illustrating the advantages of each. In the first set of simulation experiments, we analyzed the performance of RB3D in terms of load distribution. As mentioned earlier, the main goal of RB3D is to not overuse the mobile device's resources and not to have overloaded supplying partners as compared to others. A good way to evaluate the load distribution is by measuring the throughput's standard deviation of supplying partners. Note that the throughput's standard deviation expresses the variability of sources' throughputs. This metric allows us to analyze the load on each source and to determine whether there are sources overloaded as compared to others. When the throughput's standard deviation is high, it implies that the difference of loads between sources is high, whereas a low throughput's standard deviation implies that the load among sources is very comparable. We compared the performance of RB3D protocol to LB3D protocol when the simulation time varies. As shown in Figure 3.11, RB3D protocol has a stable throughput's standard deviation ranging from 8000 Bytes to 8200 Bytes. Note that the throughput's standard deviation is an average value, and that the 3D mesh packet size is 6000 Bytes. This implies that the throughput's standard deviation is very low. The LB3D protocol does not perform well in terms of load, and the difference in throughputs between sources exceed 14000 Bytes as the simulation time increases. RB3D provides better results than LB3D with a 45% of improvement.

In a second set of experiments, we analyzed the behavior of RB3D and LB3D in terms of response time when we vary the simulation time as illustrated in Figure 3.12. When the simulation time is small around 100s, LB3D and RB3D have similar response times. As we increase the simulation time, we can clearly see that RB3D behaves much better than LB3D with 30% of improvement. This difference is actually expected. In fact, RB3D tries to relieve the overloaded sources and distributes the load among the non-busy and non overloaded suppliers. Therefore, requests do no queue up at the source before being answered. However, in LB3D, suppliers are only selected based on their location and proximity from the requester. Therefore, a given source may be selected by different requesters and may thus be overloaded with requests queuing up, which obviously affects and increases the response time.

In the next set of experiments, the server overhead is illustrated. We analyzed this metric with variation in simulation time and plotted the results in Figure 3.13. As illustrated, RB3D behaves better than LB3D in terms of average requests answered by the server with 12% of improvement. This difference is mainly due to the criteria taken

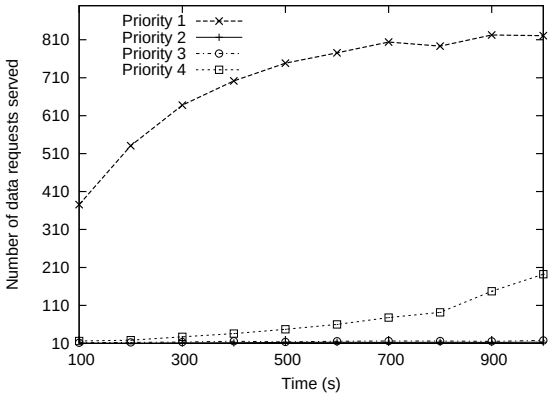

Figure 3.10: Distribution of 3D Data Request Messages

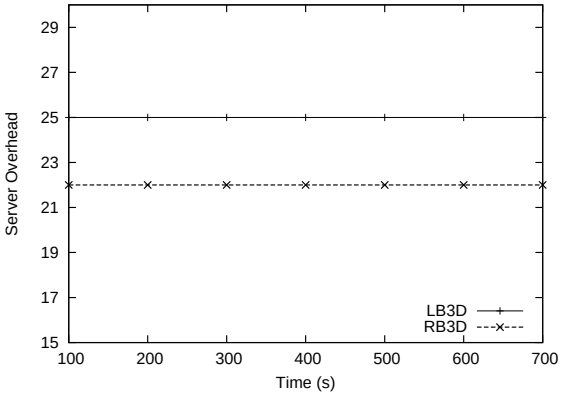

Figure 3.13: Server Overhead as simulation time varies

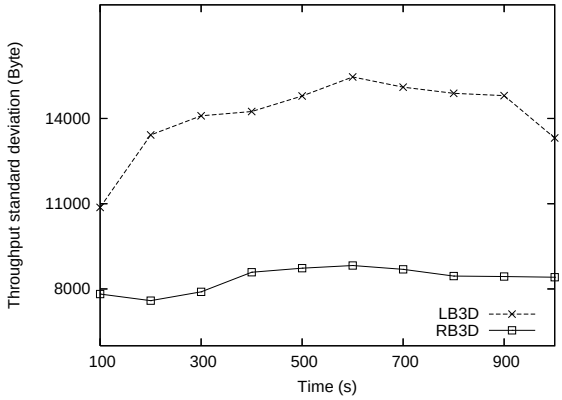

Figure 3.11: Throughput standard deviation

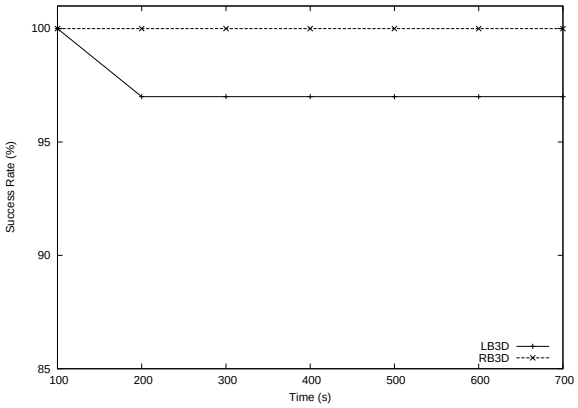

Figure 3.14: Fill Ratio

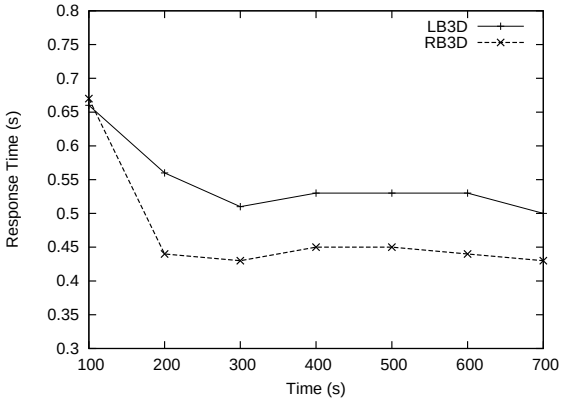

Figure 3.12: Response Time LB3D vs RB3D

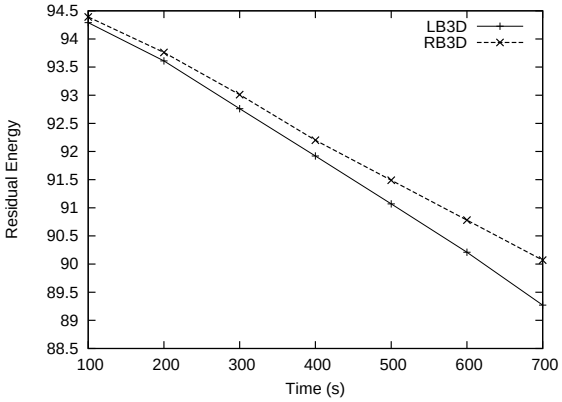

Figure 3.15: Energy consumption

by the protocols during the supplying partner selection. LB3D is location based, which means that if no sources exist in the area, the RM is responsible for streaming the relevant 3D data.

Figure 3.14 illustrates the fill ratio as the simulation progresses. As illustrated in Figure 3.14, we can see that RB3D performs better than LB3D with 5% of improvement. These results were expected. As a matter of fact, LB3D selects the sources based on their locations and does not take into account their streaming activities. The sources may be overloaded and the data requests may be dropped, which affects the fill ratio. However, RB3D tries to relieve the overloaded sources and tries to balance the load among the sources reducing the data loss and improving the fill ratio.

Finally, we analyzed the energy consumption when executing both RB3D and LB3D protocols. Figure 3.15 illustrates the residual energy on peers when the simulation time is varied. We set the initial energy of the mobile device to 95%, and we measured the residual energy. For a simulation time of 100s, the energy consumption is not very significant for both protocols and turns around 94.5%. As we increase the simulation time, the gap between the protocols is clearer and we can notice that RB3D consumes 5% less energy than LB3D. These results are expected. As a matter of fact, frequent 3D streaming overuses the supplier's resources and dissipates quickly its energy. RB3D tries to unburden the overloaded sources by distributing the requests among the remaining suppliers. Therefore, RB3D avoids a given supplier from being frequently selected, which saves its energy. As for LB3D, it does not take into account the supplier's resources during the source selection, which leads to the quick dissipation of the mobile device's energy.

3.7 Conclusion

In this chapter, we presented our P2P based 3D streaming system over thin mobile devices which we refered to as MOSAIC. Our system is composed of four modules including the neighboring data collection module, the source discovery module, the source selection module and the 3D streaming module. In the source selection module, we proposed two supplying partner protocols: location-based and resources-based supplying partner protocols. The former aims at choosing the best supplying partners taking into account their proximity to the requester. As for the resources-based supplying partner protocol, it intends to improve the streaming performance of the selected suppliers and balance the load among them by not overloading them and relieving the overloaded ones. Given that frequent 3D streaming overuses the mobile device's resources and dissipates quickly

its energy, it is crucial to take the energy demands into account when selecting potential sources. A lack of energy degrades the system performance considerably in terms of response time and impedes the P2P behavior. In the next chapter, we shall describe our energy management control for supplying partner selection protocols.

Chapter 4

Energy Management Control Mechanisms for Supplying Partner Selection Protocol

In this chapter, we propose an energy management control mechanism for the supplying partner protocol in P2P networks based 3D streaming. During the supplying partner selection, we wish to take into consideration the source's energy factor in order to minimize the energy consumption, and relieve the overloaded sources.

Our proposed protocol can be represented by a phase composed of 4 steps as illustrated in Figure 4.1: *update phase; supplying partner classification; supplying partner selection; and 3D streaming distribution.* At the begining of each update phase, each peer broadcasts a beacon message that contains information relative to its position, speed, buffer indices, and load. Only peers within the same sub-region process the received beacons in order to keep track of their neighboring peers' positions, their loads and the 3D data they hold. This information is collected and a list of neighbors (L_{Neigh}) and an initial list of potential sources (L_{Supp}) are computed. During the supplying partner selection step, the list of potential sources is sorted and each peer applies our proposed protocols 1L-EB and 2L-EB in order to build the final potential sources list. Both protocols will be described more in details in Sections 4.1 and 4.2. During the 3D streaming phase, requests are distributed among multiple sources in order to speed up the download time.

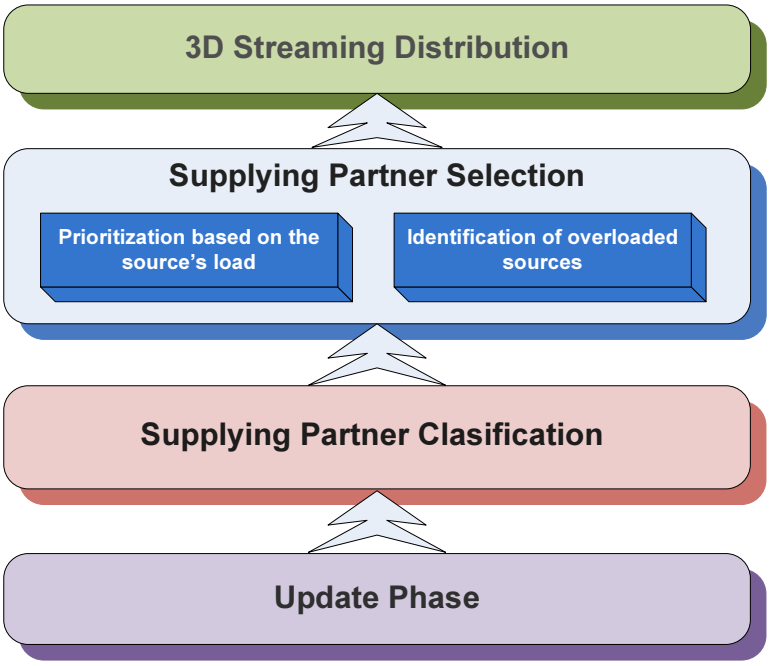

Figure 4.1: Phases of the Energy Management Protocols

4.1 1L-EB Supplying Partner Protocol

In this section, we introduce our one-level energy based supplying partner protocol *1L-EB*. We refer to this protocol as one-level, since it is only applied at the requester side. As we know, energy is an important and critical factor when dealing with P2P based 3D streaming over thin mobile devices. Our proposed protocol, 1L-EB, takes into consideration the energy constraint during the selection of potential sources, and aims to save the energy consumption of suppliers. In order to evaluate the energy consumption on potential sources, we first propose a new load estimator that is a function of the mobile device's energy, then we describe our proposed 1L-EB protocol.

4.1.1 Load Estimation

We define the load on potential sources as the amount of 3D streaming that has to be performed. The load is obviously affected by the the number of served requests and can be expressed as a function of the source's residual energy. We believe that both factors affect the mobile device's energy. On one hand, the higher the number of served requests, the lower the mobile device's energy will be. This is due to the frequent 3D streaming

that considerably decreases the mobile device's energy. On the other hand, when a potential source has low energy, then selecting it again to stream 3D data dissipates its energy even more.

There are two cases where a supplier is considered overloaded: First when the supplier has a high number of served requests, and second, when the source has low residual energy. Overloaded suppliers badly affect the system performance since requests queue up at the sources which results in delay and high response time. Overloaded supplier's energy is also dissipated faster, which shortens the source's energy lifetime and consequently affects the system's P2P behavior. This leads to the system becoming centralized where requests are served by the server, which could result in a server bottleneck.

In order to estimate the load on a given source, we propose a new metric where the load is expressed in terms of the mobile device's residual energy ($Consumed_{Energy}$) and its realization ratio as illustrated in Equation (4.4). We define $SR(n)$ as the number of served requests for the source n and Ren_n as the source's residual energy and the source's realization ratio (RR%) as the ratio of served requests by each source.

$$ASR(n) = \sum_{i=0}^{X} SR(n)/X \tag{4.1}$$

$$RR\%(n) = ASR(n)/Max_{ASR} \tag{4.2}$$

$$Consumed_{Energy}(n) = 1 - (Ren_n/Max_{Energy}) \tag{4.3}$$

$$Load(n) = \omega\ RR\%(n) + \mu\ Consumed_{Energy}(n) \tag{4.4}$$

In Equation (4.1), *ASR(n)* expresses the load tendency of the source n, and is calculated by the Average of Served Requests of n during the last X update phases. We believe that in order to estimate the load of a given source, we should take into account its load behavior for the last X update phases.

In Equation (4.2), we estimate the $RR\%$ that is a normalization of the $ASR(n)$ and that has therefore a value in [0,1]. In Equation (4.3), the load of the source n as a function of its residual energy Ren_n is estimated. We first express the percentage of residual energy. We then evaluate the $Consumed_{Energy}(n)$ as a function of the percentage of residual energy. The lower the residual energy, the higher $Consumed_{Energy}(n)$ will be. $Consumed_{Energy}(n)$ has been normalized in order to have a value in [0,1]. Both loads $Consumed_{Energy}(n)$ and *RR%(n)* are used to express the *Load(n)* of the source n as illustrated in Equation (4.4). Since $Consumed_{Energy}(n)$ and *RR%(n)* are both normalized, then *Load(n)* will have a value in [0,2].

We added weight factors ω and μ in order to study the influence of both loads when we vary ω and μ. The protocol can be then adapted based on the purpose of the application. If ω is 0 then the load is estimated only as a function of the source's residual energy. The protocol, in this case, relieves nodes with low residual energy and tries to increase their lifetime. When μ is 0, the load is only expressed as a function of the realization ratio $RR\%$. The protocol, in this case, tries to avoid the selection of sources that served a large number of requests by favoring the response time regardless of the source's residual energy. When setting both ω and μ to 1, our protocol favors the selection of sources that have a high residual energy with a low realization ratio.

In the following, we shall describe our one-level energy based (1L-EB) supplying partner protocol.

4.1.2 Description of the 1L-EB protocol

In our MOSAIC system, 1L-EB is applied once a list of potential sources is computed and the load on each potential source is estimated. Our proposed protocol is applied on the requester side and takes into account the energy constraint during the supplying partner selection. Instead of choosing its suppliers randomly from the list of potential sources, the requester distributes its requests based on information relative to the sources' loads. In our protocol, we aim to minimize the energy consumption of mobile devices. It is preferable to select a source that has high residual energy. In addition, the selected source should have a low $RR\%$ that denotes a low energy consumption due to low streaming activity.

Based on the load expression illustrated in Equation (4.4), the requester sorts the potential sources according to their loads. Sources are prioritized in an ascending order and requests are then distributed starting with sources with a minimum load. If the loads are the same for the potential sources, then they are sorted again based on their residual energy. In this case, sources are prioritized in a descending order; sources with higher energy are requested first.

In our design, the requester has the ability to select multiple sources and distribute the required data among them in order to speed up the acquisition time. By selecting the sources with lighter loads, our proposed protocol manages the energy consumption of the sources, distributes the load, and relieves the overloaded suppliers. This improves the system's overall performance in terms of response time and energy savings.

This protocol is applied at the requester which plays an important role in minimizing

the energy consumption in the system. However, when loads on potential sources are compared, we can detect a source that is overloaded. An overloaded source is one that served more requests and had more streaming activity than its neighbors. In order to further distribute the load among suppliers, it would be preferable not to select the overloaded source. Therefore, each source can participate in the energy management control, relieve itself from additional load and save its energy. This motivated us to improve 1L-EB to incorporate the supplier's role in energy management control during supplying partner selection. The result is our two-level supplying partner protocol.

4.2 2L-EB Supplying Partner Protocol

Our two-level energy based supplying partner protocol (2L-EB) is applied on both the requester and the supplier sides. Both levels are complementary and play important roles in the energy management control mechanism. Figure 4.2 illustrates the roles of both the supplier and the requester. Level 1 represents the requester's side, while Level 2 describes the supplier's role.

Similar to 1L-EB, the requester in 2L-EB prioritizes the potential sources in an ascending order based on their loads and selects the sources that have the least load. The supplier frequently compares its load with its neighbors' loads to detect whether it is overloaded in comparison. The supplier's role is described more in detail in Algorithm 2.

As mentioned earlier, peers in the VE exchange beacon messages at the begining of each phase. These beacon messages include relevant information related to the peer's coordinates, its speed, the 3D data it holds, and its load during the preceding phase. This information is used to keep track of the neighbors' positions and their 3D data availability. The load information contained in the beacon message is used to verify the state of the peer. Each peer is responsible for comparing its load with the loads of its neighbors. After processing this information, the peer is able to detect whether or not it is overloaded.

Let us consider a mobile peer P, and its load as Load_P. After receiving beacon messages, P computes its list of neighbors L_{Neigh}(P) and determines the neighbor that has the lowest load (Load_{Min}). P then determines its state, i.e., overloaded or not, by performing the difference Dif_{Load} between its load and (Load_{Min}) as shown in Equation 4.5, and comparing Dif_{Load} to a predefined value δ[1] using Equation 4.6 as illustrated below.

[1]The value of δ will be determined empirically.

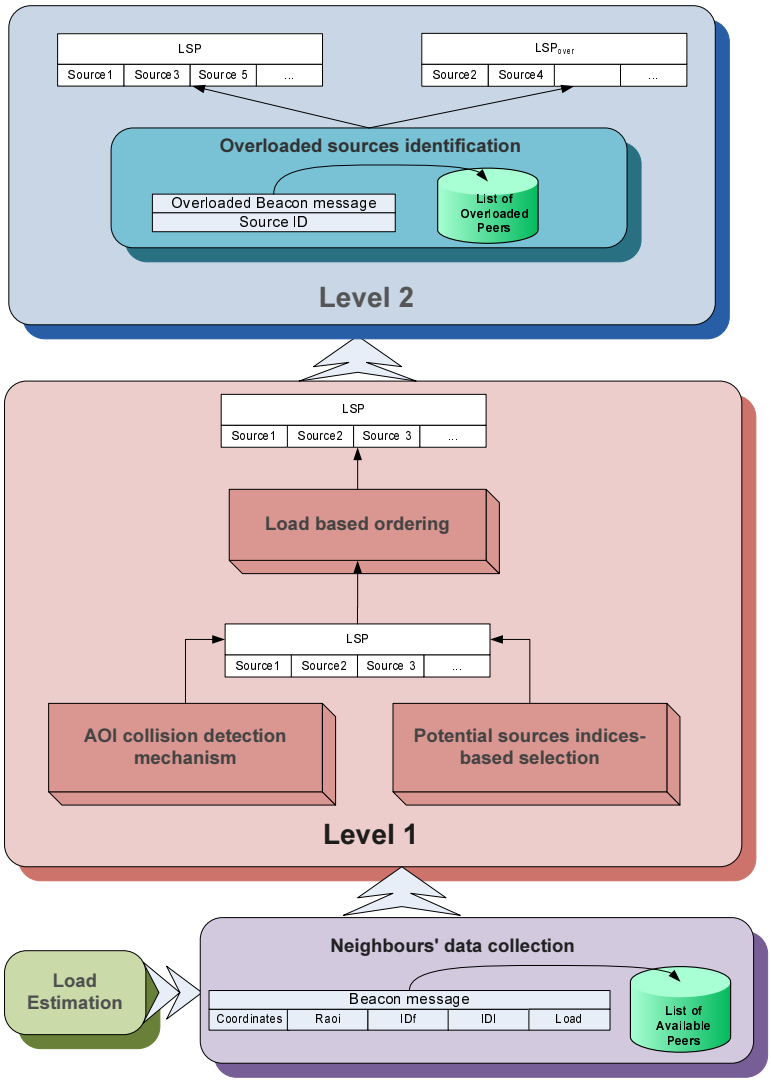

Figure 4.2: Supplying Partner Selection

Algorithm 2 Source Level Control

Require: L_{Neigh}(P), $Load_P$, L_{Supp}(P), $L_{Overload}$(P)
1: $LOAD_{Min} \leftarrow$ Min.Load(L_{Neigh})
2: **if** $Load_P$ - $LOAD_{Min} > \delta$ **then**
3: Peer P is overloaded
4: Peer P sends an Overloaded beacon message
5: **end if**
6: Wait for a timer T
7: **for** each message in $Buffer_{Overload}$ **do**
8: Remove Source S from L_{Supp}(P)
9: Add S to Overloaded List $L_{Overload}$
10: **end for**
11: Clear $Buffer_{Overload}$
12: Distribute the streaming among sources in L_{Supp}(P)
13: **if** data is still needed **then**
14: **while** Data is still needed **do**
15: check data availability $L_{Overload}$
16: **if** data available **then**
17: send Data Request
18: **else**
19: check next element in $L_{Overload}$(P) or ask Server if no more elements in $L_{Overload}$(P)
20: **end if**
21: **end while**
22: **end if**

$$Dif_{Load} = Load_P - Load_{Min} \tag{4.5}$$

$$State(P) = \begin{cases} Overloaded & if\ Dif_{Load} >= \delta \\ Not\ overloaded & if\ Dif_{Load} < \delta \end{cases} \tag{4.6}$$

When P is declared overloaded, P broadcasts an *overloaded beacon* message to its neighbors asking them to not select it as a supplier, and to remove it from the list of potential sources $L_{Supp}(P)$. When the neighbors receive the overloaded beacon message, they put P in a special buffer that we refer to as $Buffer_{Overload}$, for later processing. Once all neighbors are verified, each peer has to wait for a certain time T, in case some *overloaded beacon* messages are received. When T expires, the $Buffer_{Overload}$ is checked and all sources contained in it are removed from $L_{Supp}(P)$ and added automatically to a special list that we refer to as $L_{Overload}(P)$. Nodes within $L_{Overload}$ are requested if none of the potential sources in $L_{Supp}(P)$ holds the required information.

Once the final list of potential sources for a given requester is determined, requests are distributed among the suppliers in $L_{Supp}(P)$ starting with those that have low loads. This improves the system performance, given that requests are distributed among available sources, and are therefore served quickly with low latency. When all potential sources

Table 4.1: Simulation Parameters

Parameter	Value
World dimension	400m * 400m
Cell size	100m * 100m
AOI radius	10.5m (max)
Node speed	1-20 m/s
Buffer size	50 packets
MM packet size	10000 Bytes
Simulation time	2000 sec
Radio Propagation Model	Two Ray Ground
Mac Protocol	IEEE802.11
Antenna Type	Omni Antena
Wireless bandwith link	11 Mbps
Mobile node RXThresh	70m
RM and Routers RXThresh	250m

in $L_{Supp}(P)$ are verified and they do not hold the required 3D data, the requester selects sources from those in $L_{Overload}(P)$ which may hold the required data. If the source in $L_{Overload}(P)$ do not hold the required 3D data, the request will be sent to the server.

By applying this procedure, i.e., removing the overloaded sources from L_{Supp}, our proposed protocol 2L-EB balances the load on potential sources while relieving the overloaded suppliers. Both sides, the requester and the supplier, participate in the energy management control and play complementary roles in minimizing the energy consumption, extending the energy on mobile devices, and improving the system performance.

4.3 Experimental Results

In this section, we present the performance evaluation of our proposed 1L-EB and 2L-EB energy management control protocols. An extensive set of simulation experiments using the NS2 [76] network simulator were carried out to obtain energy consumption and response time related evaluations. The simulation parameters are presented in Table 4.1. Each simulation run was repeated several times with a confidence interval of 95%.

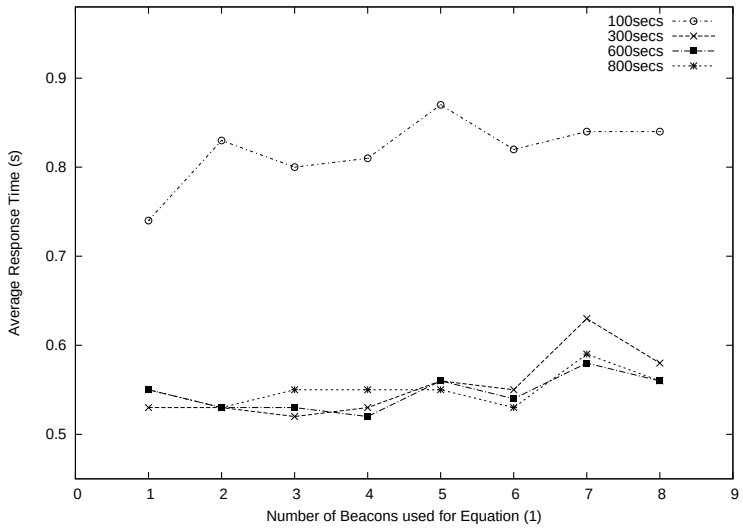

Figure 4.3: Variation of the number of phases X in Equation (4.1) ($\omega = 1$,$\mu = 0$).

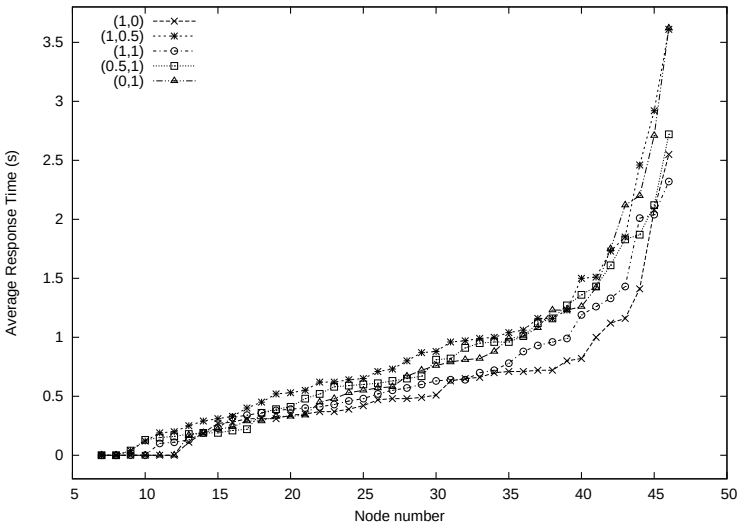

Figure 4.4: Average response time for different values of (ω,μ) in Equation (4.4).

4.3.1 Performance Metrics

In order to evaluate the performance of our protocol, we used a set of metrics:

1. *Fill Ratio*: this metric represents the visual quality of the scene. It is defined to be the ratio of received objects over the total number of objects within the user's AOI. The higher this ratio is, the higher the number of displayed objects is and the better the quality of the scene is

2. *Response Time (end-to-end delay)*: this metric represents the average amount of time measured from the instant a data packet containing the 3D object is originated until the packet is successfully received at the final destination node. This metric can also illustrate the data acquisition time that indicates how fast the 3D object can be displayed on the mobile device.

3. *Server Overhead*: this metric illustrates the ratio of 3D data packets that have been sent by the server in case there are no available supplying partners with respect to the number of requests.

4. *Load distribution*: This metric represents the amount of streaming activity performed by each node. It can also be expressed in terms of the number of requests successfully served. A high number of served requests reveals that the source is overloaded and that its energy is dissipated.

5. *Energy consumption*: This metric shows the energy that has been consumed due to computation, i.e., rendering and streaming activities.

The results are presented as follows. We first show the effects of the Equation (4.4) load parameters when using the proposed 1L-EB and the effects of δ parameter when adding Two-Level controls. Then a comparison of 1L-EB and 2L-EB is presented, highlighting the advantages of using the Two-Level energy management control protocol. Finally, the proposed 2L-EB protocol is compared to our previously described supplying partner algorithms LB3D and RB3D.

4.3.2 Evaluation of load parameters

The number of phases X used in Equation (4.1) to determine the realization ratio has several effects on the protocol performance. To study the X parameter's influence, we

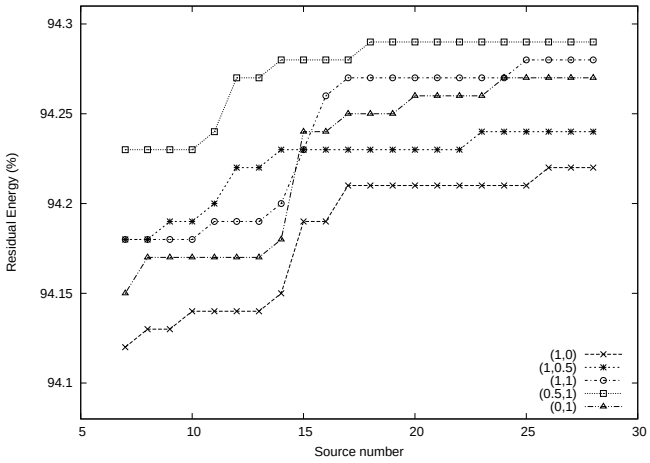

Figure 4.5: Energy dissipation for different values of (ω,μ) in Equation(3.2).

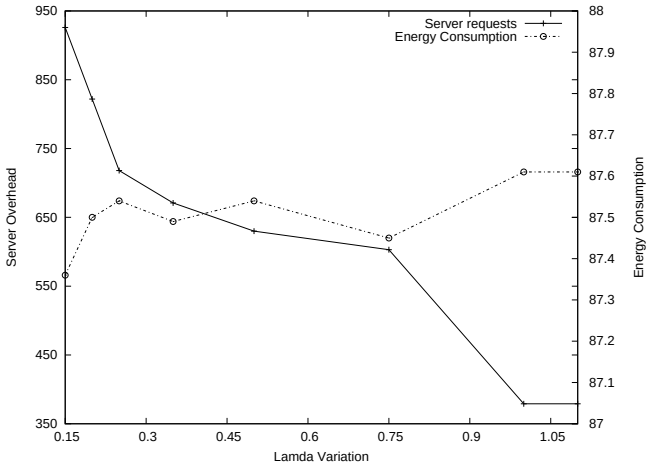

Figure 4.6: Server Overhead and residual energy when δ varies.

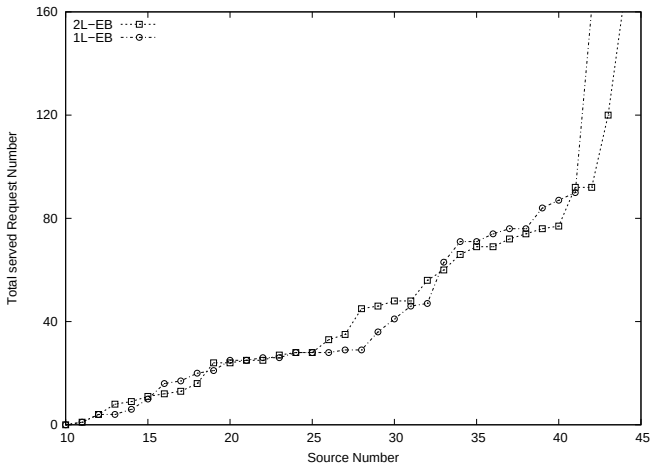

Figure 4.7: Average served requests per source node.

used 1L-EB protocol and we set $\omega = 1$ and $\mu = 0$ in Equation (4.4). Figure 4.3 shows how the average response time is affected using different values of X when the simulations progresses. As we cn see in Figure 4.3, the simulation lasting 100 sec can not give an idea about the best value of X since the average response time is very high and exceeding 0.85 sec for X= 5 phases. Simulations lasting 300, 600 and 800 sec have comparable values in terms of response time that varies for all of the simulations between [0.52, 0.56], except for X = 7, where the response time exceeds 0.62 sec. From Figure 4.3, we can observe that the lowest response time, for the three simulations (300, 600 and 800 sec), is given for X = 4. Therefore, we identify X empirically and we set it to 4.

The initialization of (ω, μ) parameters determines the weight of the realization ratio and residual energy in Equation (4.4) for the node load estimation, and therefore for the distribution of requests. Using the 1L-EB protocol and $X = 4$, we studied the effects of different values of both parameters on the resulting response time and energy consumption of the system. Figure 4.4 shows the obtained average response time per source node with $(\omega, \mu) = (1, 0)$, $(1, 0.5)$, $(1, 1)$, $(0.5, 1)$, and $(0, 1)$. As illustrated in the figure 4.4, the results of response time for all of the combinations are very comparable. For the first 20 nodes, the response time is less than 0.5s for the different values of ω and μ. The response time exceeds then 0.5s for nodes in [20,35] and goes above 1s for nodes in [35, 45]. As illustrated in the figure 4.4, the simulation $(1, 0)$, which only takes into account the realization ratio, has a lower response time than the remaining simulations. This is due to the fact that in the simulation $(1, 0)$, the load is expressed only in terms of the realization ratio i.e., served requests, where nodes with high number of served requests are alleviated. The protocol thus tries to avoid the selection of sources that served a large number of requests by favoring the response time regardless of the source's residual energy. From the figure 4.4, we can also observe that the simulation $(1, 1)$ which gives the same weight for both the realization ratio and the residual energy, gives the second best response time. The figure confirms therefore that combining the residual energy with the realization ratio influences the response time.

Figure 4.5 illustrates the sorted energy savings obtained when varying (ω, μ). The residual energy varies between [94.1, 94.3]. As shown in the figure 4.5, the simulation $(0.5, 1)$ obtained the best results with the highest residual energy outperforming the simulation $(1, 0)$ and improving it by 87%. This is due to the fact that the simulation $(0.5, 1)$ relieves the nodes that have low residual energy and an average RR%. Simulation $(0.5, 1)$ presents also better results than simulation $(0, 1)$ that only looks for relieving low residual energy nodes regardless of its RR%. Simulation $(1, 0)$ which had the best average

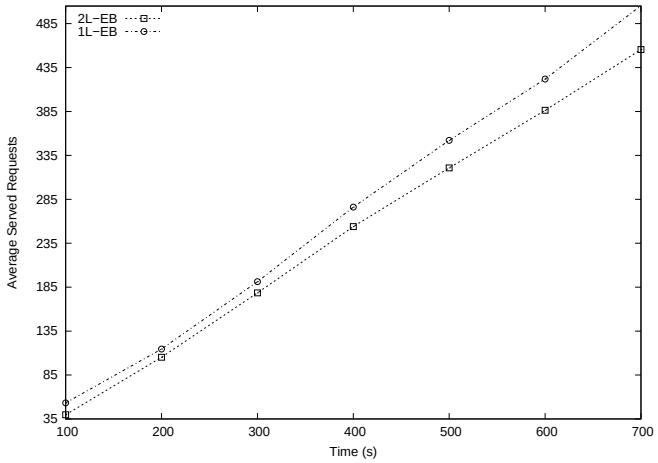

Figure 4.8: Average served requests.

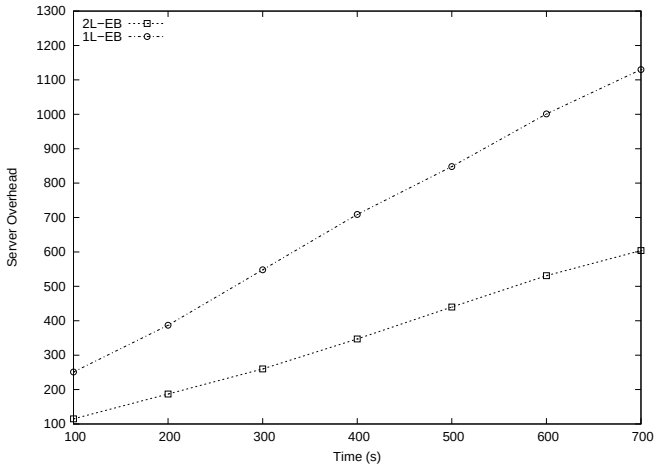

Figure 4.9: Server Overhead.

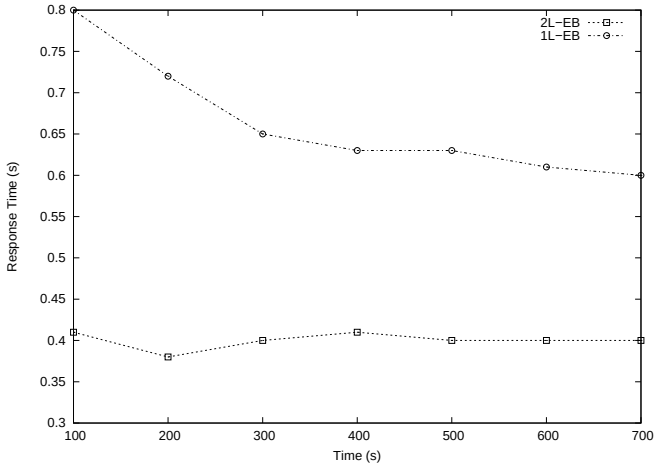

Figure 4.10: Average response time.

response time in Figure 4.4, now shows poor energy savings. This result confirms our expectations, since 1L-EB in this case relieves the peers that have a high RR%, regardless of the energy of the sources. Simulation $(1, 1)$ continues to be a good option for saving energy, since 1L-EB relieves the nodes with a high RR% and low residual energy.

We believe that the ω and μ parameter values could be adapted depending on the purpose of the application. In this work we consider both, response time and energy savings equally important so we set (ω,μ) = (1,1) for the next set of experiments. The performance of 2L-EB protocol depends on the initialization of the δ parameter. We recall that δ is the predefined value that determines if a given source is overloaded in comparison to its neighbors. Each source in 2L-EB compares its load with the loads of its neighbors. If the difference of loads exceeds δ, then the source is considered to be overloaded. The source has to send an overloaded beacon to inform the neighbors about its state and to ask them to remove it from the list of potential sources L_{Supp}.

Figure 4.6 presents the average number of requests served by the server (following the left scale), as well as the residual energy of the sources (following the right scale). As illustrated in Figure 4.6, when δ is small less than 0.2, the server overhead is considered high. As we increase the value of δ, we observe that the server overhead decreases. This shows that the server is asked fewer times to stream 3D data. When δ is small, the source is frequently detected as being overloaded compared to its neighbors. This implies that our proposed protocol tries to relieve the overloaded sources and requests, in this case, are either distributed among other sources or served by the server. This increases the server overhead. However, when we increase the values of δ, sources have more chances to stream data, which minimizes the server overhead.

As for energy savings, we can observe in Figure 4.6 that when δ is small, the residual energy is low, probably due to the number of *overloaded beacon* messages that many sources have to send when identified overloaded. The 2L-EB protocol achieves better energy dissipation when the value of δ increased having best performance when $\delta = 1$. As we increase δ, the 2L-EB protocol balances the load and therefore the energy in the system, minimizing the energy consumption.

Based on the above observations, we chose to set $X = 4$ and $\delta = 1$ for the remaining experiments, since they gave better results in terms of energy savings and response time.

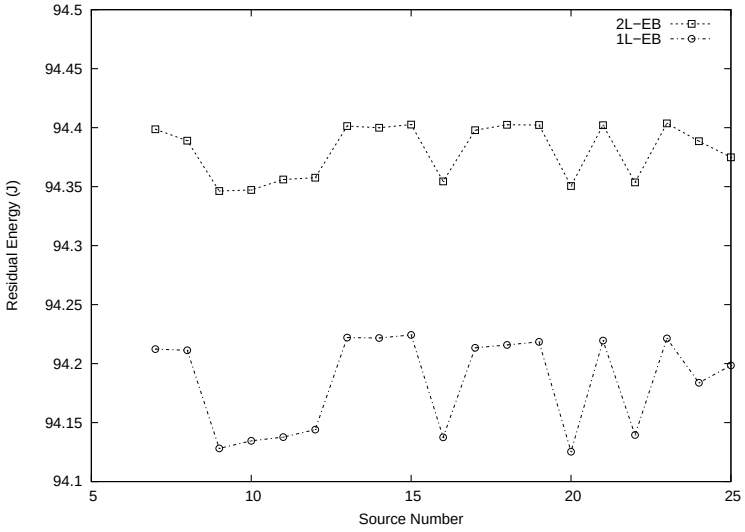

Figure 4.11: Residual energy per source in 1L-EB and 2L-EB.

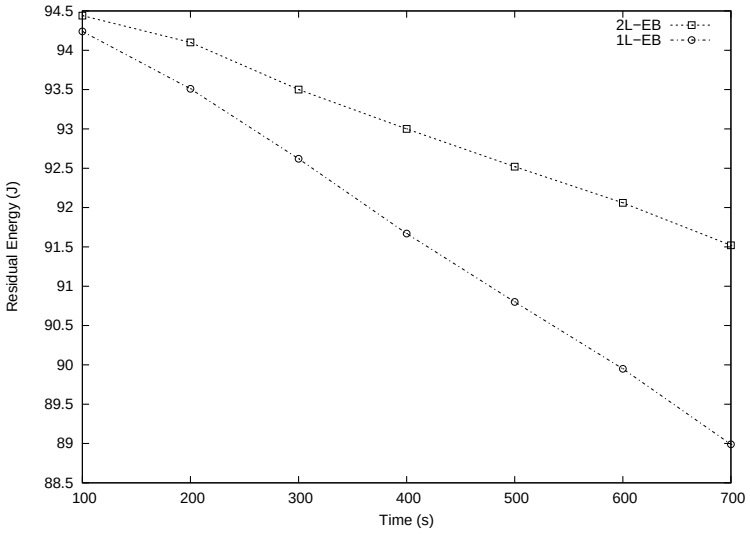

Figure 4.12: Residual energys.

4.3.3 1L-EB vs 2L-EB: A Comparative Study

In section we will present a comparative study between the two variant protocols 1L-EB and 2L-EB. The average number of served requests per supplier when applying 1L-EB and 2L-EB is illustrated in Figure 4.7. As shown, we can see that both approaches get almost an equivalent distribution of requests among suppliers, which means that both approaches succeed in distributing the load among sources. When comparing the average number of served requests when the simulation time progresses, as presented in Figure 4.8, we can see that 2L-EB has a lower number of served requests as compared to 1L-EB. When the simulation time is less than 300s, we notice from the figure 4.8 that the average number of served requests for 1L-EB and 2L-EB is very comparable. However, as we increase the simulation time, 2L-EB stars to have 20% less of served requests compared to 1L-EB. This is due to the two-level control, where both the requester and the supplier work in conjunction to relieve the overloaded sources.

In the next set of experiments, we investigated the server overhead. This is considered a good indicator of the system performance, as a smaller average number of requests served by the server implies an increase in the peers interactions. Figure 4.9 shows the average number of requests served by the server for both protocols 1L-EB and 2L-EB. As illustrated in the figure 4.9, 2L-EB has 56% less requests served by the server as compared to 1L-EB.

Figure 4.10 illustrates the change in system response time through the simulation when executing the 1L-EB and 2L-EB protocols. The response time for 1L-EB varies between [0.6s, 0.8s], while the response time for 2L-EB is in the interval [0.3s, 0.5s]. As shown in Figure 4.10, when the simulation time is 100s, 2L-EB gives a better response time than 1L-EB with 49% of improvement. As we increase the simulation time, the response time for 2L-EB remains better with 40% of improvement as compared to 1L-EB. These results confirm our initial thoughts where avoiding overloaded sources with the 2L-EB protocol results in fewer server requests and obtains better results on response time when compared with the 1L-EB protocol.

Results related to the energy savings of the 1L-EB and 2L-EB protocols are shown in Figures 4.11 and 4.12. Figure 4.11 shows the residual energy of source nodes whose initial energy was set to 95%. For the remainder source nodes whose initial energy was set to 85%, a similar behavior was observed. As illustrated in Figure 4.11, we notice that 2L-EB protocol clearly saved more energy than 1L-EB protocol with 40% of improvement. This is again due to the fact that in 2L-EB both the supplier and

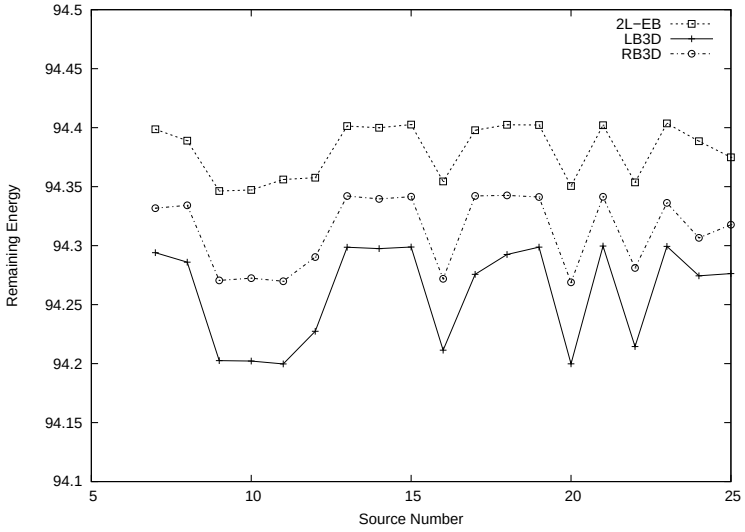

Figure 4.13: Residual energy per source.

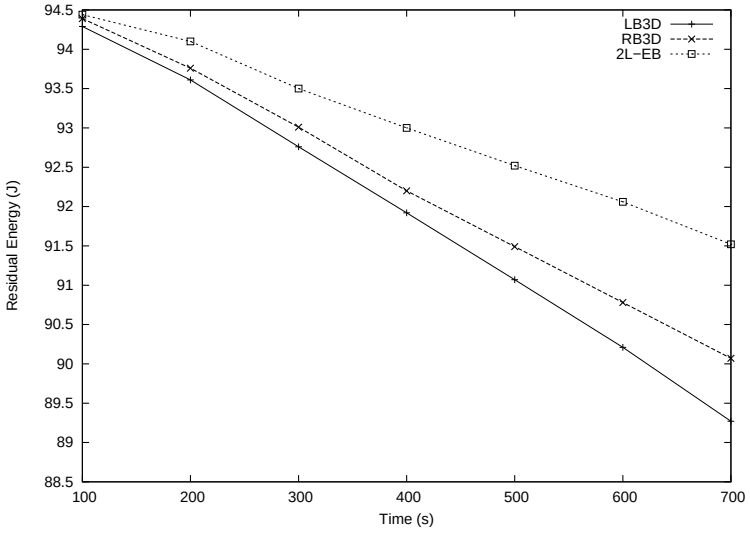

Figure 4.14: System residual energy when time varies.

the requester participate in the energy management control, whereas in 1L-EB only the requester manages the enegy. In Figure 4.12, we expressed the residual energy of the system as we vary the simulation time. We can notice that as we increase the simulation time, the difference between the residual energy when applying 1L-EB and 2L-EB becomes more important. The whole system's residual energy stayed higher during the simulation using 2L-EB, compared with the 1L-EB protocol, as it is shown on Figure 4.12 with 60% of improvement.

4.3.4 2L-EB vs RB3D vs LBED: A Comparative Study

In our earlier results as illustrated in the section 4.3.3, 2L-EB gave better results in terms of energy savings, response time and average number of served request when compared with 1L-EB. Therefore, in this section, we settle with comparing 2L-EB with LB3D and RB3D.

Figure 4.13 illustrates the results we have obtained when comparing these protocols. As we can see the residual energy of the suppliers when using the three protocols, LB3D, RB3D, and 2L-EB. As illustrated, 2L-EB performs better than RB3D and LB3D with 45% and 60% of improvement simultaneously. The results clearly show that the best protocol for energy savings is our proposed 2L-EB approach. This is due to the fact that only 2L-EB takes into consideration the energy factor when choosing the supplying partners. RB3D and LB3D consider other factors, such as the source's location or the source's resources, which do not try to minimize the energy consumption. In Figure 4.14, we show the variation of the residual energy when the simulation time progresses. As illustrated, we observe that as the simulation time increases, 2L-EB performs better than LB3D and RB3D with 40% and 70% simultaneously.

Another metric that indirectly shows how the P2P system performs, is the server overhead. We analyzed this metric and plotted the results in Figure 4.15. As we can see, when the simulation time progresses, our proposed 2L-EB approach resulted in 50% less requests served by the server when compared to LB3D and 20% less requests served by the server when compared to RB3D. This is mainly due to two facts. First, RB3D tries to relieve sources that have a high number of requests resulting in requests to the server. Second, LB3D selects the suppliers based on their locations; however the selected sources may not have the required information, which also results in requests to the server.

In Figure 4.16, we present the average response time of sources when running the three protocols LB3D, RB3D, and 2L-EB. As the simulation time progresses, we noticed that

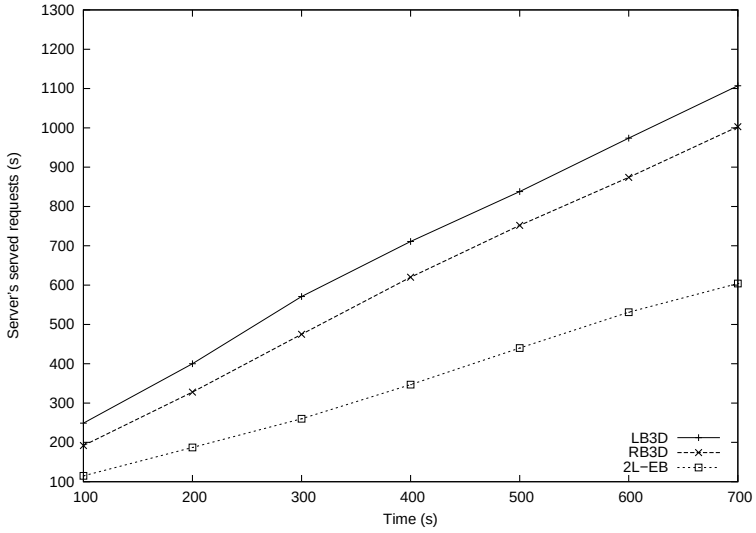

Figure 4.15: Server Overhead.

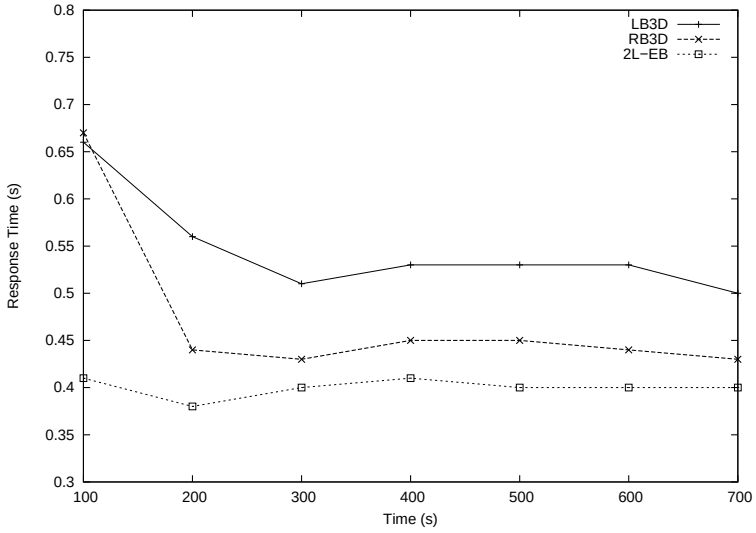

Figure 4.16: Average response time (s).

RB3D obtained better response times than LB3D with 20% of improvement, however we can clearly see that our approach 2L-EB returns the best results when compared to RB3D with 65% of improvement. This is due to the fact that 2L-EB manages the load on sources to relieve the overloaded source, which leads to low response time.

The set of obtained results presented in this section indicated clearly how our 2L-EB energy based supplying partner protocol positively affected both the energy savings and the system's response time.

4.4 Conclusion

Energy is an important factor to take into account when dealing with 3D streaming over thin mobile devices. In this chapter, we propose 1L-EB and 2L-EB, our suite of energy management control protocols for supplying partner selection in the P2P 3D streaming based class of applications. In 1L-EB, it is the responsibility of the requester to manage the energy consumption of sources. To do so, we propose a new source load estimator that takes into account two factors: the source's residual energy and its number of served requests. The requester uses each source's load information to prioritize the suppliers in an ascending order and to distribute the requests among them starting with the sources that have the lowest load. In 2L-EB, both the requester and the supplier play important and complementary roles in the energy management control. The supplier, in 2L-EB, is responsible for estimating its own load and comparing it with its neighbors' loads. If the supplier is considered overloaded in comparison to its neighbors, it is then responsible for informing them by sending an overloaded beacon. Performance evaluation of both protocols shows that they improved the system's overall performance in terms of response time and energy savings. Moreover, when compared to previously proposed mobile P2P 3D streaming protocols (LB3D and RB3D), experiments showed that 2L-EB performs better in terms of energy savings and response time. However, all of the proposed protocols considered only sources that are within one hop of the requester, which leads in some cases to turning toward the server if the requested peers do not hold the required data. This does not favor the P2P interactions of the system and induces delay, request contention, and server bottlenecks. These later issues could be avoided by examining peers that are at multiple hops from the requester and that may hold the requested data. In fact, other peers who have been in the same area as the requester, may still hold the required 3D data, and could therefore be selected as suppliers in order to alleviate the load on the server. In the following chapter, we shall introduce our

multihop supplying partner selection protocol for 3D streaming based systems over thin mobile devices.

Chapter 5

MULTIPLY: A Multihop Discovery Supplying Partner Protocol

In this chapter, we propose a **multi**hop sup**ply**ing partner selection technique coupled with a content delivery technique (MULTIPLY) for P2P 3D streaming systems over thin mobile devices. Our proposed protocol allows the selection of multiple supplying partners in order to speed up the downloading time. Our previous proposed protocols, i.e., LB3D, RB3D and 2L-EB only analyzed peers that are one-hop from the requester. This led some requests to be served by the region master (RM) given that peers that are one hop from the requester did not hold the required data. However, other peers, who have been in the same area before may still hold the relevant 3D data and may thus not be selected given that they are not within the radio range of the requester. In our MULTIPLY protocol, we propose to consider multihop supplying partners. By all means, the analyzed potential peers have to be in the same area as the requester since they have a high probability to hold the required information in their buffer. Each given requester can have multiple sources. In fact, we believe that the multi-source concept improves the streaming performance of the system since delay is greatly reduced, and data loss is minimized.

Given that our protocol is based on a multihop discovery of supplying partners, each peer maintains a path maintenance table used to keep track of the forwarded and served requests. As illustrated in Table 5.1, the path maintenance table is composed of four fields:

- *Sequence number* field: represents the sequence number of the request packet. The data packet corresponding to a given request must have the same sequence number

as the request packet. This assumption was applied in order to detect whether a given request has been served or not.

- *Origin* field: represents the address of the original requester. This way, we make sure that requests are not mistaken, and are not processed twice. Moreover, we ensure that 3D data corresponding to a given request are forwarded to the correct requester.

- *Predecessor* field: indicates the address of the node that forwarded the request to the receiving node. Obviously, this field will have the address of the original node, if the receiving peer is at one hop from the requester. During the 3D data content delivery, the 3D packet should take the same path taken by the request packet. To this end, each node forwards the 3D packet to the node contained in the *Predecessor* field: of the corresponding request. Some exceptions exist and will be discussed in Section 5.1.4.

- *Served* field: indicates whether the 3D data packet, corresponding to the request, has been received and processed or not.

Table 5.1: Path Maintenance Table

Field	Description
Sequence Number	The sequence number of the request packet
Predecessor	The address of the predecessor peer
Origin	The origin of the request
Served	A Flag to indicate whether the 3D packet has been received

In the following, we shall describe the behavior of our proposed multihop discovery supplying partner protocol.

5.1 Multihop Discovery Supplying Partner Protocol

In this section, we shall introduce our muLtIhop Supplying partnEr technique for 3D streamiNg systems running over mobile devices, that we refer to as MULTIPLY. Our proposed protocol is composed of four phases that are repeated each T_{phase}: (1)the initial

phase, (2) the supplying partner search and selection phase, (3) the acknowledgement phase, and (4) the 3D data delivery phase

5.1.1 The initial phase

During the initial phase, each mobile peer broadcasts a beacon message to the mobile peers within its radio range. However, only the neighboring peers, that are within its region, process the received beacon messages. All of the remaining peers discard the update message. Beacon messages are sent periodically every T_{phase}, in order to minimize the number of messages sent through the network, and therefore minimize the bandwidth consumption, and the communication overhead.

Based on the information contained within the beacon messages received, each mobile peer keeps track of its neighbors, i.e., those located within its region, and creates its own list of neighbors that we refer to as (L_{Neigh}). Note that the neighbors of a given requester are the peers located at one hop from the requester and that are within its radio range. Each beacon message is also routed to the area's relative RM in order to allow it to monitor the positions of mobile peers within its region. Once L_{Neigh} is created, each mobile peer creates its list of potential sources (that we refer to as L_{Source}) by identifying the sources that may have the relevant 3D data needed.

5.1.2 The Supplying partner discovery and selection phase

During this phase, every peer that needs data initiates the supplying partner search phase. Algorithm 3 illustrates the steps taken by the requester.

As stated, when a given peer P requires data, it first checks its list of neighbors L_{Neigh}, i.e., the peers that are located at one hop from the requester. If more than one potential source exists, then the streaming is distributed among them in order to speed up the downloading time and to distribute the load among the sources. In the case where a sole supplier exists, only a part of the required 3D data is requested. The remaining 3D data is obtained by applying the multihop protocol. This has been applied in order not to overuse the resources of the supplier.

In the multihop protocol, the requester broadcasts its data request to a specified list whose members are part of the requester's neighbors. Before broadcasting its request, the requester computes a set of direction peers that we refer to as $L_{Direction}$. The computed set contains the peers that are located in the movement direction of the requester and that are within its radio range. We believe that these peers have a higher probability

Algorithm 3 Requester Side

Require: $L_{Direction}$(P), L_{Neigh}(P), L_{Source}(P), halfPlane(P,P.direction)
1: P requires Data(seqNum)
2: L_{Source}(P) is checked
3: **if** 3D data is available in L_{Source}(P) **then**
4: Use One-hop Supplying selection technique
5: **else**
6: **for all** Neighbor IN L_{Neigh}(P) **do**
7: **if** Neighbor IN halfPlane(P,P.direction) **then**
8: $L_{Direction}$(P).add(Neighbor)
9: **end if**
10: **end for**
11: Send DataReq(seqNum, $L_{Direction}$)
12: DataReq.hop = DataReq.hop + 1
13: Start a timer for σ (s)
14: When timer times out
15: **if** !(DataPack is received) **then**
16: Send DataReq(seqNum, RM)
17: **end if**
18: **end if**

to hold the required information since they are in the field of view of the requester, and have therefore the same view of the VE. Hence, all of the peers that are in the half plane perpendicular to the direction of the requester are selected and added to $L_{Direction}$ as illustrated in Figure 5.1.

Once the $L_{Direction}$ is computed, the requester broadcasts its request (containing the sequence number of the 3D data needed) to the members of $L_{Direction}$ and starts a timer σ. By all means, it is already known in advance that the members of $L_{Direction}$ do not have the relevant 3D data in their buffers, given that the requester is applying the multihop protocol.

When a given peer receives a broadcasted data request, it first examines its list of sources L_{Source} to verify whether one of its potential suppliers holds the relevant 3D data. When the 3D data is found at a given supplier, the peer forwards (unicast) the data request to the relevant supplier, and updates its path maintenance table. However, when the 3D data is not found at the members of L_{Source}, the peer computes a set of nearby neighbors that we refer to as $L_{closeNeighbor}$. This latter set contains all of the neighbors that are within a certain distance β from the peer. We believe that neighbors that are nearby the peer, in terms of distance, have a higher probability to hold the required information as compared to farther ones. Moreover, requesting nearby peers reduces the transmission delay. Hence, once $L_{closeNeighbor}$ is computed, the data request is forwarded to the members of $L_{closeNeighbor}$, the maintenance path table of the peer is

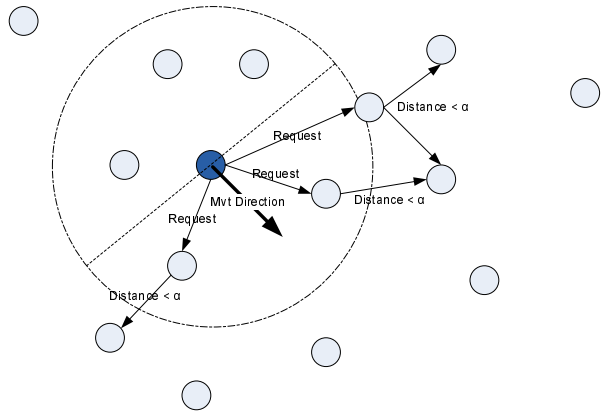

Figure 5.1: Source Selection Process

updated as well as the data request's hop number.

Our proposed protocol has the ability to avoid loops during the transmission of the data request or the delivery of the 3D data. Such loops occur when the requester receives its own data request via its neighbors, or when the source receives its own 3D data packet via intermediate nodes. In order to address this issue, our protocol ensures to keep the path taken by each data request by maintaining a list of predecessors. When a peer receives a given data request, it examines the route taken by the request packet. The data request is discarded if the peer is the origin of the request, or if it has already received the request packet before. Otherwise, the peer verifies whether it holds the required data. If it is the case, then the 3D data packet is sent using the same path taken by the request. In the opposite case, i.e., the peer does not hold the required 3D data, it forwards the data request to the members of its $L_{closeNeighbor}$.

The source discovery process, illustrated in Algorithm 4, is repeated until the 3D data is localized at one supplier, or until the number of hops reaches a predefined value δ. We limit the number of hops in order not to flood the network, to reduce the unnecessary workload and to avoid affecting the network performance. In the following section, we shall present the acknowledgement phase.

5.1.3 Acknowledgement phase

During the acknowledgement phase, relay/intermediate nodes play an important role in the success of the 3D data delivery process. In fact, when the requester sends a data request to its $L_{Direction}$ members, some paths may be unsuccessful and leads to a NACK message sent back to the requester; while others paths may have localized a source and

Algorithm 4 Relay Node Side

Require: $L_{closeNeighbor}$, pathTable, Buffer(P), L_{Source}(P), L_{Neigh}(P)

```
1: P received a DataReq
2: potSource ← Buffer(P).begin()
3: while ! ((potSource.IDf < DataReq.seqNum) && (DataReq.seqNum < potSource.IDl)) do
4:     potSource ← Buffer(P).next()
5: end while
6: if (potSource != Buffer(P).end()) then
7:     DataPack.SeqNum ← DataReq.seqNum ;
       DataPack.origin  ← DataReq.origin ;
       DataPack.dest    ← DataReq.sender
8:     if radio Signal strength (P,DataReq.dest) > λ then
9:         Send DataPack(seqNum, DataReq.dest)
10:    else
11:        Send DataPack(seqNum, Router)
12:    end if
13: else
14:    if DataReq.hop < δ then
15:        for all pSource IN L_Source(P) do
16:            if Data is available at pSource then
17:                Forward DataReq(seqNum,pSource)
18:                P.pathTable[index][0] ← DataReq.seqNum;
                   P.pathTable[index][1] ← DataReq.origin;
                   P.pathTable[index][2] ← DataReq.dest;
                   P.pathTable[index][3] ← 0;
                   index ++
19:            else
20:                for all Neighbor IN L_Neigh(P) do
21:                    dist← distance (P,Neighbor)
22:                    if dist < β then
23:                        L_closeNeighbor(P).add(Neighbor)
24:                    end if
25:                end for
26:                Forward DataReq(seqNum,L_closeNeighbor)
27:                DataReq.hop ++
28:                P.pathTable[index][0] ← DataReq.seqNum;
                   P.pathTable[index][1] ← DataReq.origin;
                   P.pathTable[index][2] ← DataReq.dest;
                   P.pathTable[index][3] ← 0;
                   index ++
29:            end if
30:        end for
31:    else
32:        Send NackPack(DataReq.seqNum, DataReq.origin)
33:    end if
34: end if
```

the 3D data may have been sent. In that case, if the NACK message reaches the requester first, the requester would take other actions to find the relevant 3D data, unaware that data is being sent back from other sources. To avoid this from happening, we keep the NACK message at the relay node for a period of ω (s), to give the chance to other sources to send the relevant 3D data. When the timer times out, the relay node verifies whether the corresponding 3D data has been received or acknowledged by the original receiver. This is accomplished by examining the *SERVED* field in the path maintenance table. A received packet corresponds to a value of 1 in the *SERVED* field. In that case, i.e., the 3D data packet has been received, the NACK message is discarded; whereas in the opposite case, the NACK message is forwarded to the predecessor. This process is repeated until reaching the original requester.

5.1.4 The 3D Data Delivery Phase

The 3D data content delivery phase is applied once a potential source is localized. When a given peer receives a data request and that the 3D data is available in its buffer, the peer creates a data packet and sends it back to the requester.

Our multihop delivery protocol involves several wireless links, which leads to a high probability of 3D data loss due to link breakages and interferences. The packet loss is more accentuated since the bandwidth is dynamic on each wireless link which increases the packet loss especially when transmitting large 3D packets. Therefore, due to the wireless nature of the application, wireless interference is a major challenge that needs to be taken into consideration, given that it greatly affects and decreases the application performance as well as the quality of the 3D media. To overcome this issue, as illustrated in Algorithm 4, our proposed protocol examines the signal strength between two nodes before sending the 3D data. We denote the signal strength as the received signal strength (RSS).

Evaluation of the Signal Strength

Mobile devices emit a high frequency energy that propagates uniformly in all directions forming areas with the same power density. These areas have a spherical shape with the mobile device as the center of the sphere. The power density on the surface of the sphere is inversely proportional to the surface area of the sphere. The relation between the received power and the distance is presented in Equation 5.2 [93].

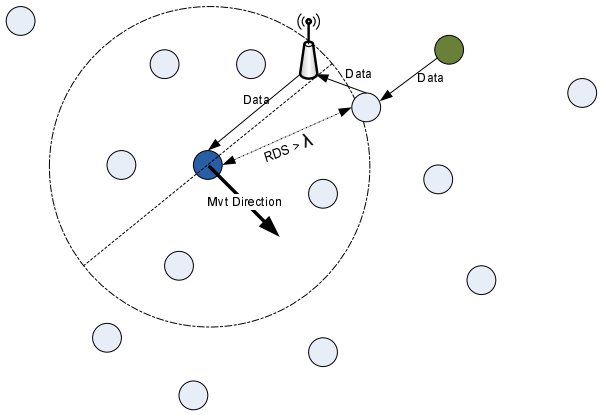

Figure 5.2: 3D data delivery technique

$$P_r = \frac{P_t * \alpha^2}{(4\pi)^2 * R^2} \quad [W] \tag{5.1}$$

Where P_r is the received power, P_t is transmitted power, α is the transmitted wave length and R is the distance that separates the transmitter and the receiver. As illustrated in the Equation 5.2, the received signal strength is inversely proportional to the distance that separates the transmitter and the receiver. For instance, when the RSS is high, the distance between the receiver and the transmitter is small.

We believe that the study of the signal strength is a very good technique to be aware of the state of the network and whether it is profitable to send the 3D data packets. This technique also helps reducing the packet loss. In fact, large data packets are the first to be dropped if the network experiences interferences. This definitely affects the system performance and generates long delays. Therefore, by studying the signal strength before sending the data packet, each source makes sure that the 3D packet is delivered to the requester or to the intermediate nodes. As a consequence, 3D data packet loss is avoided or greatly minimized and the system reliability is considerably improved.

Description of the Delivery Technique

In our proposed protocol, the signal strength between the source and its predecessor is measured. When the signal strength is greater than a threshold value λ, the data packet is sent to the predecessor. Otherwise, i.e., the signal strength is low, the packet is sent to the closest router that will be responsible for forwarding the data packet to the original requester as illustrated in Figure 5.2. The RSS analysis is a good technique to ensure

the reliable delivery of the 3D data packet and to decrease the data packet loss.

When a given peer receives a data packet, it looks for the corresponding request in its path maintenance table. As illustrated by Algorithm 5, once the peer finds the corresponding request in the path maintenance table, it examines the value of the *SERVED* field. If the *SERVED* is set to 1, then the data packet has been already received, and should thus be discarded. In the opposite case, i.e., *SERVED* is set to 0, the peer verifies whether it is an intermediate node or the original requester of the 3D data packet. In the case where the peer is a relay node, it sets *SERVED* to 1, and examines the RSS to its predecessor. The 3D data packet is forwarded to the predecessor if the RSS is greater than λ, while the 3D data packet is forwarded to the closest router, if the RSS is weak.

When the peer is the original requester, the 3D data is displayed and kept in its buffer. Intermediate nodes along the streaming path decide whether they want to keep a copy of the 3D packet in their buffer, depending on their available storage space.

Algorithm 5 Reception of a Data packet

```
Require: P.pathTable, searchReq, L_Direction
P receives DataPack(seqNum)
sendACK (DataPack.seqNum , DataPack.sender)
seqNumber ← DataPack.seqNum ;
searchReq ← find(P.pathTable, DataPack.seqNum , DataPack.origin)
if searchReq.Served == 0 then
    searchReq.Served ← 1
    if P == DataPack.origin then
        sendACK (DataPack.seqNum , L_Direction)
        Display (DataPack.data)
        Buffer(P).add(DataPack.data)
    else
        signal ← Received Signal strength Indicator (P,DataReq.dest)
        timer ← random(0,1) * signal
        while (timer != 0) && (!canceled) do
            Do nothing
        end while
        if (searchReq.Served == 1) || (canceled) then
            Discard(DataPack)
        else
            Forward DataPack(seqNumber,searchReq.sender)
        end if
    end if
else
    Discard(DataPack)
end if
```

Sending duplicate 3D data to the requester is not cost-effective given that the bandwidth is limited. Therefore, before forwarding the 3D data to the requester, the intermediate node should wait γ (s). This is applied in order to give the opportunity to

other intermediate nodes, that may be close to the requester and whose data has a low probability to be lost, to send the 3D data to the requester. The value of γ should be inversely proportional to the distance that separates the intermediate node and the original requester. The distance can be expressed using the signal strength as illustrated in Equation 5.2. Therefore, γ is proportional to the signal strength. Hence, once the timer of the closest intermediate node times out, it forwards the 3D data packet to the original requester. Upon receiving the relevant 3D data, the original requester broadcasts an ACK message that cancels all of the remaining intermediate nodes' timers scheduled to send the requested 3D data.

In the case where the requester's timer times out and the required 3D data has not been received, or in the case where a NACK message has been received, the requester should take other action to acquire the relevant 3D data. In our protocol, as mentioned in Algorithm 3, we propose that the requester turns toward the RM to acquire the relevant 3D data.

5.2 Performance Evaluation

In the present section we shall present the performance evaluation of our proposed multihop supplying partner protocol MULTIPLY. An extensive set of simulation experiments using NS2 simulator [76] with the settings presented in Table 5.2 has been carried out. Each simulation run has been repeated several times with a confidence interval of 95%. In the simulation, a number of nodes have been created to represent the mobile users in the VE. Each mobile node has a different speed and moves respecting the random way-point movement model. The initial position of each node is also randomly processed. After each phase T_{phase}, a new speed is generated for each mobile node. The speed of each node varies from 0.5 m/s to 15 m/s. For each speed 100 scenarios have been generated. In each scenario, each node has initial random location, destination and speed that is uniformly distributed between the minimum and the maximum speeds. Each node travels from its starting location to its destination at the speed that was specified. Once it gets to the required destination, a new destination and speed are assigned. This process is repeated until the end of the simulation time.

5.2.1 Performance Metrics

In order to evaluate the performance of our protocol, we used a set of metrics:

Table 5.2: Simulation Parameters

Parameter	Value
World dimension	400*400
Cell size	100*100
AOI radius	10.5 (max)
Node speed	0.5 - 15 m/s
Buffer size	50 packets
MM packet size	1000 Bytes
Simulation Time	2000 (s)
Radio Propagation Model	Two Ray Ground
Mac Protocol	IEEE 802.11
Antenna Type	Omni Antenna
Wireless bandwidth link	11Mbps
Mobile node RXThresh	70m
RM and Routers RXThresh	250m

- *Fill Ratio*: this metric represents the visual quality of the scene. It is defined to be the ratio of received objects over the total number of objects within the user's AOI. The higher this ratio is, the higher the number of displayed objects is and the better the quality of the scene is.

- *Routing load (routing overhead)*: this metric represents the total number of routing packets forwarded by the relay nodes.

- *The 3D data packet loss ratio*: this metric represents the rate of loss of 3D data packets. The packet loss can be caused by different factors such as the network signal degradation, or the signal-to-noise degradation and distance that separates the transmitter and the receiver, just to mention a few.

- *Latency (end-to-end delay)*: this metric represents the average amount of time measured from the instant a data packet containing the 3D object is originated until the packet is successfully received at the final destination node. This metric can also illustrate the data acquisition time that indicates how fast the 3D object can be displayed on the mobile device.

- *Server Overhead*: this metric illustrates the average number of 3D data packets that have been sent by the server in case there are no available supplying partners.

- *Server Request Ratio (SRR)*: This metric represents the ratio of served requests by the server. The lower the SRR is, the lower load is on the server and the better the system scalability is.

The results are presented as follows. We first analyze the effect of the distance β on the success rate of the protocol. We evaluate the success rate of the protocol by measuring the fill ratio. Second, we show the effects of varying the number of hops when using our proposed MULTIPLY protocol. Third, we show the performance of our proposed protocol MULTIPLY when varying the RSS threshold. Forth, our proposed MULTIPLY is compared to our previously presented supplying partner algorithms LB3D, RB3D and 2L-EB. Finally, we evaluate the performance of our MULTIPLY protocol when changing the mobility models.

5.2.2 Evaluation of the Protocol's Parameter

During the supplying partner discovery phase, a given requester starts by looking for the relevant 3D data in its $L_{Direction}$ members. If the data is not available, then the request is forwarded to the $L_{closeNeighbor}$ members of the relay nodes. The members of the $L_{closeNeighbor}$ are determined based on their distance from the relay node. A given peer is part of $L_{closeNeighbor}$ if its distance from the relay node does not exceed a threshold value β.

In this set of experiments, we study the effect of the distance β and try to determine the best value of β to use in the remaining set of experiments. To do so, we measured the fill ratio as well as the data acquisition time when varying the number of hops as well as the values of β. We varied the values of β from 5m-25m. During this set of experiments, the signal strength was not measured, and the 3D data packet was forwarded to the requester following the same path taken by the data request.

The fill ratio when varying the number of hops as well as the values of β is illustrated in Figure 5.3. As shown, we can see that for the different values of hops, β = 10m, has the highest fill ratio as compared to the remaining values of β. Figure 5.4 presents the data acquisition time as we vary the number of hops as well as the values of β. From the figure, β = 5m, has the highest data acquisition time, since the distance is short and not enough peers may be discovered in the area, which makes the data acquisition time long. As we increase the values of β, we can see that the results are very comparable. This suggests, and based on results obtained in Figure 5.3, that the best value for β is 10m.

In the proposed protocol, when a given requester does not find the relevant 3D data, it forwards its data request to its neighbors that also forward it to their neighbors in case the relevant 3D data is not localized. The data request is kept being forwarded up to

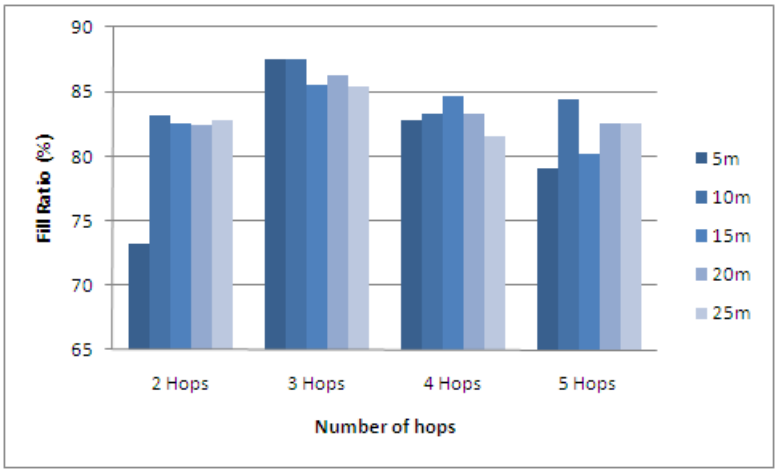

Figure 5.3: Fill Ratio when varying β parameter

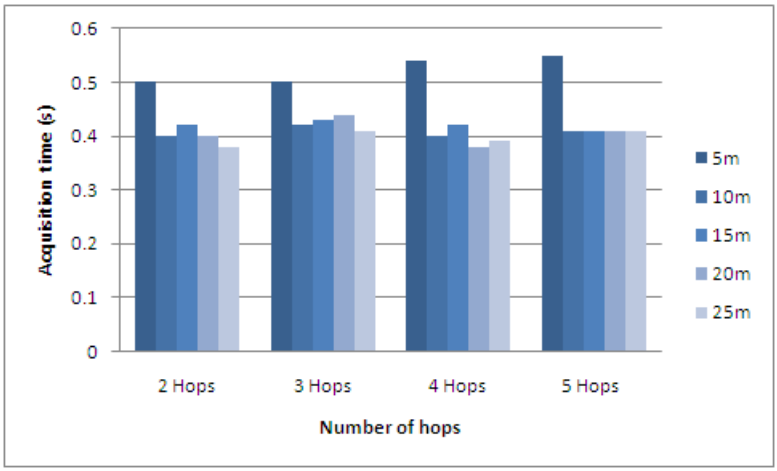

Figure 5.4: Acquisition time when varying β parameter

δ hops. To study the δ parameter's influence on the system's behaviour, we varied the number of hops. In our experiments, δ may be 2, 3, 4 or 5 hops. We believe that when δ exceeds 5 hops, there is a very small probability that the nodes hold the relevant 3D data. We fixed the value of β to 10m. Once the relevant 3D data is found, the 3D data packet is forwarded to the original receiver. In this set of experiments, the link quality is not tested. The 3D data packet takes therefore the same path as taken by the data request.

In the first set of experiments, we measured the routing overhead that is used to illustrate the number of routing packets forwarded by the intermediate nodes. To do so, we varied the number of hops δ from 2 hops to 5 hops with different simulation times that also vary from 100s to 700s. The results are illustrated in Figure 5.5. As we can see, when we increase the simulation time, the routing overhead for the different values of δ increases linearly. As a conclusion, when we increase the number of intermediate nodes (relay nodes) by increasing the number of hops, the routing overhead increases due to more connections between the sources and the requester. This makes the network overloaded and affects its efficiency.

In the second experiment, we measured the average number of data requests successfully served and its fill ratio as we vary the value of δ as well as the simulation time. The results are shown respectively in Figure 5.6 and Figure 5.7. As illustrated in Figure 5.6, as we vary the simulation time, we notice that the best results are obtained when $\delta = 3$ hops with 15% of improvement when compared to $\delta = 4$ and $\delta = 5$, and 35% of improvement when compared to $\delta = 2$. The same tendency and results are maintained when we increase the simulation time. When we measure the fill ratio as illustrated in Figure 5.7, we can clearly see that the best results are obtained when $\delta = 3$ hops.

The average acquisition time is also measured as we vary the value of δ. The results are illustrated in Figure 5.8. As we increase the simulation time, we clearly notice that the best results are obtained when $\delta = 3$ hops while the worst results are obtained when $\delta = 5$ hops. The average acquisition time when $\delta = 2$ hops is high compared to δ equal to 3 or 4 hops. This may be due to the fact that when the requester does not find the relevant data in sources up to 2 hops, it has to request the RM, which may take a long time to acquire the data, due to remote access.

The server overhead, and the SRR% as the simulation time varies are represented in Figures 5.9 and 5.10 simultaneously. We believe that the best factor to show whether the P2P system is performing well, is to analyze the number of 3D data packets sent by the server or in other terms the server overhead. The smaller the server overhead is,

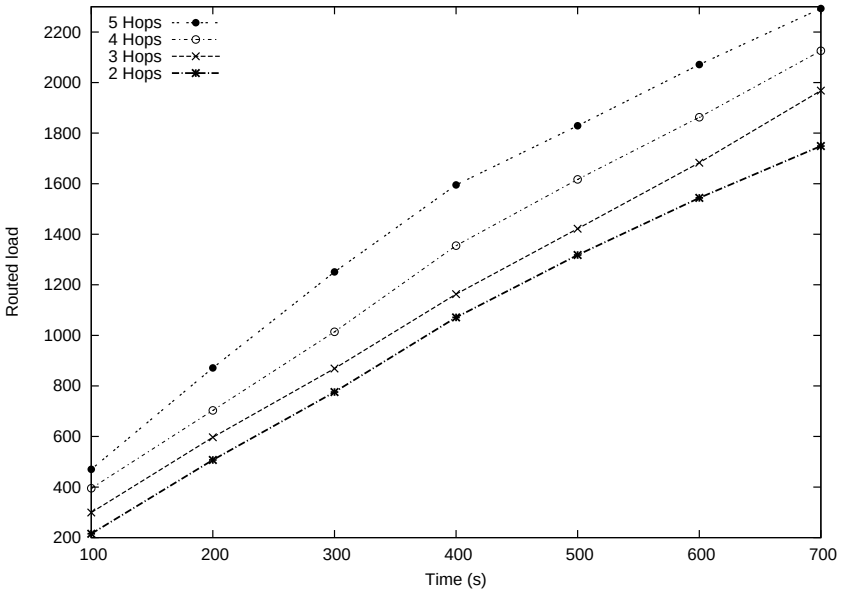

Figure 5.5: Routed load.

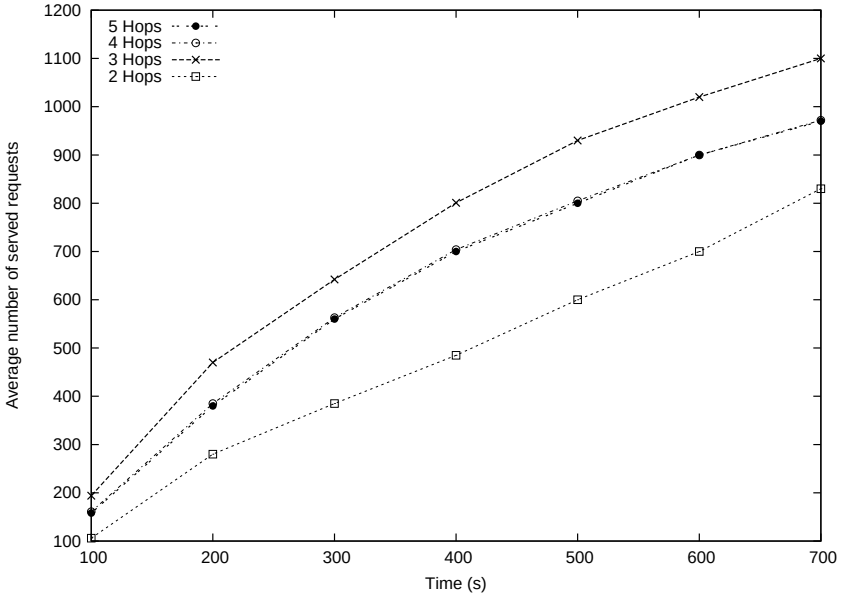

Figure 5.6: Average served requests.

the bigger the P2P interactions is, thus favoring the P2P systems response times. From the Figure 5.9, we can clearly see that the highest server overhead results is obtained for $\delta = 2$ hops. This confirms our theory related to the fact that when the number of hops is small, the requester would turn towards the server asking for relevant data. As for $\delta = 3$, 4, 5 hops, the results are very comparable stating a low server overhead and confirming that the P2P interaction is significant. These same results are confirmed in Figure 5.10 that illustrates the server request ratio SRR%. For $\delta = 2$ hops, the SRR% is 11% compared to SRR% = 5% for $\delta = 3$, 4, 5 hops. These results indicate that the P2P interaction is favored when the number of hops is high.

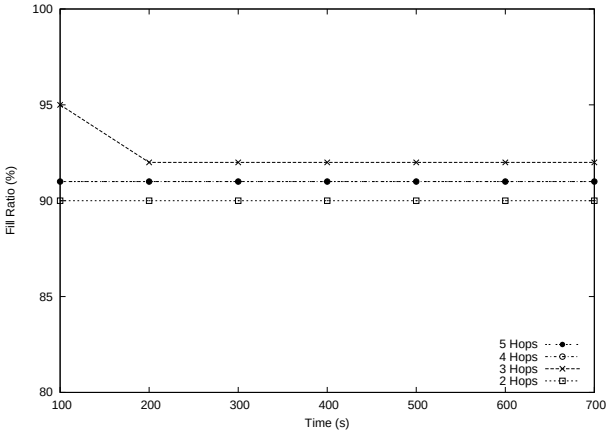

Figure 5.7: Fill Ratio

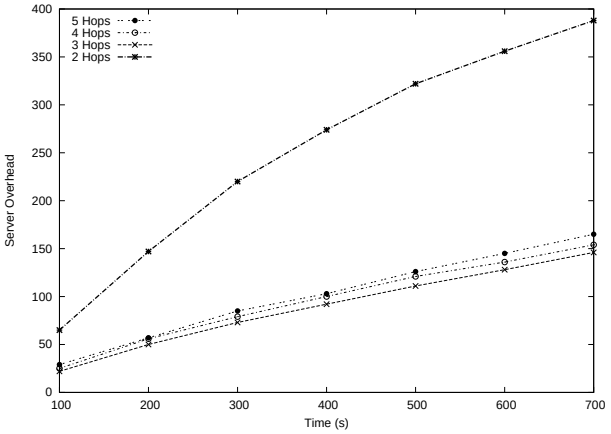

Figure 5.9: Server Overhead

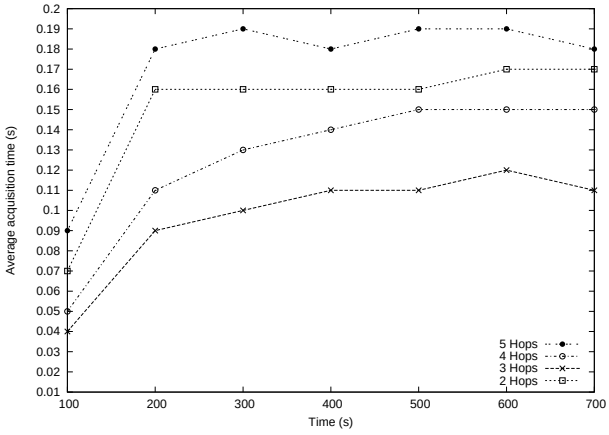

Figure 5.8: Acquisition Time (s)

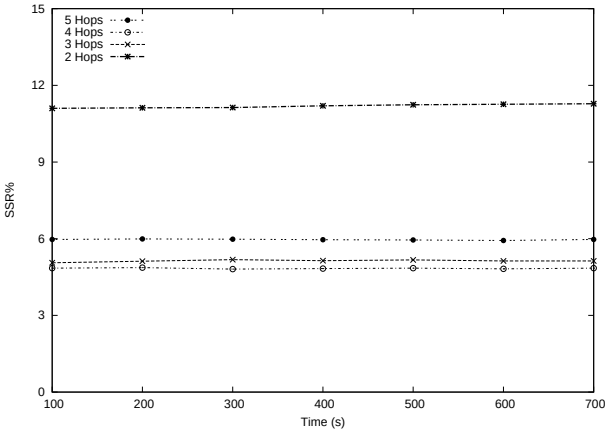

Figure 5.10: Server Request Ratio (SRR%)

Table 5.3: Link Quality Parameters

Parameter	Value
Radio Propagation Model	Two Ray Ground
Antenna Type	Omni Antenna
Transmit Power	0.281838
Signal Frequency	9.14e+08
Transmit Antenna Gain	1
Receive Antenna Gain	1
Simulation Time	2000 (s)
Mac Protocol	IEEE 802.11
Wireless bandwidth link	11Mbps
Mobile node RXThresh	70m
RM and Routers RXThresh	250m

In the next set of experiments, we evaluated the number of 3D data packet loss as we vary the simulation time. The results are presented in Figure 5.13. As illustrated in the figure, we can see that the average number of packet loss increases linearly as we increase the simulation time. The least number of packet loss is noticed for $\delta = 2$ hops, while the highest number of packet loss is observed for $\delta = 5$ hops. This confirms our thoughts that state that when the number of hops is small, the server would be responsible for sending the relevant data, and the packet loss would be therefore small. However, as we increase the number of hops, the packet loss increases, given that many fragile links are involved and sending large 3D data packets would be challenging.

Based on the results obtained, we can clearly notice that there is a trade off between the data acquisition and the packet loss. A small number of hops induces a high data acquisition time imposed by the system, given that the requester has to acquire the relevant 3D data from the server, involving therefore remote access costs and times. Although, a large number of hops allows the requester to locate the relevant data, it induces a high packet loss given that the environment is dynamic, and the peers are mobile causing therefore fragile links. We believe that the best value for δ should be 3.

In the proposed 3D data delivery mechanism, once the relevant 3D is found and has to be delivered, the source decides whether to send the 3D packet to its predecessor, i.e., the one that delivered the data request, or to the closest router. This condition has been added, since large 3D packets are sent. To ensure the reliable transfer of data, the RSS between the source and its predecessor is measured. The packet is sent to the predecessor if the signal strength is larger than a threshold λ, otherwise the packet is sent to the closest router. In the following set of experiments, we fixed the value of δ to 3 hops and we varied the values of λ. We expressed the received signal strength RSS

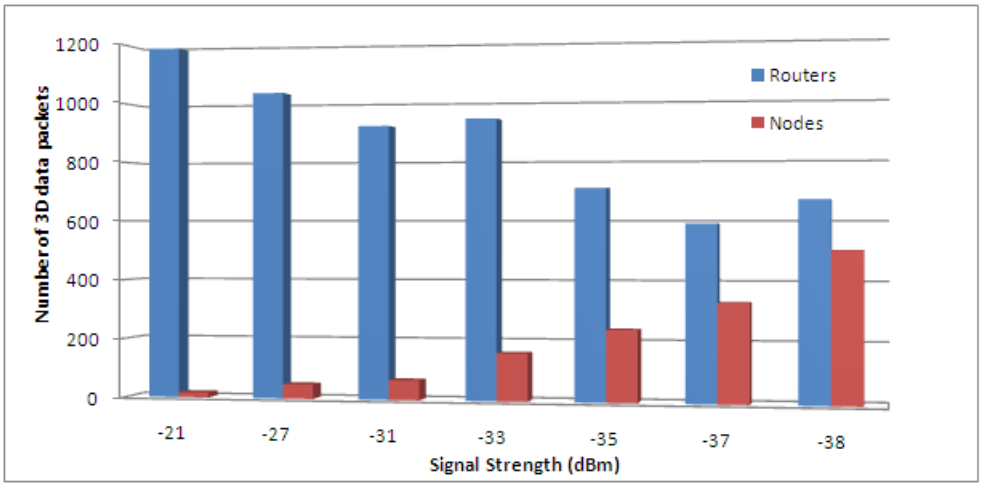

Figure 5.11: Number of Requests routed by relay nodes/routers when λ varies.

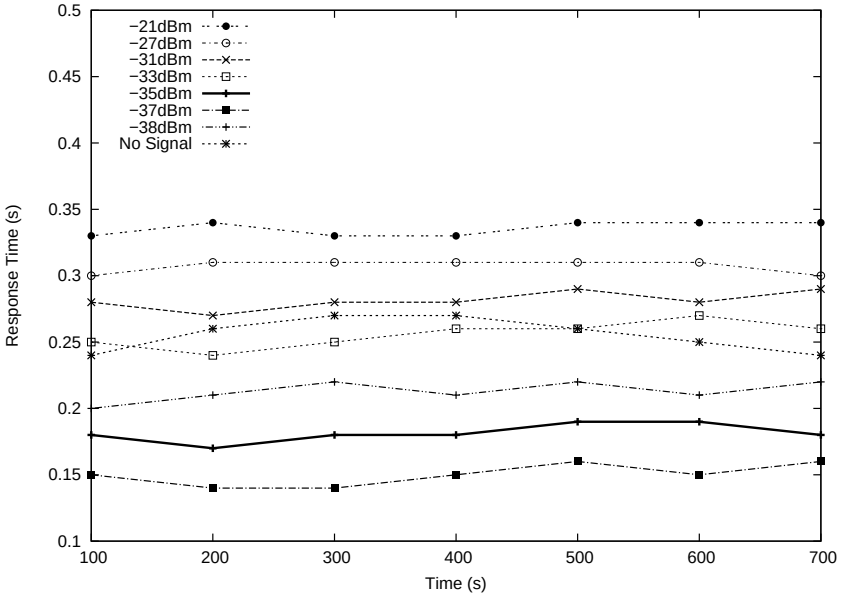

Figure 5.12: Response Time when λ varies.

Table 5.4: RSS-Distance Mapping

P_r (dBm)	Distance (m)
-38 dBm	35m
-37 dBm	30m
-35 dBm	25m
-33 dBm	20m
-31 dBm	15m
-27 dBm	10m
-21 dBm	5m

in terms of distance based on the Equation 5.2. Recall that RSS between two nodes is inversely proportional to the distance that separates the relative nodes [93].

$$P_r = \frac{P_t * \alpha^2}{(4\pi)^2 * R^2} \quad [W] \tag{5.2}$$

For this set of experiments, we used the following settings presented in Table 5.3. We chose 7 values of λ and we compared the obtained results in order to find the best signal strength threshold to be adopted in the rest of the experiments. We also tested the protocol, when the RSS is not measured and that the 3D packets are sent back to the original requester based on the path taken by the data request packet. The threshold values that have been used in the experiments as well as their equivalent distances (distance between the sender and the receiver) are presented in Table 5.4.

In this set of experiments, we evaluated the average acquisition time when varying the signal strength. The results are expressed in terms of the simulation time and are illustrated in Figure 5.12. As shown in the figure, as the simulation time increases, the average acquisition time obtained when the signal strength is equal to -37dBm remains among the best compared to the rest of the signal strengths.

This graph when presented alone does not express the expected results given that the best acquisition time is obtained when the signal strength is low, i.e., when the distance between the source and the next hop is high. However, if we take a look at the Figure 5.11, we can clearly see that results of Figure 5.12 are correct. Figure 5.11 illustrates the number of 3D packets that have been routed using the routers instead of using the original path taken by the data request. When the signal strength is low, i.e., RSS = -21dBm (distance is 5m), we clearly notice that a high number of 3D packets (95% of the 3D packets) have been routed using the routers, while only a very small

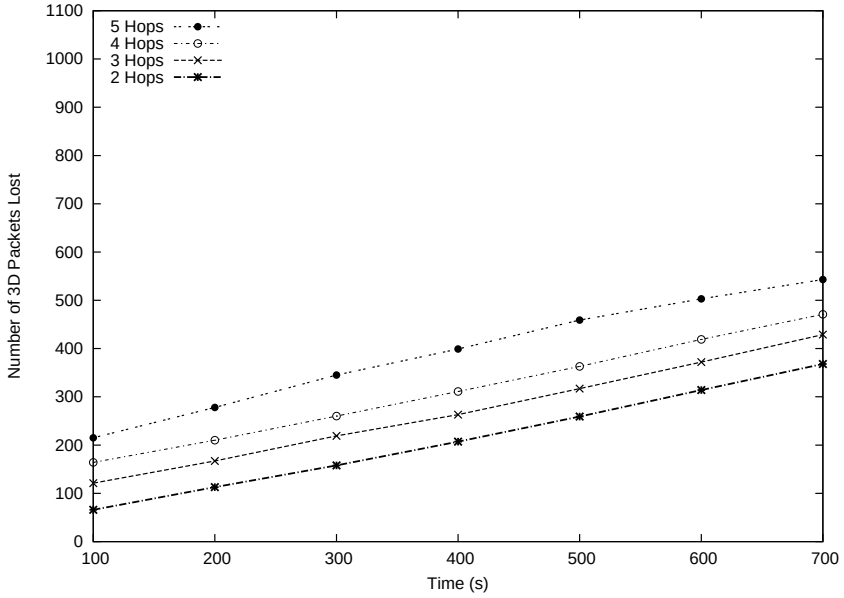

Figure 5.13: Number of 3D packets lost as the simulation varies

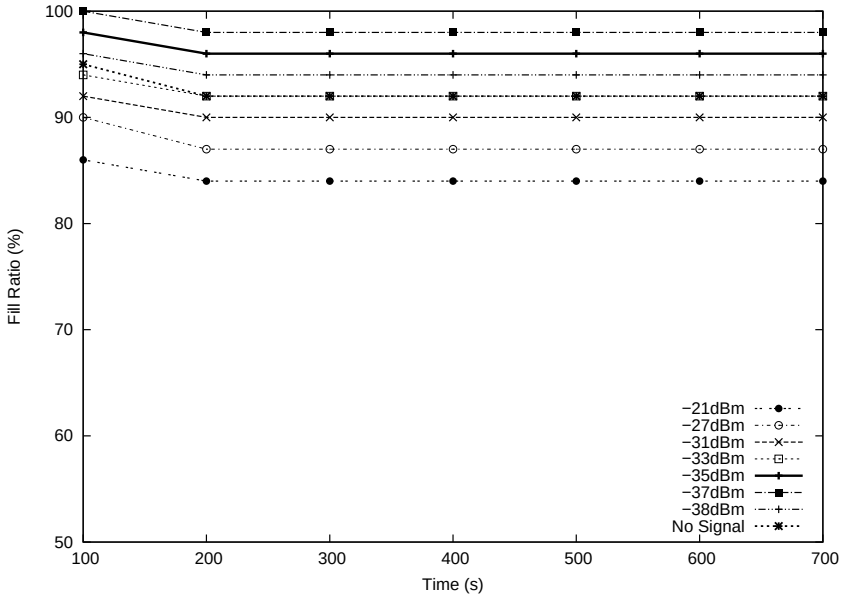

Figure 5.14: Fill Ratio.

number of 3D packets have been sent using the original path taken by the data request. As we increase the value of the signal strength threshold, we notice that the number of 3D packets using the original path is increasing, while the number of packets using the routers is decreasing. These results can be easily explained. As a matter of fact, when the threshold of the signal strength is high, i.e., the distance that separates the two users is small, almost all of the 3D packets will be delivered to the original requester through the routers given that it is hard to find a peer located in the sphere of radius λ. This may lead to congestion and consequently to a significant delay, given that a high number of packets is queuing up at the routers. However, when the signal strength threshold is low, i.e., the distance that separates the source and the next hop is high, more users may be available and may participate in the content delivery.

In the next set of experiments, we evaluated the fill ratio representing the rate of 3D objects that have been successfully received and displayed. The results obtained while evaluating the fill ratio as a function of the simulation time are presented in Figure 5.14. As shown, when the simulation time is 100s, the fill ratio varies between 87% for RSS = -21dBm (distance is 5m) and 100% for RSS = -37dBm (distance is 30m). As we increase the simulation time, the curves maintain the same behavior and the worst results are obtained for RSS = -21dBm with 84% of fill ratio at 700s, while the best fill ratio, 98%, is obtained when RSS= -37dBm. We can also note that "No signal" and RSS = -33dBm have very comparable results as we increase the simulation time.

We measured the 3D packet loss ratio for the different values of λ as we vary the simulation time. The results are illustrated in Figure 5.15. As shown in the Figure 5.15,as we increase the simulation time, we notice that the least rate of 3D packet loss is recorded for λ equal to -37dBm, while the highest 3D packet loss is revealed for λ equal to -21dBm. The obtained results are easily explained. As a matter of fact, as mentioned earlier, the packet loss can be caused by several factors including signal degradation, packet drop due to congestion or to the distance that separates the transmitter and the receiver. Definitely, the distance factor can not be the cause given that packet loss for λ = -21dBm (distance is 5m) should be less than λ = -37dBm (distance is 30m). However, the congestion is most probably the cause for the packet loss in our case. In our protocol, the transmitter measures the signal strength to the receiver and if it is less than a threshold, the packet is sent to the closest router. When the signal strength is strong, i.e., the distance that separates the transmitter and the receiver is small (5m for example), not many users are included in the desired radius which leads to the 3D packets being routed to the closest router. This leads to congestion at the router and

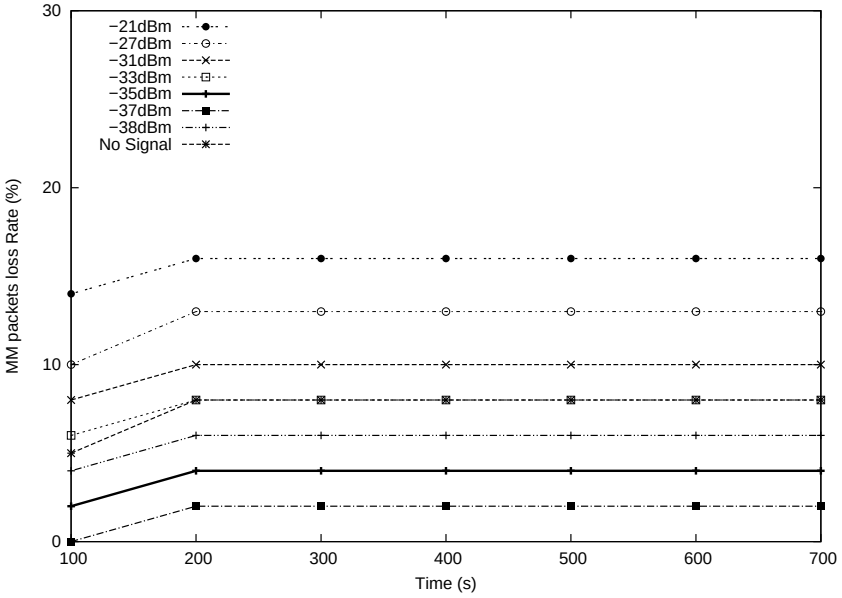

Figure 5.15: 3D data packets loss as the simulation varies

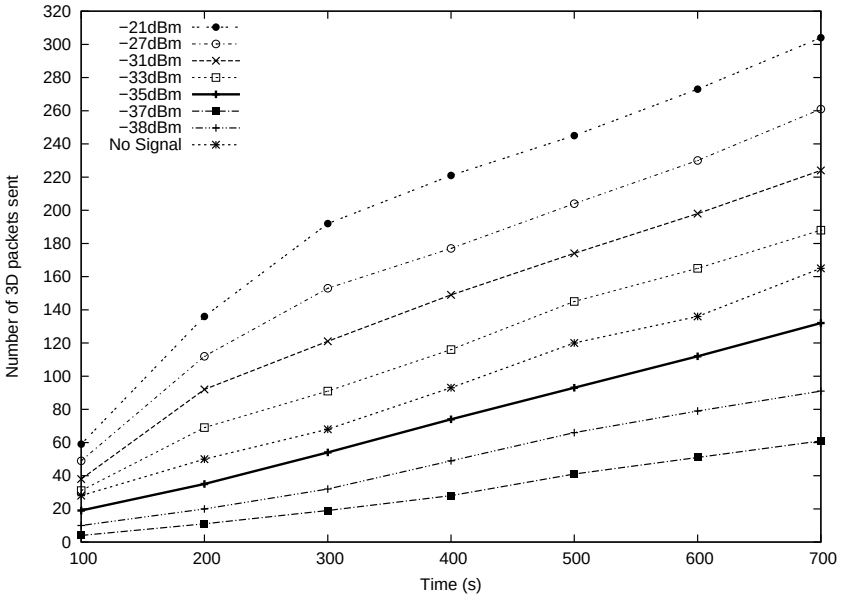

Figure 5.16: Server Overhead

3D packets are therefore dropped, which increases the packet loss rate. In the opposite case, when the signal strength is low (-38dBm for example), a higher number of users may participate in the delivery process, which decreases the packet loss rate.

The server overhead, the server request ratio ($SRR\%$) as well as the average number of 3D data packets sent by the nodes are illustrated in Figures 5.16 and 5.17 respectively. These experiments have been added given that they illustrate the P2P interaction among the sources and how it is favored over the client-server approach. A factor that can indirectly show how the P2P system behaves, is the server overhead that can be expressed in terms of the average number of 3D data requests answered by the server. The smaller this number is, the higger the P2P interactions is. As illustrated in Figure 5.16, the number of 3D data packets sent by the server increases linearly for the different values of λ and as the simulation time progresses. We can clearly see that the highest number of 3D data packets served by the server is recorded for λ = -21dBm, while the least number of 3D data packets served by the server was seen for λ = -37dBm. This confirms again the results obtained earlier and that explained that for λ = -21dBm, there are no enough relay peers within a radius of 5m. This implies that the 3D data packets are forwarded to the routers which leads to a high rate of packet loss and a large number of retransmission. By all means, the retransmission are carried out by the server, thereby the high server overhead. From the graph, we can also note that results obtained when we do not test the signal during the transmission of the 3D packets, are very comparable to the results obtained when λ = -33dBm. These results are also confirmed in Figure 5.17 that illustrates the SRR%. A lower SRR% means that the interaction between peers is high and that the system scalability can be improved. As shown in Figure 5.17, the lowest SRR% 2.5% is obtained for RSS= -37dBm, while the highest SRR% 8.5% is obtained when RSS= -21dBm.

From the results obtained, we can notice two important facts: In the first fact, when the signal strength threshold between the nodes is high, there are not enough nodes within the circle, and 3D data packets will be routed to the closest router which leads to 3D data packets loss due to congestion. While in the second fact, when the signal strength is very weak, the packet loss is also high mainly due to the large packets' size and to the wireless medium. Therefore, and based on the results obtained, we choose to have the value of λ equal to -37dBm which corresponds to a distance of 30m between the peers.

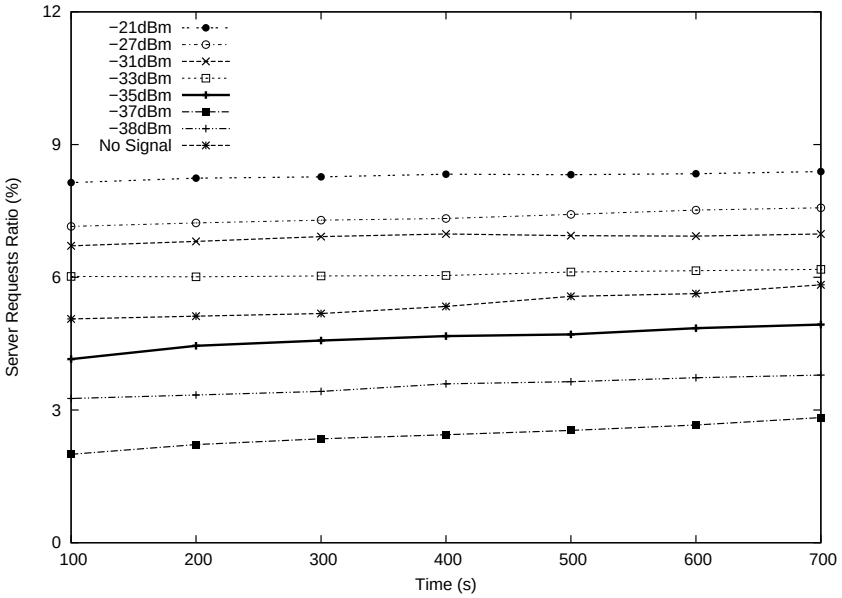

Figure 5.17: Server Request Ratio (SRR%)

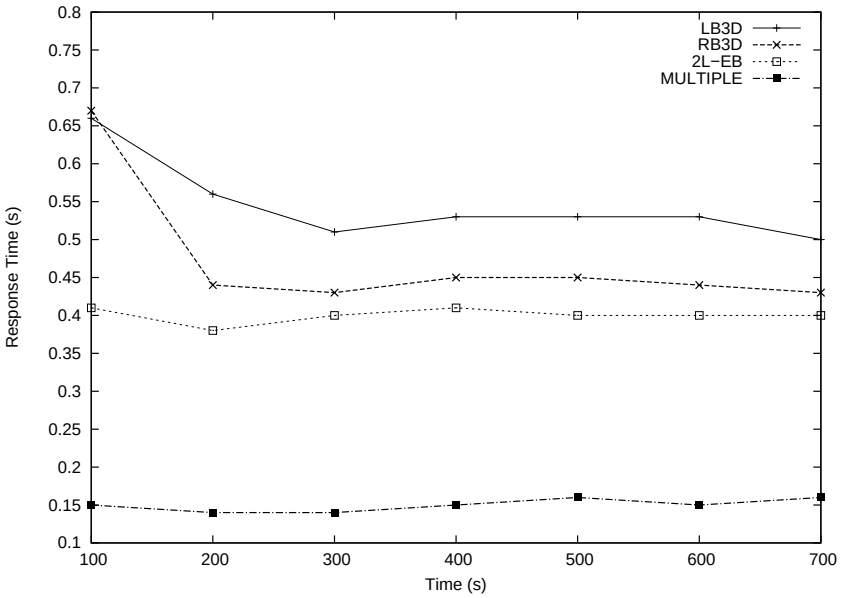

Figure 5.18: Average response time

5.2.3 MULTIPLY vs LB3D vs RB3D vs 2L-EB: A Comparative Study

In this section, we should compare the performance of MULTIPLY with our previously presented supplying partner algorithms, namely LB3D, RB3D and 2L-EB. We recall that LB3D is a location based supplying partner protocol, RB3D is a resource based supplying partner protocol and 2L-EB is an energy based supplying partner protocol. All three protocols (LB3D, RB3D, 2L-EB) are one-hop supplying partner selection protocols.

In the first set of experiments, we analyzed the response time for the four protocols. The results are illustrated in the Figure 5.18. As shown, MULTIPLY has a significant low response time compared to the LB3D, RB3D and 2L-EB. As the simulation time progresses, MULTIPLY improved the system's response time by 63% compared to 2L-EB, by 67% compared to RB3D and 73% compared to LB3D. These results are expected. In fact, and as explained earlier, when the number of hops is small (i.e., equal to 1 in the case of LB3D, RB3D and 2L-EB), the requester would have a high chance to acquire the relevant data from the server, which induces remote access cost and a long data acquisition time. As we increase the number of hops, the requester has a higher number of potential sources and the data could be located and acquired faster. Since many hops are involved, there is a high probability that the packet would be dropped due to the fragility of the wireless link. However, given that in MULTIPLY the signal strength is analyzed before sending the 3D data packet, the probability of packet drop is therefore decreased and the response time is thereby increased.

The server overhead is analyzed and the obtained results are illustrated in Figure 5.19. As shown, MULTIPLY has a very low server overhead when compared to LB3D, RB3D and 2L-EB. MULTIPLY reduced the server overhead by 87% when compared to 2L-EB, 92% when compared to RB3D and 93% when compared to LB3D. When testing the server request ratio (SRR%), as illustrated in Figure 5.20, MULTIPLY outperforms 2L-EB, RB3D and LB3D. As shown, the SRR% for MULTIPLY is around 2%, while it is 15%, 22% and 25% for 2L-EB, RB3D and LB3D respectively. These results confirm that MULTIPLY greatly reduces the server intervention and favores the P2P interaction among peers, given that the requester has the ability to find multihop potential sources.

The fill ratio that represents the rate of 3D objects successfully received and displayed is illustrated in Figure 5.21. 2L-EB and RB3D outperform MULTIPLY and LB3D by having 100% of fill ratio, compared to 98% and 97% for MULTIPLY and LB3D respectively. These obtained results for MULTIPLY are mainly due to the fact that various

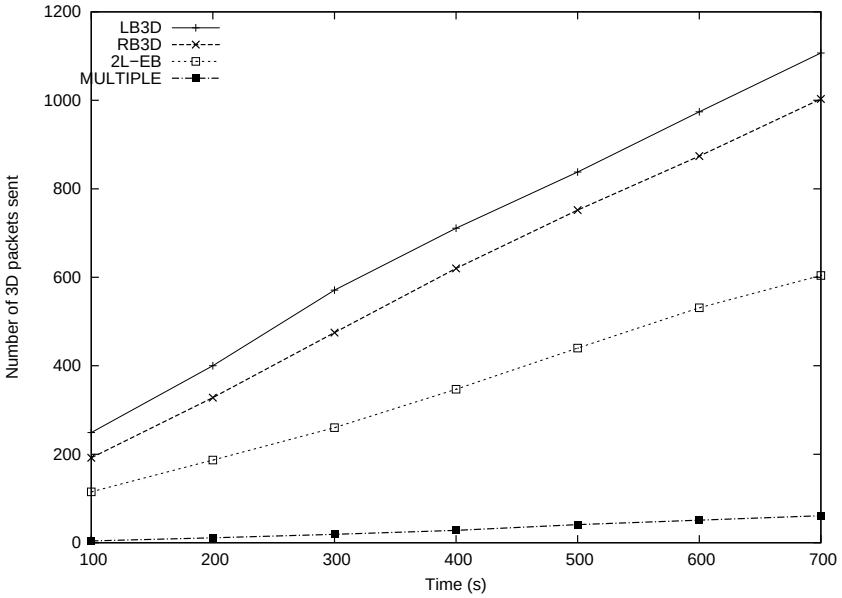

Figure 5.19: Server Overhead

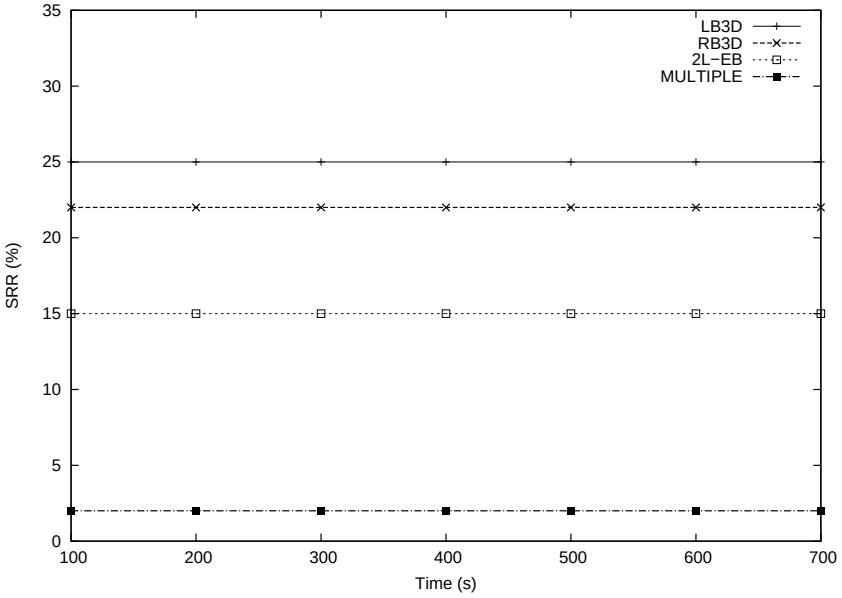

Figure 5.20: Server Request Ratio (SRR%)

wireless links are involved, and given that the environment is dynamic, the wireless links become therefore fragile. This is confirmed in Figure 5.22 that illustrates the packet loss rate. We can clearly see that for MULTIPLY and LB3D, the packet loss is 2% and 3% respectively.

Based on the results obtained, we can notice that there is a trade off between the data acquisition time, packet loss and P2P interaction. When the number of hops is small, and that no relay nodes are involved, the packet loss rate is very small, however, the acquisition time is long and the P2P interaction is limited. As we increase the number of hops, the packet loss slightly increases, while the acquisition time as well as the P2P interaction are greatly improved.

5.2.4 Impact of Mobility

In this section, we aim at studying the effect of the mobility on our proposed multihop supplying partner protocol MULTIPLY. To do so, we are taking into consideration four mobility models, namely: Random walk (brownian) [47], random waypoint [48], random direction [47,48] and reference point group mobility (RPGM) [47,48]. Before proceeding further with the experiments, let us first review the four mobility models we shall use in our experiments.

Mobility models:

1. **Random Walk Mobility Model (Brownian)**: This model, that is widely used by the research community and that is referred to as the Brownian model, has been designed to mimic the unpredictable movements of the particles in physics [47]. The Brownian model is a memoryless mobility model given that it does not keep track of the past locations and speeds of a given mobile node. In this model, the mobile node moves randomly in the environment. Starting from its current location, the mobile node chooses a random direction and speed (independent from its past speed and direction) to reach his following position. The movement change of the mobile node is determined either following a time constant or a distance traveled [48]. In other words, the mobile node's new position is changed every interval of t seconds or every distance traveled d meters. The range of the speed and direction are predefined at the beginning of the simulation. The new speed follows a uniform distribution or a Gausssian distribution and has to be in the ranges [speed$_{min}$, speed$_{max}$] and the direction in the range of [0, 2π]. If the new position of the

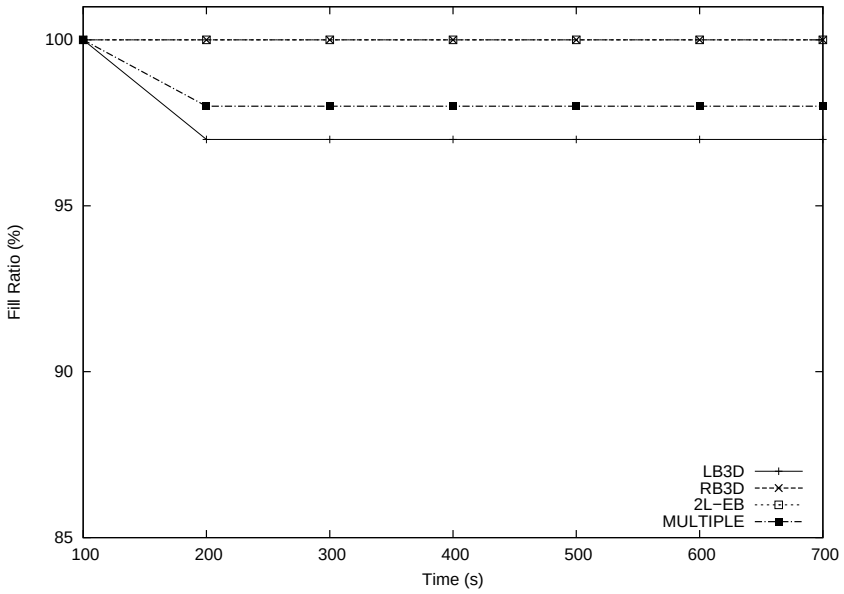

Figure 5.21: Fill Ratio.

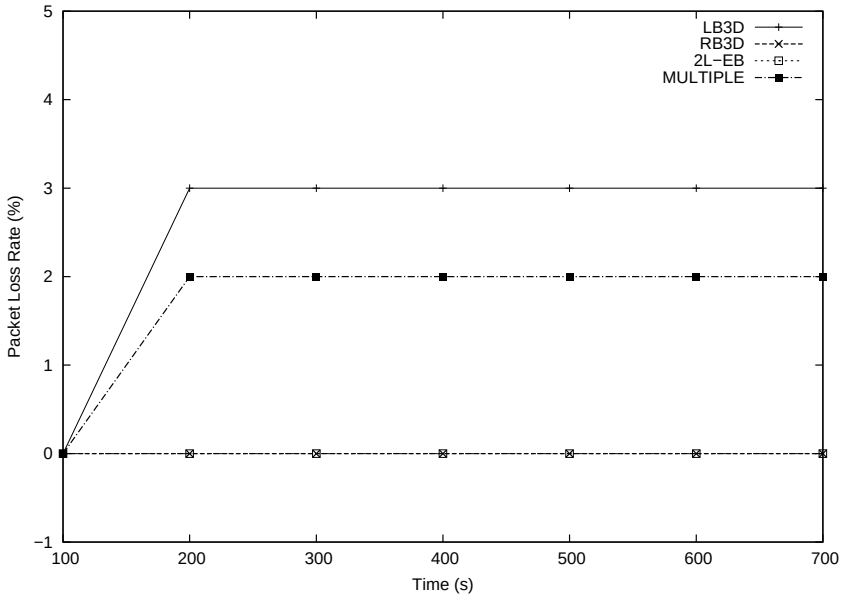

Figure 5.22: Packet Loss Rate.

mobile node is outside the simulation's environment borders, than the mobile node changes his movement, and is bounced back to the simulation environment following the incoming direction.

2. **Random Waypoint Mobility Model (RWPM)**: The Random Waypoint mobility model is considered a benchmark mobility model for MANET routing protocols. It is very similar to the random walk mobility model with the addition of pause times between changes of the mobile node's movements [48]. The mobile node moves randomly in the environment and before starting a new random movement, it stays at its current location for a certain period of time, called a pause time T_{Pause}. Once the pause time expires, a random direction and speed are chosen and the mobile node travels towards its new position at its new speed. The new speed follows a uniform distribution and has to be in the interval [V_{min}, V_{max}]. In the RWPM, V_{max} and T_{Pause} determine the behavior of the network. A network is considered relatively stable if T_{Pause} is large and V_{max} is small [47], and is considered highly dynamic in the opposite case, i.e., V_{max} is high and T_{Pause} is small. RWPM is widely used, however it suffers from one major issue namely the node wave. In fact, as the simulation time is increased, the node density becomes unbalanced, where the nodes tend to cluster at the center region, leaving the borders of the environment completely empty.

3. **Random Direction Mobility Model (RDMM)**: The RDMM has been designed to address the density wave problem that the RWPM suffers from [47, 48]. In RDMM, the mobile node chooses a random direction and speed similar to RWPM, and travels in that direction towards the border of the simulation environment. Upon reaching the border, the mobile node pauses there for T_{Pause}. Once T_{Pause} expires, the mobile node chooses a new direction having an angle in the interval [0, π] and travels towards the border in that direction.

4. **Reference Point Group Mobility Model (RPGM)**: Brownian, RDMM, and RWPM have several limitations given the random movement of the mobile nodes that do not represent realistic scenarios. In fact, the mobile nodes have unusual mobility patterns such as a sharp stop or a sudden turn. Moreover, all of the above mentioned random models do not respect the geographic restrictions of the movement and do not keep any temporal or spatial dependency of the velocity of the mobile node [47]. The RPGM respects the spatial dependency, where the mobile

nodes moves with respect to the movements of other mobile nodes [48]. In RPGM, the nodes are clustered into groups with a group leader responsible for determining the movement behavior of the entire group [47]. The movement of the group leader can be represented by a motion vector that characterizes the movement of the leader as well as the movement of the entire group. The movement of the group leader also defines the speed and direction of the mobile nodes movement. The mobility of each member of the group follows a reference point that follows the movement of the group leader.

Let us now turn to our experimental results. In the first set of experiments, we studied the success rate of the protocol when changing the mobility models. The success rate is represented in terms of the fill ratio, i.e., the ratio of the successfully received objects over the total number of objects within the user's field of view. The results are illustrated in Figure 5.23. As shown, RPGM has the highest fill ratio with 99% while Brownian has the worst fill ratio with 96%. These results are expected. In fact, in RPGM, the mobile nodes are clustered into groups and move with respect to the movements of other mobile nodes. Therefore, a given requester, will always find several sources within its neighbors, given that they have the same movement, the same view and will therefore need the same 3D objects. Moreover, given that the mobile nodes are clustered into groups and moving towards the same direction, they are therefore near to each other. 3D Data acquisition time as well as the packet loss will thereby be significantly reduced. This is confirmed in Figures 5.24 and 5.25 respectively. In the Borwnian model, the nodes move randomly in the environment. Therefore, a given requester can not find easily potential sources to acquire the relevant 3D data. Moreover, given that the nodes are not moving in the same direction, sending the required 3D data to the requester has to be most of the time through the routers, which increases the 3D packet loss and the 3D data acquisition time. This is confirmed in Figures 5.24 and 5.25 respectively. RDMM and RWPM have comparable results and their fill ratio is 98%.

In the second set of experiments, we studied the packet loss rate as we vary the simulation time for the different mobility models. The results are illustrated in Figure 5.24. As shown, RPGM has the least rate of packet loss while the Brownian model has the highest rate with respectively 0.5% and 4%. These results can be easily explained. In RPGM, the mobile nodes are clustered into groups moving towards the same direction. The RSS between a given source and its predecessor is strong due to the proximity of nodes. This leads the 3D data packet delivery to be achieved through the mobile nodes, reducing thereby the data packet loss rate. Contrariwise, in the Brownian model, the

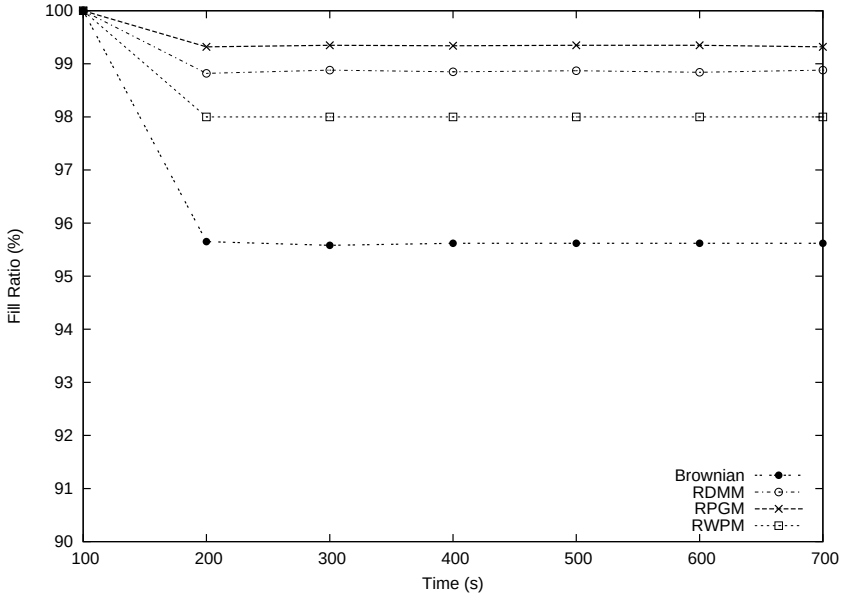

Figure 5.23: Fill Ratio

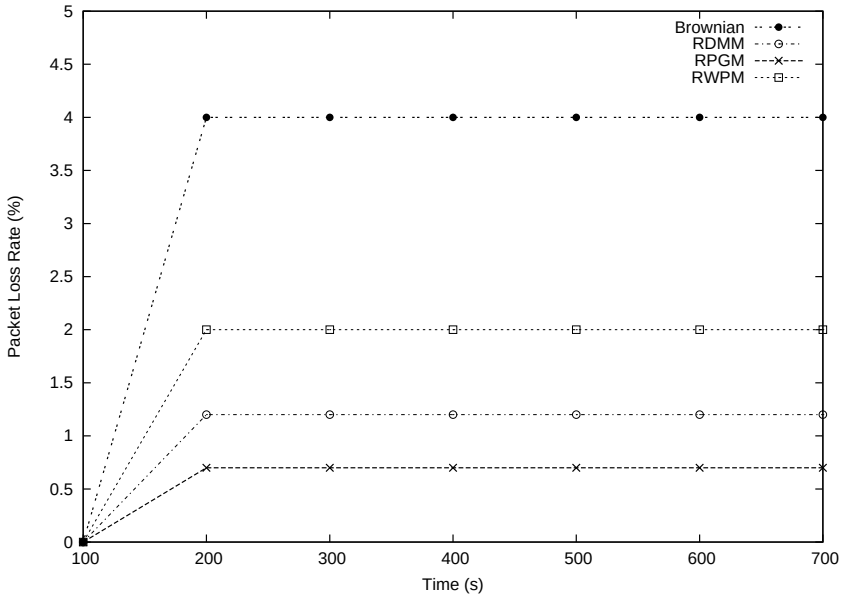

Figure 5.24: 3D Data packet loss.

mobile nodes move randomly in the environment. This makes it difficult for a given requester to find potential sources and most of the data delivery actions will be performed through the routers. By all means, this leads to a high packet loss given that data packets queue up at the router. As for RDMM and RWPM, the packet loss rate is comparable with 1% and 2% respectively. RDMM gives better results than RWPM given that the mobile node pauses for T_{Pause} at the final position before changing its direction allowing the neighboring nodes to acquire data. During T_{Pause}, the data transmission is stable i.e., there are no packet loss due to the wireless medium or link breakages. As a conclusion, no matter what the mobility model is, MULTIPLY behaves very well and the data packet loss rate remains very low not exceeding 4%.

The 3D data acquisition time is also studied and represented in Figure 5.25. As illustrated, the highest data acquisition time is obtained with the Brownian model while the lowest data acquisition time is obtained with the RPGM model. These results are expected. As explained earlier, given that in the RPGM model, the mobile nodes are moving in groups, the data delivery is achieved via the mobile nodes which leads to a low data acquisition time. In the Brownian model, the data delivery is accomplished through the routers which leads to high data packet loss and a long time to acquire the 3D data.

The server request ratio has also been studied and the results are presented in Figure 5.26. As illustrated, the SRR% is very comparable to the Brownian, RDMM, and RWPM models with a value of 3%. As for the RPGM model, the SRR% is very low with 1% mainly due to the group mobility nature. As a matter of fact, when the mobile nodes are clustered and moving in the same direction, a given requester has a higher chance to find the potential sources within its neighbors and to acquire thus the 3D data quickly without any server's help. However, when the mobile nodes are moving randomly in the environment (in the case of Brownian, RDMM, and RWPM), a given requester expresses difficulties to find the source and may lead sometimes to inquiring from the server. Nevertheless, the SRR remains very low and does not exceed 3%, given that our MULTIPLY protocol tries to find the potential sources before requesting the server.

5.3 Conclusion

In this chapter, we presented our proposed multihop supplying partner protocol coupled with an efficient content delivery technique for MANET-based 3D streaming systems. Our proposed protocol aims at finding the potential sources that hold the relevant 3D

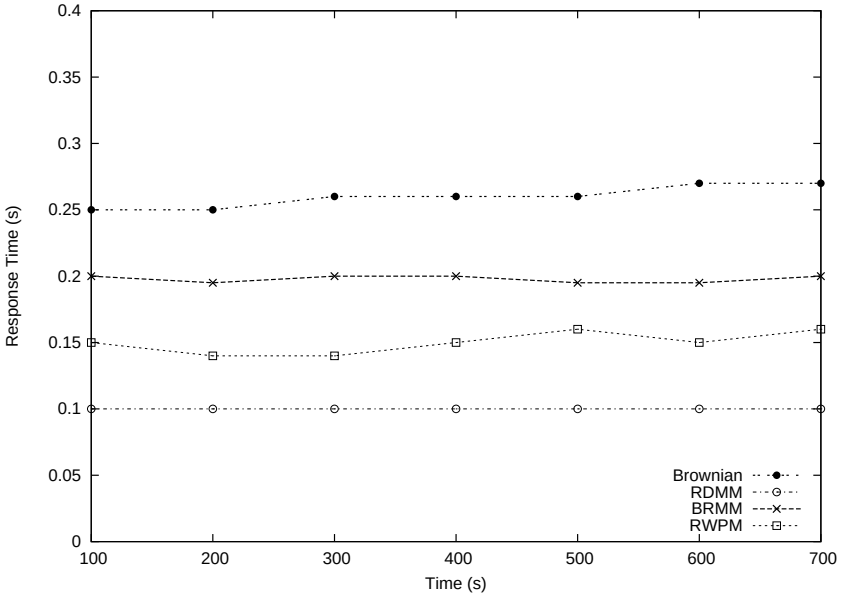

Figure 5.25: 3D data acquisition time.

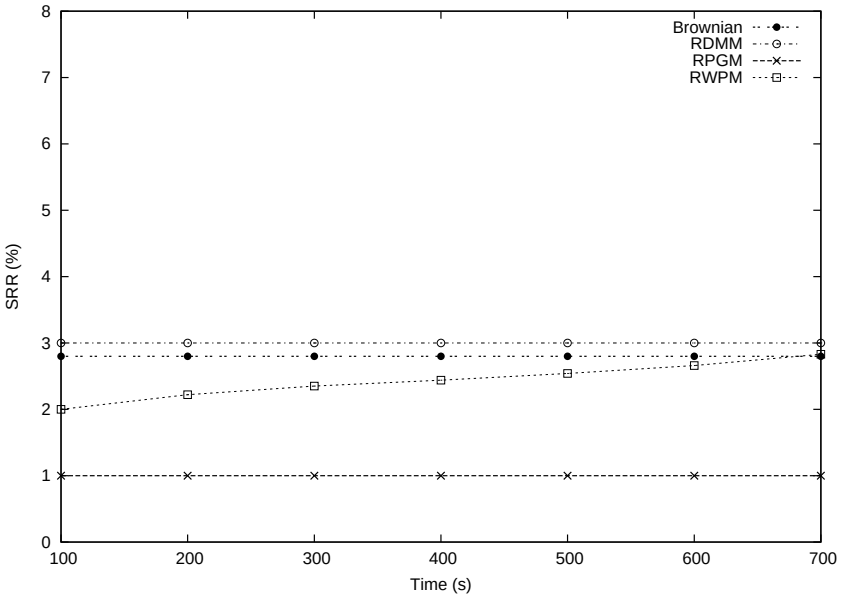

Figure 5.26: Server request rate.

data, while overcoming the wireless link limitations by analyzing the RSS when sending back the data. The performance evaluation of our protocol shows that it improves the data acquisition time while keeping a low packet loss. We also studied the impact of mobility on the performance of MULTIPLY. In order to show this impact, we applied different mobility models, namely: RPGM, RWPM, RDMM and the Brownian models. The performance evaluation of our protocol shows that MULTIPLY performs better with the RPGM model where the data acquisition time, packet loss and SRR% are greatly reduced. The obtained results with the other mobility models remain very promising given that the delay and packet loss are kept very low.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

In this thesis, we first presented a state of the art of 3D streaming techniques, as well as a state of the art of supplying partner techniques for multimedia streaming. We then presented our P2P based 3D streaming system which we refer to as MOSAIC as well as a suite of supplying partner protocols for P2P 3D streaming systems over thin mobile devices. This thesis presents four main contributions: (1) The design and implementation of MOSAIC our P2P based 3D streaming system over thin mobile devices; (2) The design and implementation of supplying partner protocols namely location based and resources based supplying partner protocols; (3) The design and implementation of two energy management control mechanisms which comprises two protocols namely, the One-Level Energy based Supplying Partner Protocol (1L-EB) and Two-Level Energy based Supplying Partner Protocol (2L-EB); (4) and finally the design and implementation of MULTIPLY which is a multihop supplying partner selection protocol.

MOSAIC is intended for mobile augmented reality (MAR) based class of applications using thin mobile devices, and has therefore to satisfy the system's and user's requirements. From the system perspective, MOSAIC has to improve the system's efficiency, which can be expressed in terms of the network bandwidth utilization, as well as the system's performance that can be captured using concepts such as *response time*, *data delivery*, and *scalability*. As for the user's requirements, MOSAIC has to take into account the acquisition time that characterizes the user's experience with the 3D streaming over thin mobile devices. In order, therefore, to decrease the network bandwidth over-utilization, we applied an IM technique assisted by a location-based filtering of messages

aiming to decrease the computation overhead. We also applied a dead reckoning technique in order to decrease the communication overhead.

MOSAIC comprises an efficient source discovery protocol, and two supplying partners protocols for P2P 3D streaming which we refer to respectively as a location-based (LB3D) and a resources-based (RB3D) supplying partner protocols.

The source discovery protocol is based on the AOI collision detection mechanism and the verification of the list of neighbors. The former aims at allowing each requester to determine which of its neighbors' AOIs collide with its own AOI. The goal of this mechanism is to discover the neighbors that are proximate to the requester and that have a high probability to hold the relevant data. As for the verification of the list of neighbors, it allows the requester to discover not proximate peers who still hold the relevant 3D data cached in their buffers.

As for the supplying parner protocols, our location based supplying partner protocol (LB3D) aims at choosing the best supplying partners taking into account their proximity to the requester. A double prioritization is applied. First, sources are ordered based on their proximity to the requester. Then a second prioritization is applied to sort the sources depending on the amount of required 3D data and its availability on each source. The location-based supplying partner protocol allows the requester to choose multiple suppliers and to distribute the load among them, optimizing the time required to get the 3D data.

The resources-based supplying partner protocol (RB3D) intends to improve the streaming performance of the selected suppliers and balance the load among them by not overloading them and relieving the loaded ones. Based on a set of criteria and rules, RB3D examines each potential source and classifies the selected ones based on their status and their realization ratio. The supplying partner selection is performed in a descending order with the lowest priority assigned to the sources that have a high RR%, whereas the highest priority is assigned to sources that have a low RR% and that are not busy streaming data to other requesters. The performance evaluation of RB3D shows that it succeeds at balancing the load among sources and at relieving the overloaded ones. When comparing RB3D to LB3D, results show that RB3D improves the fill ratio by 5% given that it takes into account the streaming activity of the supplying partners. Moreover, RB3D reduces the server overhead by 12% as well as the data acquisition time by 30% when compared to LB3D.

Given that we are dealing with thin mobile devices with limited capabilities and knowing that frequent 3D streaming overuses the mobile devices resources, it is crucial

to take the energy demands into account when designing a supplying partner protocol for P2P 3D streaming applications. A lack of energy degrades the system performance considerably in terms of response time and impedes the P2P behavior. We therefore propose an energy management control for supplying partner protocols. Our proposed scheme, which we refer to as 2L-EB, is based on two levels: the requester side and the supplier side, which play important and complementary roles. The supplier has the responsibility for monitoring its own load and informing its neighbors when it is detected overloaded. As for the requester, it is responsible for examining the load on its suppliers and distributing it among non overloaded suppliers reducing their energy consumption. The performance evaluation of our energy management control protocol shows that it improved the system's overall performance in terms of response time and energy savings. A variant protocol of 2L-EB, which we refer to as 1L-EB has been used to analyse the performance of 2L-EB. Recall that 1L-EB is based on the requester side only. The comparative study of 1L-EB and 2L-EB shows that 2L-EB reduces the energy consumption by 40% as well as the server overhead by 56%. Morever, 2L-EB reduces the data acquisition time by 49% when compared to 1L-EB. When comparing 2L-EB to RB3D and LB3D, results show that 2L-EB reduces the energy consumption by 45% and 60% respectively and the server's streaming activity by 50% and 20% respectively.

The previously mentioned supplying partner protocols select sources that are at one hop from the requester. However, several other sources remaining at multiple hops from the requester may hold the relevant 3D data and may be better candidates to be selected. We therefore propose our multihop supplying partner protocol which we refer to as MULTIPLY. Our protocol is coupled with an efficient content delivery technique, which considers the wireless link limitation by examining the signal strength of the route taken before sending the 3D data packet. The performance evaluation of MULTIPLY shows that it improves the data acquisition time while keeping a low packet loss. The comparative study of MULTIPLY, 2L-EB, RB3D and LB3D shows that MULTIPLY improves the response time by 63% compared to 2L-EB, 67% compared to RB3D and 73% compared to LB3D. Moreover, MULTIPLY reduces the server overhead by 87% when compared to 2L-EB, 92% when compared to RB3D and 93% when compared to LB3D. As for the fill ratio, results show that 2L-EB and RB3D outperform MULTIPLY and LB3D by having 100% of fill ratio, compared to 98% and 97% for MULTIPLY and LB3D respectively.

6.2 Future Work

For our future work, we intend to work on the design and implementation of a secure supplying partner protocol as well as a visibility determination protocol. Further studies involve investigating DDM techniques similar to those used in large scale distributed simulations may be required. We plan also to study our protocols analytically to understand further the behavior of our proposed schemes.

P2P based 3D streaming system over thin mobile devices includes several peers collaborating together to stream 3D data. However, this may create security concerns regarding the trustability of the sources and the integrity of the 3D data streamed. Therefore, there is a strong need to design a secure supplying partner protocol for 3D streaming class of applications over thin mobile devices. In the future, we plan to design and implement a secure supplying partner protocol that aims at authentifying the sources and insuring their trustability in one hand; and guarantying the integrity and protection of the 3D data streamed to the requesters in the other hand.

As for our second future work, we plan to design and implement a visibility determination protocol. Given that thin mobile devices with limited capabilities are used, streaming all of the objects located in the user's field of view or its AOI is not considered efficient and may overuse the mobile device resources. We plan therefore to design and implement a protocol capable of determining the objects that need to be streamed to the user taking into account its interests as well as its available mobile device resources.

Bibliography

[1] M. Halvorsen, T. Plagemann, and M. Siekkinen, *Video Streaming Over MANETs: Reality or Fiction?*, Proceedings of the 4th International Mobile Multimedia Communications Conference (MobiMedia '08), 2008.

[2] S.Y. Hu, et al., *FLOD: A framework for peer to peer 3D streaming*,Proceedings of the IEEE Conference on Computer Communications INFOCOM, pages: 1373-1381, 2008.

[3] S.Y. Hu, S.C. Chang, J.R. Jiang, *Voronoi state management for peer to peer MMOG*, 5th IEEE In Consumer Communications and Networking Conference, pages: 1134-1138, 2008.

[4] WL Sung et al., *Selection strategies for peer to peer 3D streaming*, Proceedings of the 18th International Workshop on Network and Operating Systems Support for Digital Audio and Video, pages:15-20, 2008.

[5] S. Feiner, B. MacIntyre,T. Holerer, A. Webster, *A touring machine: prototyping 3D mobile augmented reality systems for exploring the urban environment*, Personal and Ubiquitous Computing, Volume 1, Issue 4, pages: 208-217, 1997

[6] C. O. Chow et al., *Enhancing real-time video streaming over mobile ad hoc networks using multipoint-to-point communication*,ACM Journal of Computer Communication, Volume 30, Issue 8, pages:1754-1764, 2007

[7] C. Chang, and S. Ger. *Enhancing 3D Graphics on Mobile Devices by Image-Based Rendering,* Proceedings of the Third IEEE Pacific Rim Conference on Multimedia: Advances in Multimedia information Processing, pages: 1105-1111, 2002.

[8] A. Yu et al. *MOPAR: a mobile P2P overlay architecture for interest management of MMOG*, Proceedings of the international workshop on Network and operating systems support for digital audio and video,pages:99-104, 2005.

[9] T. Taleb et al., *Multi-Source Streaming in Next Generation Mobile Communication Systems*, IEEE International Conference on Communications, (ICC '08), pages: 296-300, 2008

[10] S. Shirmohammadi, I. Kazem, D.T. Ahmed, M. El-Badaoui,J.C. De Oliveira,*A visibility driven approach for zone management in simulations*, Simulation, Volume 4, Issue 5, pages: 215-229, 2008.

[11] D.T. Ahmed et al. *Improving Online Gaming Experience Using Location Awareness and Interaction Details*, Journal of Multimedia Tools and Applications, pages: 1-18, 2011

[12] A. Boukerche et al., *An end-to-end virtual environment streaming technique for thin mobile devices over heterogeneous networks*,Computer Communications, Volume 31, Issue 11, pages: 2716-2725, 2008.

[13] A. Boukerche et al. *Scheduling and buffering mechanisms for remote rendering streaming in Virtual walkthrough class of applications*, Proceedings of the 2nd ACM international workshop on Wireless multimedia networking and performance modeling, pages: 53-60, 2006.

[14] A. Boukerche, R. Jarrar, R.W Pazzi, *An efcient protocol for remote virtual environment exploration on wireless mobile devices*, Proceedings of the 4th ACM workshop on Wireless multimedia networking and performance modeling, pages: 45-52, 2008

[15] B. Schmidt-Belz and F Hermann, *User Validation of a Nomadic Exhibition Guide*, Human Computer Interaction with Mobile Devices and Services (Mobile HCI), pages: 86- 97, 2004

[16] A. Boukerche and R. W. Pazzi. *Remote Rendering and Streaming of Progressive Panoramas for Mobile Devices*, In Proceedings of the 14th Annual ACM international Conference on Multimedia, MULTIMEDIA 06, pages: 691-694, 2006.

[17] S.Y. Wu et al. *Headlight prefetching and dynamic chaining for cooperative media streaming in mobile environments*, IEEE Transactions in Mobile Computing, Volume 8, Issue 2, pages: 173-187, 2009.

[18] S.Y. Wu, Cheng-en He, *QoS-aware dynamic adaptation for cooperative media streaming in mobile environments*, IEEE Transactions on Parallel and distributed systems, Issue 99, pages: 1-13, 2010

[19] F. Wu, E. Agu and C. Lindsay, *Adaptive CPU Scheduling to Conserve Energy in Real-Time Mobile Graphics Applications*, Advances in Visual Computing Lecture Notes in Computer Science, Volume 5358, pages: 624-633, 2008

[20] G. Reitmayr et al., *Mobile collaborative augmented reality*, Proceedings of the IEEE and ACM International Symposium on Augmented Reality (ISAR), pages: 114-123, 2001

[21] P. Dahne, J.N Karigiannis,*Archeoguide: System architecture of a mobile outdoor augmented reality system*, Proceedings International Symposium on Mixed and Augmented Reality(ISMAR), pages: 263-264, 2002

[22] T. Hollerer, S. Feiner, T. Terauchi, G. Rashid, D. Hallaway, *Exploring MARS: developing indoor and outdoor user interfaces to a mobile augmented reality system*, Computers and Graphics, Volume 23, Issue 6, pages: 779-785, 1999

[23] R. Cavagna et al., *P2P network for very large virtual environment*, Proceedings of the ACM Symposium on Virtual Reality Software and Technology (VRST), pages 269-276, 2006.

[24] C.H. Chien et al., *Bandwidth aware P2P 3D streaming* , International Journal of Advanced Media and Communication(IJAMC), Volume 4 Issue 4, pages: 324-342, 2010.

[25] Lu Fan, Phil Trinder, and Hamish Taylor, *Design issue for peer-to-peer massively multiplayer online game*, Proceedings of the 2nd IEEE International Workshop on Massively Multiuser Virtual Environments (MMVE 09), pages: 111, March 2009

[26] M. Brachtl, J. Slajs, and P. Slavk, *PDA based navigation system for a 3D environment*, Computers and Graphics, Volume 25, Issue 4, pages: 627-634, August 2001.

[27] J.Chim, R.W.H Lau, H.V. Leong, A. Si,*Cyberwalk: a web based distributed virtual walk through environment*, IEEE Transactions on Multimedia,Volume 5, Issue 4, pages: 503-515, December 2003

[28] D. Ciullo et al., *Network awareness of P2P live streaming applications: A measurement study*, IEEE Transactions in Multimedia, Volume 12, Issue 1, pages: 54-63, 2010

[29] J. Guo,Laxmi Narayan Bhuyan, *Load balancing in a cluster-based web server for multimedia applications*, IEEE Transactions on Parallel and Distributed Systems, Volume 17, Issue 11, pages: 1321-1334, 2006

[30] L. Mcmillan. *An image-based approach to three-dimensional computer graphics,* Technical report, Ph.D. Dissertation, UNC Computer Science TR97-013, 1999.

[31] L. Mcmillan and Gary Bishop, *Plenoptic modeling: an image-based rendering system,* Proceedings of the 22nd annual conference on Computer graphics and interactive techniques (SIGGRAPH '95), pages: 39-46, 1995.

[32] F. Lamberti, C. Zunino, A. Sanna, A. Fiume, and M. Maniezzo, *An accelerated remote graphics architecture for PDAS*, Proceedings of the Eighth international Conference on 3D Web Technology, pages: 55-62, 2003.

[33] J. Royan et al., *Networked based visualization of 3D landscapes and city models*, IEEE Computer Graphics and Applications (CGA), Volume 27, Issue 6 pages: 70-79, 2007.

[34] G. Thomas, G. Point, K. Bouatouch. *A client-server approach to image-based rendering on mobile terminals*. Technical Report, ISSN 0249-6399, January 2005.

[35] A. Doran, S. Mondet, R. Grigoras, G. Morin, W.T. Ooi, F. Boudon,*A demonstration of MobiTree: Progressive 3D tree models streaming on mobile clients*, Proceedings of the seventeen ACM international conference on Multimedia, pages: 955-956, 2009

[36] J. Peltotalo et al., *A Real-Time Peer-to-Peer Streaming System for Mobile Networking Environment*, IEEE INFOCOM workshops, pages: 1-7, 2009.

[37] Google Street View, *https://developers.google.com/maps/documentation/javascript/streetview*, Last accessed in May 2012

[38] QuickTime VR (QTVR), *http://www.quicktimevirtualreality.com*, Last accessed in May 2012

[39] OpenGL, *www.opengl.org*, Last accessed in May 2012

[40] D.M. Chen et al. *Streaming mobile augmented reality on mobile phones*, The 8th IEEE International Symposium on Mixed and Augmented Reality, pages 181-182, 2009

[41] J. Krosche, J. Baldzer, S. Boll, *MobiDENK: mobile multimedia in monument conservation*, IEEE MultiMedia , Volume 11, Issue 2, pages: 72-77, 2004

[42] D. Wagner et al., *First steps towards handheld augmented reality*, Proceedings of the 7th IEEE International Symposium on Wearable Computers, pages: 127-135, 2003

[43] J. Newman, D. Ingram, A. Hopper*Augmented reality in a wide area sentient environment*, Proceedings of the IEEE and ACM International Symposium on Augmented Reality (ISAR'01), pages: 77-86, 2001

[44] E.J. Berglund, D.R. Cheriton, *Amaze: a multiplayer computer game*, IEEE Software, Volume 2, Issue 3, pages: 30-39, May 1985

[45] J. Gausemeier et al. *Development of a real time image based object recognition method for mobile AR-devices*, Proceedings of the 2nd international conference on Computer graphics, virtual Reality, visualization and interaction in Africa, pages: 133-139, 2003

[46] H. Hoppe. *Progressive Meshes*, Proceedings of the 23rd annual conference on Computer graphics and interactive techniques (SIGGRAPH'96) , pages: 99-108, 1996.

[47] Fan Bai, *Mobility Modeling and Analysis in Wireless Ad Hoc and Sensor Networks*,PhD Dissertation, 2005

[48] T. Camp et al., *A survey of mobility models for Ad Hoc network research*, Wireless communications and mobile computing: special issue on mobile AdHoc Networking: Research, trends and applications, Volume 2, Issue 5, pages: 483-502, 2002

[49] S. Kircher, and M. Garland. *Progressive multiresolution meshes for deforming surfaces*, In Proceedings of the 2005 ACM Siggraph/Eurographics Symposium on Computer Animation, pages: 191-200, 2005

[50] T. Akenine-Moller, T. Moller, E. Haines, *Real-Time Rendering*, A. K. Peters, Ltd., Natick, MA, USA, 2002.

[51] M. Lindeberg et. al, *Challenges and techniques for video streaming over mobile ad hoc networks*, Multimedia Systems, volume 17, pages: 51-82, 2011.

[52] J. Peng, and C. J. Kuo. *Geometry-guided progressive lossless 3D mesh coding with octree (OT) decomposition*, ACM Transactions on Graphics (TOG) - Proceedings of ACM SIGGRAPH, Volume 24, Issue 3, pages: 609-616, July 2005

[53] S. Yang, C. Lee, and C. J. Kuo. *Optimized mesh and texture multiplexing for progressive textured model transmission*, Proceedings of the 12th Annual ACM international Conference on Multimedia, pages: 676-683, 2004

[54] Yang, W. Wu, K. Nahrstedt, G. Kurillo, R. Bajcsy, *ViewCast: View dissemination and management for Multi-party 3D tele-immersive environments*, Proceedings of the 15th international conference on Multimedia, pages: 882-891, 2007

[55] M. Isenburg, P. Lindstrom, *Streaming meshes*, IEEE Visualization, pages: 231-238, November 2005.

[56] J. El-Sana and A.Varshney. *Generalized view-dependent simplification*, Proceedings of EURO GRAPHICS 99, pages: 83-94, 1999.

[57] J. El-Sana and Y. J. Chiang. *External Memory View-Dependent Simplification*, Proceedings of EUROGRAPHICS00, pages: 139-150, 2000

[58] HP iPAQ 210 Enterprise Handheld [http://h10010.www1.hp.com/wwpc/ca/en/sm/WF05a/21534
215348-64929-215384-215384-3544499.html], Last accessed October 2011

[59] H. P. Shiang et al., *Informationally Decentralized Video Streaming Over Multihop Wireless Networks.*, IEEE Transactions on Multimedia, Volume 9, Issue 6, pages 1299 - 1313 , 2007

[60] J. Lluch, R. Gaitn, E. Camahort, and R. Viv. *Interactive three-dimensional rendering on mobile computer devices*, Proceedings of the 2005 ACM SIGCHI international Conference on Advances in Computer Entertainment Technology, pages: 254-257, 2005

[61] T. Fang, Z. Tian,*Efficient storage and progressive rendering of multi-resolution mesh*, Advances in Multimedia Information Processing (PCM), Volume 48, pages: 814-821, 2007

[62] J.S. Boulanger, J. Kienzle, C.Verbrugge, *Comparing Interest management algorithms for massively multiplayer games*, Proceedings of 5th ACM SIGCOMM workshop on Network and system support for games, pages: 1-12, 2006.

[63] R. Pajarola and J. Rossignac, *Compressed Progressive Meshes*, IEEE Transactions on Visualization and Computer Graphics, Volume 6, Issue 1, pages: 79-93, 2000

[64] Khronos Group, *OpenGl ES- The standard for Embedded Accelerated 3D Graphics* http://www.khronos.org/opengles/. 2004. Last accessed October 2011

[65] S. Han, M. Lim, D. Lee, S.J. Hyun, *A scalable interest management scheme for distributed virtual environments*, Computer Animation and Virtual Worlds, Volume 19, Issue 2, pages: 129-149, 2008

[66] M. R.Macedonia, M. J. Zyda, D. R. Pratt, P. T. Barham, and S. Zeswitz. *NPSNET: A Network Software Architecture for Large- Scale Virtual Environment.* Presence, Vol.3(4), pp. 265-287, 1994

[67] H. Li, B. Prabhakaran, *Smart Decision Module for Streaming 3D Meshes over Lossy Networks*, Proceedings of the International Conference on Distributed Multimedia Systems and International Workshop on Visual Languages and Computing (DMS/VLC'04), pages: 275-278, 2004.

[68] F. Li, R. Lau, D. Kilis, L. Li, *Game-on-demand: An Online Game Engine Based on Geometry Streaming*, ACM Transactions on Multimedia Computing, Communications, and Applications (TOMCCAP), Volume 7, Issue 3, 2011

[69] S. Dobbyn, J. Hamill, K. OConor, and C. OSullivan. *Geopostors: a real-time geometry impostor crowd rendering system*, Proceedings of the 2005 Symposium on interactive 3D Graphics and Games (SI3D 05), pages: 95-102, April 2005.

[70] S. Jeschke, M. Wimmer, H. Schumann, and W. Purgathofer. *Automatic impostor placement for guaranteed frame rates and low memory requirements*, Proceedings of the Symposium on interactive 3D Graphics and Games (SI3D 05), pages: 103-110, April 2005.

[71] Frey, Royan, Piegay, Kermarrec, Anceaume, Le Fessant, *Solipsis: A Decentralized Architecture for Virtual Environments*, IEEE international Workshop on Massively Multiuser Virtual Environments (MMVE'08), pages: 29-33, 2008

[72] D. Gotz, *Scalable and adaptive streaming for non linear media*, Proceedings of the 14th annual ACM international conference on Multimedia, pages: 357-366, 2006

[73] T. Richardson, Q. Stafford-Fraser, K. R. Wood, and A. Hopper. *Virtual Network Computing*. IEEE Internet Computing, Volume 2, Issue 1, pages: 33-38, 1998.

[74] Greg Humphreys, Mike Houston, Ren Ng, Sean Ahern, Randall Frank, Peter Kirchner, and James T. Klosowski. *Chromium: A Stream Processing Framework for Interactive Graphics on Clusters of Workstations.* ACM Transactions on Graphics, Volume 21, Issue 3, pages: 693-702, July 2002.

[75] H. Xu et al., *Improving QoS in peer-to-peer streaming media system*, Journal of Computational Information Systems, Volume 6, Issue 5, pages: 1387-1395, 2010.

[76] NS2 -The Network Simulator, *http://www.isi.edu/nsnam/ns/*, Last accessed in May 2012

[77] Field of View, *http://www.guidinglight.com/field-of-view/encyclopedia.htm*, Last accessed in May 2012

[78] S. Chen and L.Williams. *View interpolation for image synthesis*, Proceedings of the 20th annual conference on Computer graphics and interactive techniques (SIGGTAPH'93), pages: 279-288, 1993.

[79] W. Cheng, D. Liu, W.T Ooi ,*Peer assisted view dependent progressive mesh streaming*, Proceedings of the seventeen ACM international conference on Multimedia, pages: 441-450, 2009

[80] J. Diepstraten and T. Ertl, *Remote Line Rendering for Mobile Devices*, Proceedings of International Computer Graphics, pages: 454-461, 2004.

[81] Y. Liu et al., *A survey on peer-topeer video streaming systems*, Peer-to-Peer Networking and Applications, Volume 1, Issue 1,pages: 18-28, 2008.

[82] K. Graffi, Sebastian Kaune, Konstantin Pussep, Aleksandra Kovacevic, Ralf Steinmetz,*Load balancing for multimedia streaming in heterogeneous Peer-to-Peer systems*, Proceedings of the 18th International Workshop on Network and Operating Systems Support for Digital Audio and Video, pages: 99-104, 2008

[83] C. Greenhalgh and S. Benford. *Boundaries, awareness, and interaction in collaborative virtual environments*, Proceedings of the Sixth IEEE Workshop on Enabling Technologies (WETICE), pages: 193-198, 1997.

[84] J.C. Xia, J. El-Sana, A. Varshney,*Adaptive real time level of detail based rendering for polygonal models*, IEEE Transactions on Visualization and Computer graphics, Volume 3, Issue 2, pages: 171-183, 1997.

[85] I. Yoon and U. Neumann, *Web-based remote rendering with IBRAC (image-based rendering acceleration and compression)*, Computer Graphics, Volume 19, Issue 3, pages: 321-330, 2000.

[86] B. Seo, R. Zimmermann, *Quantitative Analysis of Visibility Determinations for Networked Virtual Environments*, Journal of Visual Communication and Image Representation, 2012.

[87] B. Seo, R. Zimmermann, *Edge indexing in a grid for highly dynamic virtual environments*, Proceedings of the 14thannual ACM international conference on Multimedia (MULTIMEDIA 06), pages 402411, 2006

[88] J. Botev et al.,*The HyperVerse Concepts for a Federated and Torrent Based 3D Web*, Intl J. Advanced Mediaand Communication, vol. 2, no. 4, 2008

[89] M.Zhu et.al, *Towards peer-assisted rendering in networked virtual environments*, Proceedings of the 19th ACM international conference on Multimedia, 2011.

[90] H. Rahimi et al., *Context-Aware Prioritized Game Streaming*, in Proc. of Workshop on Interactive Ambient Intelligence Multimedia Environments, in Proceedings of IEEE International Conference on Multimedia and Expo (ICME 2011), 2011.

[91] R. Jarrar, *Image-based Rendering Protocols for Remote Interactive Walkthroughs on Mobile Devices*, Matser Thesis, 2009.

[92] J.H. Clark, *Hierarchical Geometric Models for Visible Surface Algorithms*, Communications of the ACM, Volume 19, Issue 10, pages: 547-554, 1976.

[93] P. Moravek et al., *Signal Propagation and Distance Estimation in Wireless Sensor Networks*, The 33rd International Conference on Telecommunications and Signal Processing (TSP), pages 189-194, 2010