A Location Aware P2P Voice Communication Protocol for Networked Virtual Environments

Gabor Papp
University of Denver
pgabor@cs.du.edu

Chris GauthierDickey
University of Denver
chrisg@cs.du.edu

Abstract

Multiparty voice communication, where multiple people can communicate in a group, is an important component of networked virtual environments (NVEs), especially in many types of online games. In this paper, we present a new peer-to-peer protocol that uses Gabriel graphs, a subgraph of Delaunay triangulations, to provide scalable multiparty voice communication. In addition, our protocol uses positional information so that voice data can be accurately modeled to listeners to increase the immersiveness of their experience. Our simulations show that the algorithms scale well even in densely populated areas, while prioritizing the sending of voice packets to the closest listeners of a speaker first, thus behaving as users expect.

CR Categories: C.2.4 [Computer Systems Organization]: Computer-Communication Networks—Distributed Systems H.5.1 [Information Systems]: Information Interfaces and Presentation—Multimedia Information Systems

Keywords: P2P, Voice Communication, Location Awareness, Virtual Environment, Delaunay Triangulation, Gabriel Graph

1 Introduction

With the increase in end-user Internet bandwidth, the number of virtual environments and multiplayer games providing multiparty voice communication (MVC) for player interaction has increased significantly. For example, World of Warcraft has millions of subscribers and provides MVC for its players [World of Warcraft]. Every major game console uses MVC as a selling feature. Further, MVC is used for conference calls, voice-chat software, and is often included in distributed collaborative systems.

To date, most multiparty voice communication software uses a client/server architecture. This architecture is useful because it provides a centralized point for authentication, administration, and security. On the other hand, it requires a large amount of bandwidth to host [Papp and GauthierDickey 2008], it is a single-point of failure, and it requires significant configuration by end-users if they are required to download and host a voice server for games they are playing.

In this paper, we present a peer-to-peer architecture for multiparty voice communication that scales well with the number of participants (which we will call *avatars*) and uses information from the virtual environment to determine how to connect nodes. This allows our protocol to send voice packets first to those who are closest to us in the virtual environment and to only cluster avatars who are in each others area of interest (AOI). In addition, the peer-to-peer MVC can be used in conjunction with both distributed and centralized networked virtual environments, with the advantage that in a client/server architecture, voice traffic can be off-loaded to the clients.

Unlike most MVC software, our protocol uses the virtual locations of avatars to help form its distribution graph. This allows positional audio to be modeled accurately to the listeners and provides an increased level of immersion. Our protocol allows anyone in the virtual world to talk to anyone else as long as their AOIs intersect. This differs from current games which limit talking to a special group, such as a party or team, or requiring them to log onto voice servers and all join the same channel. We note that clearly separate channels are trivially supported in our protocol and the NVE interface can easily allow voice messages to be blocked. On the other hand, because our protocols can determine who should hear a voice packet, more realistic environments can be created and virtual meeting areas can be more accurately modeled.

Our protocol relies on *Gabriel* graphs, which are subgraphs of a Delaunay triangulation of the entire graph, but can be calculated locally [Matula and Sokal 1980]. Gabriel graphs have the important property that any two closest neighbors are guaranteed to be connected–thus voice packets which should go to neighbors will be sent to the closest neighbors first and then possibly relayed to further nodes in the network. This reduces latency between neighbors since they exchange packets with each other in a single overlay hop. It further limits the average number of neighbors any single node has, which limits the required bandwidth by nodes.

Our results demonstrate that our protocol works well even in densely populated areas, which we simulate by increasing the AOIs of each avatar to cover proportionally larger areas of the virtual space. With 1024 avatars all within hearing range of each other, we measured an average of 17 hops between a speaker and its listeners, each with an average of 4 directly connected neighbors. To put this value in context, we note that this density of avatars is approximately an order of magnitude larger than the most densely populated area in World of Warcraft [Pittman and GauthierDickey 2007].

The main contribution of this work is a location aware peer-to-peer multiparty voice communication protocol. By considering locations in the virtual world, avatars are organized such that closer neighbors receive voice packets first before they are sent to farther neighbors. Thus, our protocol will allow virtual environments to both scale, due to the distributed nature of the protocol, and to model the virtual environment in a manner that users expect.

2 Background

Research related to our work on multiparty voice communication includes measurement studies about voice patterns and codecs and research on architectures for efficient multiparty voice communications, including cost-effective routing algorithms. While much research has been produced over the last several years regarding peer-to-peer networks, most of it has been focused around distributed

---

VRST 2009, Kyoto, Japan, November 18 – 20, 2009.

hash tables which map a *key* to a *value* over a P2P network. NVEs differ from typical P2P applications because peers have coordinates in a 2 or 3 dimensional virtual space. As such, we can take advantage of this additional information to efficiently route between nodes.

Voronoi diagrams, Delaunay triangulations and Gabriel graphs has been a topic of research in graph theory and communications for many years. For example, Lee and Lam describe a protocol for calculating Delaunay triangulations in a dynamic network [Lee and Lam 2007] while Matula and Sokal explore the properties of Gabriel graphs [Matula and Sokal 1980]. We take these graph theory ideas and apply them to the domain of multiparty voice communication.

Multiparty voice communication is one piece of a larger spatial model for interaction between groups of people in a virtual world, as described by Benford et al [Benford et al. 1994]. In this model, awareness, auras, focus, and nimbus are defined to describe the way objects and actors interact. The concept of an aura and focus are similar to the concept of an area of interest used to define who is within hearing range in our work.

Dowlatshahi and Safaei developed a multiparty voice communication algorithm for P2P networks where streams are selectively mixed acording to upstream and downstream requirements, thereby reducing bandwidth requirements between peers [Dowlatshahi and Safaei 2006]. However, a minimum spanning tree is constructed based purely on delay constraints and *coordinates* in the physical network–thus, nodes may be far apart in the virtual space, but have to forward packets for which they have no interest in. Elleuch and Houle also use a tree-based media distribution network where mixer nodes and leaf nodes are distinguished [Elleuch and Houle 2008]. This setup allows resource-constrained devices to act only as receivers, while more powerful devices can act as senders, receivers and mixers. Unique to their design, they separate the control network, which administers memberships to various voice conferences, and the media flow network, which is solely for disseminating and mixing voice packets.

Wu and Li use a combination of rateless codes for voice packets and an algorithm to find an optimal set of peers for packet distribution to minimize peer-to-peer delay and to combat fluctuations in network conditions [Wu and Li 2005]. While the rateless codes allow a peer to reconstruct voice packets even when a portion of them may not have arrived, the optimal peer organization depends on knowing the delay and bandwidth of the peer-to-peer links, which may limit its effectiveness as accurately calculating delay and bandwidth between peers is a difficult problem. As a result, these values need to be estimated and result in a sub-optimal solution. Note that similar to Elluech and Houle's work [Elleuch and Houle 2008], the tree is organized around delay between peers and not distance in the virtual world.

Jiang and Chen propose an AOI-based voice chatting protocol for MMOGs [Jiang and Chen 2007]. They claim that their approach improves the way players communicate one on another and provides a more realistic virtual environment. Our approach has similar goals, however we do not just take in account the distance between the initiator and receivers but we also account for their relative locations, resulting in shorter paths to all neighbors of a speaker. Moreover, our solution does not simply assumes that the location information is available for any node, but is built on the delaunay triangulation, which is a proven method for neighbor maintenance.

Perhaps the closest work to ours is a geo-routing algorithm in planar graphs for ad-hoc wireless networks developed by Muhammad [Bin Muhammad 2007]. The geo-routing algorithm is based on Gabriel graphs that are constructed using the coordinate of the nodes under the assumption of a static graph. We also use Gabriel graphs, but our solution works for highly dynamic virtual environments and additionally uses Delaunay triangulations for neighbor maintenance.

3 P2P Voice Communication

Many types of networked applications fall in the realm of networked virtual environments, though collaborative virtual environments and multiplayer games tend to make up the bulk of the applications. In these systems, participants are represented in the virtual world as *avatars*, which can interact with the environment and with each other. Because a participants' hands may be occupied controlling movement and interaction with the virtual world, voice communication is both a desirable and convenient method of communication between avatars.

Our multiparty voice communication protocol is suitable for all types of networked virtual environments, but its strength lies in being decentralized and in combining meta-information with the voice packets. In particular, our protocol sends the position and orientation of speakers and listeners, while maintaining low bandwidth and delay. In NVEs such as multiplayer games, the voice communication protocol has both delay and bandwidth requirements. High delay affects the interactivity of the protocol and speakers are more likely begin talking at the same time due to hearing silence on their end (as voice packets from someone already talking have yet to arrive). Particularly with games, which may have hundreds or thousands of players, bandwidth of the voice communication protocol competes with the bandwidth required by the game.

In our protocol, we assume that each node has a position in a 2 dimensional space (though we can extend this to 3 dimensions) and an *Area of Interest*, or AOI, that indicates the farthest distance centered at the avatar's position that voice can be heard from. The protocol works by computing a Gabriel graph (described in Section 3.1) for nodes in the system in a completely distributed fashion. When an avatar talks, the voice packets are sent via an AOI limited broadcast from the talking node to its neighbors. Neighbors continue to forward the messages as long as their neighbors fall within the AOI of the talking node. Unlike previous protocols where two neighbors in the graph may be connected because they are close by a metric such as delay, messages in our protocol only travel to nodes that are possibly interested in them (i.e., they are within the AOI of the sender). This reduces overall traffic and prevents nodes from acting purely as relays.

To handle joining and leaving the network and to assist in calculating the Gabriel graph, nodes also maintain a Delaunay triangulation (which can be done in a distributed fashion [Lee and Lam 2007]). Note that if we combine our protocol with a client/server based virtual environment, maintaining the Delaunay triangulation is no longer necessary because the server can calculate neighbor sets on the server and inform each avatar which Delaunay neighbors they have. In this setup, peers would then perform a distributed calculation of the Gabriel graph and communicate between themselves without needing to further involve the server.

Throughout the paper, we use the following notation:

- n : the number of nodes in the network
- $v_0, \ldots, v_{n-1}$: the nodes in the network (or vertices of the constructed graph)
- $\overline{v_i v_j}$: an edge between v_i and v_j

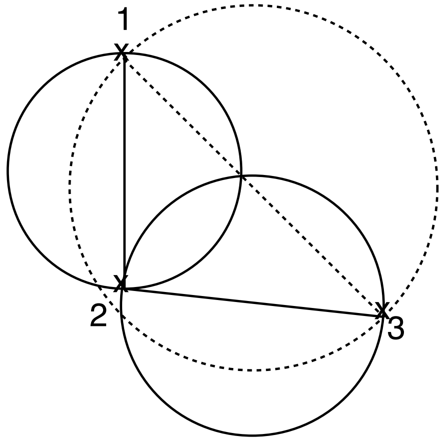


Figure 1: *Gabriel graph of three nodes in the plane. The graph is computed by adding an edge between two vertices if a disc which uses the edge as its diameter does not contain any other vertices, thus an edge is set between vertices 1, 2 and 2, 3. However, an edge is not set between vertices 1 and 3 because the disc formed by that edge contains vertex 2.*

- $AOI(v_i)$: indicating the Area of Interest of node v_i. The $AOI(v_i)$ a scalar value that indicates the radius of a circle centered at the position of v_i.

3.1 The Gabriel Graph and Its Properties

A Gabriel graph is a type of graph that connects a set of vertices in the Euclidean plane under the following rule: two vertices v_i and v_j are connected by an edge whenever the disc with the line segment $\overline{v_i v_j}$ as its diameter contains no other points from the given point set. Figure 1 illustrates the Gabriel graph of three nodes. Both $\overline{v_1 v_2}$ and $\overline{v_2 v_3}$ are edges of the graph. However, $\overline{v_1 v_3}$ is not an edge because the disc encircling this edge contains the vertex v_2.

Gabriel graphs are related to Delaunay triangulations in that a Gabriel graph is completely contained within a Delaunay triangulation and can be derived from it in $O(n)$ steps, where n is the number of vertices in the Delaunay triangulation.

In addition, Gabriel graphs contain both the Euclidean minimum spanning tree (MST) and the nearest neighbor graph. The MST ensures that the fewest edges are used when broadcasting from a speaking node to its listeners. The nearest neighbor graph is a graph such that for any pair of vertices, (v_i, v_j), a directed edge exists between v_i and v_j if and only if v_j is closer to v_i than any other vertex.

We have chosen to use Gabriel graphs because these properties give us a close approximation to real voice communication:

1. It always contains the nearest neighbor, which means any two avatars that are inside of each others' hearing range and are the closest to each other will always be directly connected (note that the Gabriel graph may contain cycles).
2. It also contains the minimum spanning tree which ensures that voice packets take as few additional paths as possible, reducing the overall traffic on the network.
3. Being a subgraph of the Delaunay triangulation means that the minimum angle between edges is maximized. Avoiding narrow triangles allows one to create a more realistic simulation since the human voice spreads at a wide angle naturally.

Note that with a Delaunay triangulation, each vertex has on average six neighbors. For a Gabriel graph, the average number of neighbors is 4. This means that we have to replicate packets on average fewer times than with a Delaunay triangulation. The tradeoff is that with fewer edges, the diameter of the graph will be larger—but this should only be a factor in very dense graphs where we must reach a large number of listeners within an AOI. We believe this is an important tradeoff because bandwidth becomes an issue as we increase the quality of the voice packets (and therefore their sizes). In fact, listeners at the end of a long path will necessarily have many other listeners *in front* of them in the virtual space, or they would have had a shorter path since the graph is based on positions in the virtual world. Thus, they would realistically find it difficult to hear someone speaking in a crowd of people. With Gabriel graphs, these packets would be more delayed, but could also be dampened to simulate crowd effects.

3.2 Greedy Routing on the Gabriel Graph

We now show that we can route a message to the closest peer to a location in the network using a greedy algorithm. Assume we have nodes in a 2D plane. All nodes have a pair of coordinates defining their positions. Define $M_{(x,y)}$ as the message being routed to location (x, y). Let $N(v_i)$ be the set of neighbors in the Gabriel graph of node v_i (recall that a neighbor is a node with an edge from itself to v_i in this case).

Theorem 3.1 *Routing a message $M_{(x,y)}$ over a connected Gabriel graph using the following greedy algorithm will always find a path to the node closest to (x, y). This greedy algorithm is defined as: When a node v_i receives the message from v_j, forward the message to the node from the set $N(v_i)\backslash v_j$ which has the closest Euclidean distance to (x, y).*

Proof. Assume the Gabriel graph is connected. Because the graph is connected a path must exist between any two nodes. We hypothesize that we can find a path from v_i to v_j by greedily choosing the neighbor v_k of v_i who is closest to v_j.

Construct a disc such that v_i and v_j lie in the disc and its diameter d is the distance between v_i and v_j. Let D_{ij} be the set of vertices that lie in the disc. We have to distinguish between two cases:

1. D_{ij} is the empty set: a direct edge exists between v_i and v_j. $\Rightarrow$ We traverse through the edge and reach our destination.
2. D_{ij} is not empty: there is no direct edge between v_i and v_j. $\Rightarrow$ Choose node $v_k \in D_{ij}$ such that $v_k \in N(v_i)$ and $|\overline{v_k v_j}|$ is minimal. Since $v_k \in D_{ij}$, the next statement also holds: $|\overline{v_k v_j}| < |\overline{v_i v_j}|$. Next, repeat the algorithm with v_k and v_j.

As we have a finite number of nodes, we get closer to the destination with every step, and eventually we get to the destination itself. ☐

3.2.1 Neighbor Sets

To maintain the necessary graphs in our protocol, each node maintains two neighbor sets:

- $D(v_i)$: the set of nodes in the network that are Delaunay neighbors, which we call the *Delaunay set*.
- $I(v_i)$: the set of nodes that are Gabriel neighbors of v_i from the Gabriel graph constructed using $D(v_i)$ and $AOI(v_i)$, which we call the *Interest set*.

The Delaunay set is used to maintain the Gabriel neighbors and handle joining and leaving. Note that the Delaunay set can be maintained via distributed Delaunay triangulation protocols [Lee and

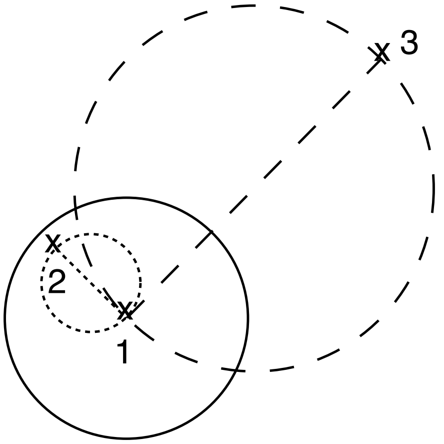


Figure 2: *AOI-limited broadcast: In the full Gabriel graph of the network, an edge from vertex 1 to vertex 3 exists. However, because each node has an AOI, this edge can be ignored since the node at vertex 3 would be unable to hear anything said by vertex 1.*

Lam 2007]. The Gabriel neighbors can be then calculcated from the Delaunay set. Each node looks at its set of Delaunay neighbors and applies the Gabriel graph algorithm to them (decribed in Section 3.1), adding those nodes with an edge in the resulting Gabriel graph to the Interest set.

Note that for the voice protocol, we are required to know which nodes are inside the AOI of a given node because we only need to route voice packets to nodes within the AOI. One concern is that with a Gabriel graph, a path may exit the AOI of a given node only to re-enter at a later point. For example, assume that we have a similar layout to Figure 2. The solid line represents $AOI(v_1)$, and the dashed lines represent the diameters and the corresponding discs. Although the disc with diameter $\overline{v_1v_3}$ does not contain a third node, v_3 is not in $I(v_1)$ because v_3 is outside of $AOI(v_1)$. This illustrates that any node outside of the $AOI(v_1)$ will never have a voice packet routed to it directly from v_1, even though it may have an edge in the corresponding Gabriel graph.

On the other hand, $\overline{v_1v_2}$ is a valid edge only when its corresponding disc does not contain another node. Any point that is inside $AOI(v_1)$ defines a disc that is also completely inside $AOI(v_1)$. As such, we only have to check the validity of edges as possible broadcast paths for the nodes in $D(v_1)$. Since these nodes are maintained by v_1 via Delaunay triangulation algorithms, routing over the Gabriel graph from v_1 only requires knowing a nodes AOI and broadcasting only to those neighbors who fall within the AOI of v_1.

3.2.2 The Voice Packet Graph and Protocol

The voice packet graph is a subgraph of the Gabriel graph of the whole network. It contains only those edges that are not longer than the radius of the AOI and may therefore be disjoint. Given the definition of a Gabriel graph and given all the nodes in the network, a connected graph would be generated. However, with our protocol, we throw out edges between nodes whose are not inside of each other's AOI since an avatar cannot hear beyond its AOI.

The transmission of the voice packets is done only along the edges of the voice packet graph. Every node that generates a voice packet attaches its coordinates and orientation to the outgoing voice packets and then sends this packet out to all the nodes that are one hop from itself. These nodes then check the coordinates of the sender node and decide which of their neighbors they have to forward the packet to. Since the voice packet graph contains all the nodes that are in each other's hearing range, delivery is guaranteed to all neighbors who may need to receive the packet. Nodes outside of the AOIs of the senders will not be part of the same partition in the voice packet graph. Packets which need to be relayed arrive later than those which are only a single hop away, ensuring that avatars closest together receive communication between each other first.

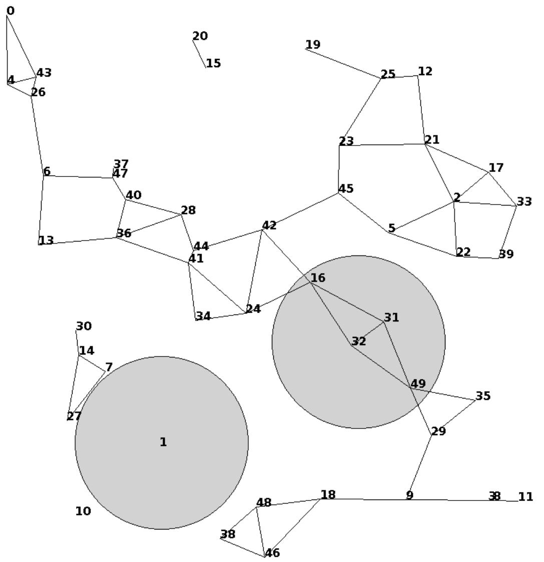


Figure 3: *An example of Voice Packet Graph constructed by our protocol. Note that some nodes are fully disconnected from the graph because their AOIs do not intersect with any other nodes. Nodes that are closest always have an edge between them. Further, some partitions in the graph have large diameters, but because voice packets include positional information, they are only forwarded to nodes within the AOI of the originating speaker. For better understanding we present the discs of node 1 and 32.*

Figure 3 shows a snapshot of the voice communication graph with 50 nodes and 0.15 radius. From this Figure, we can see that nodes close to each other are connected but the entire graph is not necessarily connected. On the other hand, long chains of nodes can be seen (e.g., a path from node 0 to node 11), but because voice packets include positional information, they are only forwarded to neighbors within the radius of the original speaker.

3.3 Building and Maintaining the Delaunay Triangulation

Every node in the network maintains its Delaunay neighbors. Using this set the nodes can calculate which other nodes are inside their AOI and which of these nodes are Gabriel neighbors. These Gabriel neighbors are then used to forward the voice packets in the network. We refer to this network as the *control network*.

3.3.1 The Delaunay Triangulation

A Delaunay triangulation is a triangulation of a set of points in a graph such that no point is inside the circumcircle of any triangle formed by any nodes in the network and can be computed in $O(n \log n)$ time for a 2D space. The Delaunay triangulation has been widely used to keep track of nodes in a network in a distributed manner. Although there are several ways calculate the Delaunay triangulation, such as the flip, the incremental, or the sweepline algorithms, none of these are distributed.

Lee and Lam focus on the design of the *join*, *leave*, and *maintenance* protocols to construct and maintain a distributed Delaunay triangulation dynamically [Lee and Lam 2007]. The *join* protocol assumes the knowledge of at least one node in the system so that it may *bootstrap* into the system. This node is then able to route the *joining* node to its closest neighbor using and appropriate routing algorithm. Next, a complete neighbor list exchange is performed recursively until no new neighbor is found. The *leave* protocol is not necessary because the *maintenance* protocol itself is sufficient enough to keep the system in a consistent state, but it can speed up this process. These protocols together are able to provide an underlying layer that keeps track of all the Delaunay neighbors of all the nodes in the system in a distributed way.

Bose and Morin investigate the different kinds of routing algorithms for triangulations [Bose and Morin 1999]. They present a greedy routing algorithm, which simply forwards the packets from a node to the neighbor which is the closest to the destination. This algorithm always guarantees the delivery of a packet inside a Delaunay graph along some path to its destination. They also present the compass and randomized compass algorithms, which require less steps on average, but still has an O(n) worst case performance. To eliminate this issue, two more sophisticated algorithms are presented and described in their work.

The above mentioned methods are sufficient enough to build and maintain a Delaunay triangulation even for highly dynamic networks, such as peer-to-peer online games. Thus, we assume that such an underlying network exists that we can use for neighbor maintenance, and later for the Gabriel graph construction.

4 Simulation Results

In this section we evaluate our protocol by simulation. Initially the nodes in our network are distributed in a 1×1 square. We run each of the simulations for 60 seconds based on two different mobility models. In our simulations we sample the network in every 100 milliseconds, for a total of 600 times during the simulation, which is representative of the rate of voice packets typical in voice communication protocols [Papp and GauthierDickey 2008]. The radius of the AOI ranges from .1 to 1.6 in 6 steps, giving an effective diameter of up to 3.2, which is equivalent to not having any limit on the range of hearing in the virtual world. We simulate from 4 to 1024 total nodes. In our results, we present the average of all the samples collected, and detailed results for the most typical cases.

4.1 Mobility Models

During the simulations, the nodes move inside the square based on the random waypoint mobility model. The speed of the nodes is simulated with the Normal distribution, where the mean of the random variable is 1 and so is the standard deviation.

The destination of the nodes is chosen based on two different distributions:

1. Uniform: The x and y coordinates are chosen uniformly and independently from each other. This simulates the traditional random waypoint mobility model.

2. Exponential: The x and y coordinates are chosen together using a three-dimensional distribution that is a superposition of two exponential distributions (Figure 4). This models a virtual reality where the nodes tend to gather around two hotspots. The secondary hotspot has a peak that is 75% of the primary hotspot. This way the nodes still move from one peak to the other, but the nodes are closer to each other and therefore form a cluster. Note that exponential distributions of players have been measured in large-scale, multiplayer games [Pittman and GauthierDickey 2007], leading us to this mobility model.

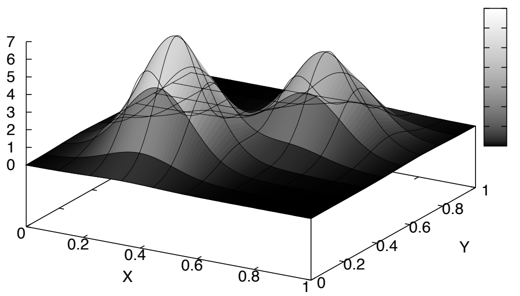


Figure 4: *Three-dimensional distribution to choose destination: Having two hotspots as destination for the mobility model allows us to investigate the effect of clusters.*

When a node reaches its destination, a new location and speed is calculated based on the distributions used.

4.2 Theoretical Boundary

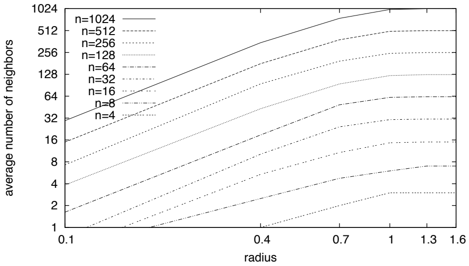
The performance of our protocol depends in part on the number of nodes that are inside of the AOI of a given node since this determines the minimum bandwidth used for voice transmission. Figure 5 shows the average number of nodes that are inside of the AOI of a given node. We ran multiple simulations, and we varied both the number of nodes participating in the network and the radius of the AOI. As expected, the average number of nodes within the AOI is proportional to the radius of the AOI. Additionally, when the exponential mobility model is used, we see a twofold increase in the number of peers within the AOI.

4.3 Load Balance and Scalability

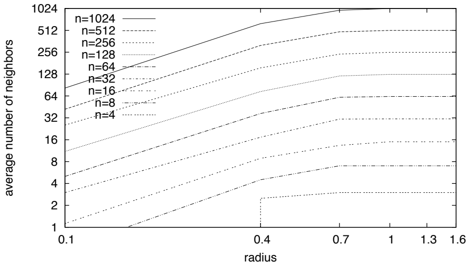
In these experiments, we measured load balancing and scalability of our protocol. Our metric for load balancing is the average degree of a node. While one possibility is to simply maintain the k-closest neighbors to communicate with, this could result in disjoint graphs. While Delaunay and Gabriel graphs will always be connected, they do not guarantee a low node degree and we therefore ran simulations to determine if these graphs have similar properties.

Our results show that both the Delaunay and the Gabriel graphs maintain a low node degree on average with a low standard deviation. Figure 6 shows that the maximum average degree of a node in the Delaunay graph is six, which is in accordance to the theoretical average. Thus, the Delaunay triangulation not only guarantees that all of the nodes are connected and therefore any avatar is reachable by any other avatar but it also creates a graph where the average node degree is low and therefore has a low bandwidth requirement for neighbor maintenance.

Our Gabriel graph protocol shows similar patterns (see Figure 7). The results show that the average number of Gabriel neighbors, ranges from 0 to 4. Note that even at its most loaded setup, where

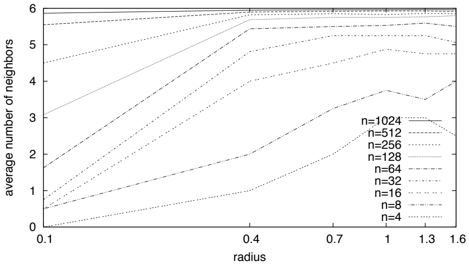


(a) Uniform distribution

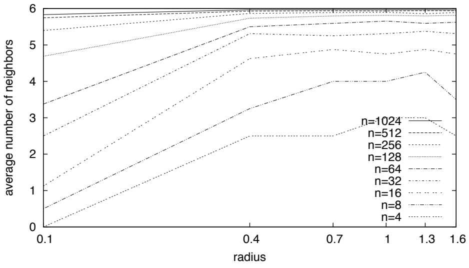


(b) Exponential distribution

Figure 5: *Average number of nodes within the AOI: The number of nodes within an AOI indicates how many peers a protocol will have to handle effectively on average. As the figure shows, the number of nodes inside the AOI increases proportionally with its radius. Note that the number of nodes in the network with the exponential distribution has almost twice as many neighbors, on average, as the uniform distribution.*

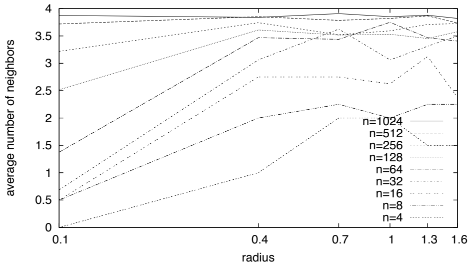


(a) Uniform distribution

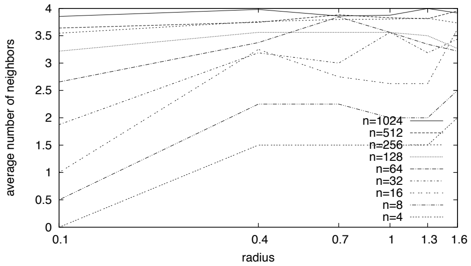


(b) Exponential distribution

Figure 6: *Average node degree for the Delaunay triangulation: as the radius of the AOI and node density increases, the average number of neighbors approaches the theoretical average of 6 neighbors per node. The average neighbor count is a measure of how many times a packet would need to be replicated to reach its listeners. Both exponential and uniform distributions showed similar results, indicating the efficacy of using Delaunay triangulations.*



(a) Uniform distribution



(b) Exponential distribution

Figure 7: *Average node degree for the Gabriel graph: as the radius of the AOI and the node density increases, the bandwidth requirements for our protocol increases. Our results show that the average neighbor count approaches its theoretical maximum of 4, indicating that the Gabriel graph scales well because fewer packets would be replicated over multiple unicast streams. As with Delaunay triangulations, Gabriel graphs were effective in both uniform and exponentially distributed populations.*

the radius of the AOI was 1.6, each node had on average only 4 Gabriel neighbors. *In other words, as the population density increases, the average node degree, and therefore bandwidth requirements, increase very slowly.*

To further understand our results we examined the generated data in detail. We are particularly interested in the worse-case scenarios, so we used only the exponentially-distributed mobility model. Figure 8 presents the detailed results of the 0.4 radius simulation run for both the Delaunay and Gabriel graphs with 16, 64, 256 and 1024 nodes respectively. We plotted the minimum, the average and the maximum number of neighbors that a node had to maintain. Note that in this stacked chart, the difference between the minimum, average, and maximum values is presented so that the height of each bar represents its value on the y-axis correctly.

Figure 8 shows that the maximum number of neighbors is never more than twice the average number of neighbors, while the minimum number of neighbors can be quite small because some nodes are isolated. In addition, we see that the maximum Gabriel node degree is consistently smaller than the maximum Delaunay node degree. While we only show the values for the radius of 0.4, we had similar results for the other radii and node densities.

As described in Section 3.2, voice packets from v_i are transmitted to all AOI neighbors since they should be able to hear the avatar speaking. However, since we have built a Gabriel graph between every node and its AOI neighbors, the transmission is done in a hop-by-hop fashion. This design not only helps balance the traffic, but generates a realistic scenario where listeners closer to the speaker hear the voice packets before farther listeners.

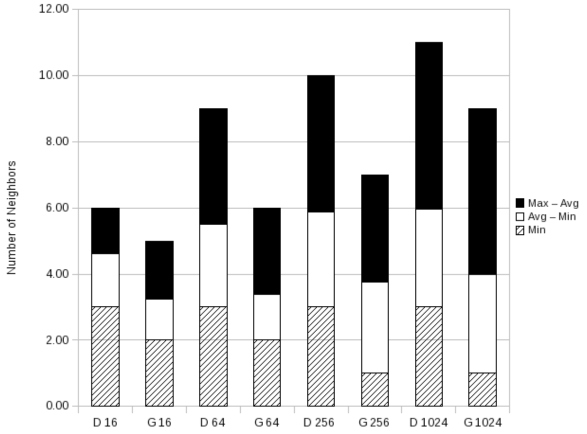
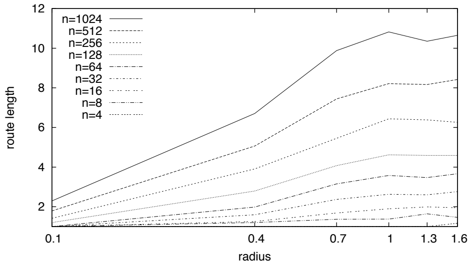
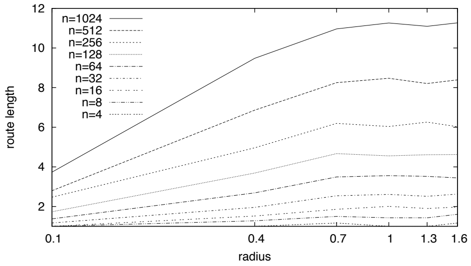


Figure 8: *Minimum, Average, and Maximum Neighbors: This figure illustrates the range of neighbors for a radius of 0.4 and compares both Delaunay and Gabriel graphs of increasing density (D16 is a Delaunay graph with 16 nodes while G16 is a Gabriel graph with 16 nodes). The results illustrate that the average, minimum, and maximum node degrees of the Gabriel graph are consistently smaller than the maximum Delaunay graph.*

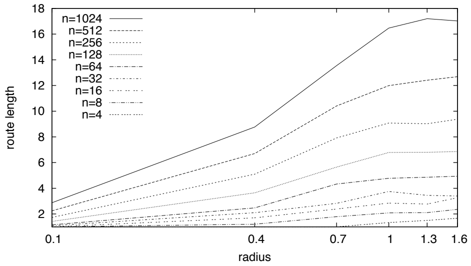


(a) Uniform distribution

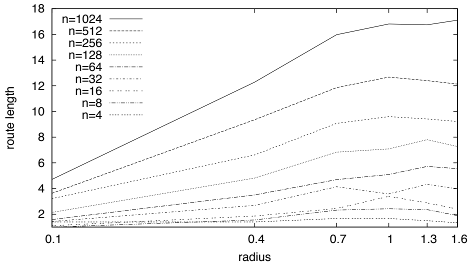


(b) Exponential distribution

Figure 9: *Average Route Length in the Delaunay Graph: this figure illustrates that the route length from the source to its destinations increases as the radius and node density increases. In particular, the Delaunay graph increases its route length logarithmically as the density and radius of the AOI increases.*



(a) Uniform distribution



(b) Exponential distribution

Figure 10: *Average Route Length in the Gabriel Graph: This figure illustrates the route length, in hops, from a speaking avatar to any other avatars within the speaker's AOI. In high density situations, such as when the radius is 1 (and encompasses almost the entire playing field), the number of hops appears to grow logarithmically with the number of nodes in the AOI.*

Figures 9 and 10 show the average number of hops for a packet to get from its source to a node in its AOI using the Delaunay and Gabriel graphs, respectively. In the worst case, the average number of hops is approximately 11 for Delaunay triangulations and 17 for our Gabriel graph protocol. However, this setup illustrates an extreme case where all 1024 nodes are within each others' hearing range and this value is an order of magnitude larger than a zone contains in World of Warcraft [Pittman and GauthierDickey 2007], for example. In a more realistic scenario where the radius of the AOI is 0.4 the average number of hops is only 9 and 11 for Delaunay and Gabriel graphs respectively.

On the other hand, even in this extreme case, the bandwidth required by each node would still be low, though the delay would be high for far away listeners due to the number of hops to forward packets to them. It can also be observed that the lower number of Gabriel neighbors results in a longer route length versus routes in the Delaunay graph. However, the Gabriel graph is a closer approximation of reality in that closer listeners to a given speaker in the virtual world will receive their voice packets before those that are farther, more accurately modeling the way that sound travels in the real world. Further, since the bandwidth required by voice communication will compete with bandwidth for the rest of the NVE, the Gabriel graph has the advantage of fewer neighbors to replicate packets to.

Again, we examined our data in detail to determine if in the worst case scenario the route length was significantly higher than it was on average. We present the minimum, the average and maximum route length for both the Delaunay and the Gabriel graphs. For this simulation we used 16, 64, 256 and 1024 nodes, our second mobility model, and we always used a radius that has a length 0.4 (see Figure 11). We found that the route length in the worst case scenario is only about 50% more than it is on average. Thus, our solution is well balanced and does not overload any of the nodes.

5 Adding Social Structures

Although our protocol focuses on building and maintaining a graph for location aware voice communication, it can be easily extended

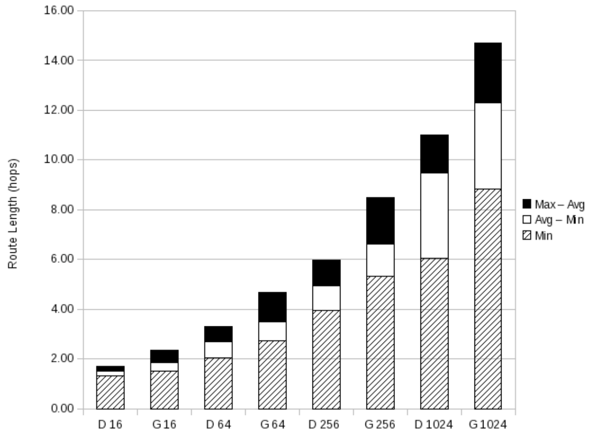


Figure 11: *Minimum, Average, and Maximum Route Lengths: This figure illustrates the range of route lengths for a radius of 0.4 in our simulations and compares both Delaunay and Gabriel graphs of increasing density. The results illustrate that the minimum, average, and maximum route length of the Gabriel graph is longer than that of the Delaunay graph.*

to maintain additional connections between the players to accommodate social structures such as guilds and *friends lists*. In these cases, a player desires to communicate with other players in the social structures which are far away in the virtual space, but using the Gabriel graph as we have designed could cause significant delay due to the number of hops it would need to take.

To handle these social structures, we propose extending the protocol to add additional edges to the graph so that players are directly connected to the other players in their social group. Fortunately, the extra edges do not effect the routing mechanism because they do not result in dead ends, or local maximums, and our original method guarantees that there is a route between any two nodes that are a part of the Gabriel graph. The drawback of this approach is the additional number of edges that a peer must support. Therefore, an alternative is to construct additional graphs for social structures such that only those members of the social structure are connected in the graph and the AOI of each member is sufficiently large to cover the entire membership. Given these parameters, a Gabriel graph will be constructed between members of the social structure allowing efficient communication between them.

6 Conclusions and Future Work

We have presented a new protocol for multiparty voice communication based on Gabriel graphs and positional information. Our protocol has five interesting properties: 1) positional information allows the voice packets to be mixed into a 2 or 3 dimensional space accurately; 2) voice packets are sent to the closest listeners first, since we have a direct link with them, who then forward them to farther listeners; 3) the average degree of any node in the system is smaller than 6 because Gabriel graphs are a subset of Delaunay triangulations; 4) the average number of hops in the system also appears to remain low, but depends on the density of players (though high density areas will mainly cause delayed voice data and not overwhelming voice data); and 5) the protocol can be used in both distributed and client/server NVEs.

Our results show that by using the Gabriel graph, packets would need to be replicated fewer times when forwarding them in an AOI-limited broadcast. Given that users tend to prefer higher bandwidth voice encodings, reducing the bandwidth requirements by a voice protocol is highly valuable.

As future work, we plan on further developing the use of Gabriel graphs to replace the need for Delaunay triangulations for neighbor maintenance and to study the voice network in a larger variety of simulation models. We also plan to investigate mixing methods to both reduce bandwidth and improve realism. Finally, we plan to implement our protocols as a library that can be used by NVEs.

References

Benford, S., Bowers, J., Fahlén, L. E., and Greenhalgh, C. 1994. Managing mutual awareness in collaborative virtual environments. In *VRST '94: Proceedings of the conference on Virtual reality software and technology*, World Scientific Publishing Co., Inc., River Edge, NJ, USA, 223–236.

Bin Muhammad, R. 2007. A geo-routing algorithm in planar graph for ad hoc wireless networks. In *Advanced Information Networking and Applications Workshops, 2007, AINAW '07. 21st International Conference on*, vol. 2, 685–689.

Bose, P., and Morin, P. 1999. Online routing in triangulations. In *ISAAC '99: Proceedings of the 10th International Symposium on Algorithms and Computation*, Springer-Verlag, London, UK, 113–122.

Dowlatshahi, M., and Safaei, F. 2006. Multipoint interactive communication for peer-to-peer environments. In *Proceedings of IEEE International Conference on Communication*.

Elleuch, W., and Houle, A. C. 2008. Multiparty voice over ip (MVoIP) peer-based system for large-scale conference support. In *Proceedings of IEEE Internation Conference on Wireless and Mobile Computing*.

Jiang, J.-R., and Chen, H.-S. 2007. Peer-to-peer aoi voice chatting for massively multiplayer online games. In *Parallel and Distributed Systems, 2007 International Conference on*, vol. 2, 1–8.

Lee, D.-Y., and Lam, S. S. 2007. Protocol design for dynamic delaunay triangulation. In *ICDCS '07: Proceedings of the 27th International Conference on Distributed Computing Systems*, IEEE Computer Society, Washington, DC, USA, 26.

Matula, D. W., and Sokal, R. R. 1980. Properties of gabriel graphs relevant to geographic variation research and clustering of points in the plane. *Geographical Analysis*.

Papp, G., and GauthierDickey, C. 2008. Characterizing and modeling multiparty voice communication for multiplayer games. In *Proceedings of the 19th ITC Specialist Seminar on Network Usage and Traffic*.

Pittman, D., and GauthierDickey, C. 2007. A measurement study of virtual populations in massively multiplayer online games. In *Proceedings of ACM NetGames*.

World of Warcraft. `http://www.worldofwarcraft.com`.

Wu, C., and Li, B. 2005. Optimal peer selection for minimum-delay peer-to-peer streaming with rateless codes. In *Proceedings of ACM Workshop on Advances in Peer-to-Peer Multimedia Streaming*.