A scalable Peer-to-Peer-overlay for real-time massively multiplayer online games

Philip Mildner, Tonio Triebel, Stephan Kopf, Wolfgang Effelsberg
Department of Computer Science IV
University of Mannheim
Germany
{mildner|triebel|kopf|effelsberg}@informatik.uni-mannheim.de

ABSTRACT

We propose a design for massively multiplayer games with focus on fast-paced action. To reflect the high demands on responsiveness and scalability, we use a fully distributed peer-to-peer network with a dynamic connection scheme. While the capacity of current client/server systems limits the number of players in a game, our system supports a huge number of users by exploiting local interests in the virtual world. By using a publish/subscribe system for message dissemination user interests can be handled efficiently. A Geocast algorithm allows information distribution to arbitrary regions of the virtual world. Simulations show that our design scales well by limiting the number of connections per user even in crowded regions. Moreover, different movement strategies are evaluated for their impact on network load and connection dynamics.

Categories and Subject Descriptors

C.2.4 [**Computer-Communication Networks**]: Distributed Systems

General Terms

Algorithms, Design, Experimentation

Keywords

networked virtual environment, peer-to-peer, scalability

1. INTRODUCTION

In the past years, the market for massively multiplayer online games (MMOGs) has increased fast. Millions of users are spending their time in virtual worlds while playing with or against other users. A major part of this kind of applications are massively multiplayer online role playing games (MMORPGs) with *World of Warcraft* being one of the most prominent representatives, but the field of MMOGs is not restricted to this genre. Another example of virtual worlds is the social platform *Second Life*, where users can freely interact in a mostly user-generated world. Nowadays, the trend goes to massive fast-paced games and massively multiplayer online first person shooters (MMOFPS) become a focus of research.

All these networked virtual environments (NVEs) have common properties: a potentially huge number of users is connected in a virtual world in order to interact with the world or other users. The fact that the users want to share a common virtual world imposes a challenge on the underlying network structure. Using a client/server architecture NVEs are limited by the capacity of the servers. Operators can circumvent these limitations by splitting up the virtual world into separate regions that are maintained by different servers or by mirroring the NVE to different worlds. Both approaches have the disadvantage that users are no longer part of the same world. Methods that transfer the NVE to a computer grid impose high costs for the operators. Moreover, the client/server architecture suffers from known problems like *single-point-of-failure* and high costs for maintenance.

In order to fix these problems, several NVE implementations based on Peer-to-Peer (P2P) technology were proposed. Thereby, the requirements decisively differ from other P2P-applications like data storage and distribution. While in classic P2P-applications both data and the position of the users in the network usually are static, P2P-NVEs have to deal with highly dynamic content. Not only do the players move through the virtual world, but due to the interaction the virtual world itself and the associated data constantly change. Moreover, connections are not established based on the topology of the underlying network, but based on decisions made in the virtual world, e.g. when users interact in the NVE. Based on these requirements a variety of P2P-NVE systems were developed.

In this paper we propose a P2P-overlay for a NVE that is designated for MMOFPS. Other than common NVE systems that are mostly tailored for MMORPGs, this system focuses on responsiveness and users' dynamics than on persistency. Therefore, a dynamic approach to connect users in a P2P-overlay is used. By minimizing the overhead for connection management, a highly responsive system can be created. Contrary to the majority of existing systems, the primary metric to connect users is not distance-based but based on the number of nearest neighbors within the NVE. While existing systems usually use a sender-oriented message distribution approach, we propose to use a publish/subscribe mechanism. Thus, overlay inconsistencies can be avoided and users' interest can be mapped to connections within the NVE efficiently. Moreover, a Geocast algorithm allows for sending messages to an arbitrary set of users based on their positions in the NVE. To evaluate the proposed concepts, the NVE system was implemented both in a simulation environment and the existing game PlanetΠ4 [18].

The remainder of this paper is organized as follows. Section 2 gives an overview of scalability issues in NVE systems and presents related work. In Section 3 the basic design of our approach is presented. Section 4 discusses the algorithms of the system that are evaluated in the following section. Section 6 concludes the paper and gives an outlook to future work.

2. RELATED WORK

With increasing popularity of MMOGs, a variety of P2P-NVE systems have been developed. Following existing online games, the majority of these systems are optimized for MMORPGs. On the other hand, fast-paced action games have not arrived at a massive online experience. Most of these games still rely on dedicated servers that a capable to host at most 32–64 players. Thereby, the requirements in terms of responsiveness and low latencies are the main challenge for building a multiplayer action game that supports a large number of players. As a consequence, games of this genre cannot simply be transferred into massive online games using available NVE designs. Instead of relying on a consistent and persistent virtual world that is achieved by increased redundancy and information dissemination time, the mostly session-based MMOFPS should focus on low response times, as FPS in general require low latencies for a proper gaming experience [1]. To accomplish that, all interacting users should be directly connected via a one-hop overlay neighborhood. As this can pose a substantial load in a P2P-environment, the communication overlay has to be designed in a way that allows an appropriate scalability of the virtual world.

Hu et al. named six issues that have to be considered when creating a NVE with *scalability* being one of the most important criteria [8]. While scalability in client/server architectures depends on the servers' resources, scalability in P2P-environments is achieved by an sophisticated algorithm that determines how users communicate with each other.

First approaches tried to optimize fully connected meshes by reducing the amount of sent messages with message filtering [6]. As a fully connected mesh inherently imposes communication costs of $O(n^2)$, a proper scalability cannot be achieved. Another approach for reducing communication cost is the use of multicast. Respective systems like [12] use application layer multicast to reduce the amount of sent messages. A major disadvantage of such systems is the increased response time that comes with the message forwarding within the multicast structure.

Instead of optimizing a fully connected network, the key to scalability is to use local connections between users. Therefore, the concept of an *area-of-interest* (AOI) is used: Similar to the real world, each user only perceives events that occur in his neighborhood. Contrary to common P2P-networks where geographic locations can be used to find appropriate connections between users, in NVEs the positions of the users' characters in the virtual world – so called avatars – have to be used to build up the connections. Consequently, users only build up connections to other users whose avatars are in the neighborhood of the own avatar. Without using a centralized connected network, mechanisms to ensure global connectivity of the network are required. This can be achieved by local management of each node or by distributed techniques like the one presented in [3] where distributed hash tables (DHTs) are used for storage and lookup of objects in the virtual world. As pure P2P-network, VON [8] uses Voronoi diagrams to keep a connection between users, whereas Solipsis [5] uses convex hulls. pSense [16] uses designated *sensor nodes* that ensure global connectivity by establishing connections to up to eight nodes that are located outside of the own AOI.

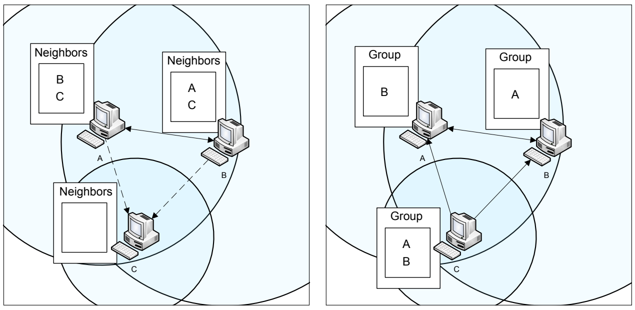
In case of a static setup of the AOI concept the virtual world is divided in parts of pre-defined sizes. Users that share a common region are connected to each other. Knutsson et al. [11] use a hybrid P2P-approach where each region is managed by a super-peer. These *coordinators* are connected separately to ensure global connectivity of the virtual world. Due to the static setup the regions do not correspond to the real AOI of the users. If an avatar is located near a border of a region, the user cannot see users of neighboring regions. Workarounds like connecting to more than one region at one time can circumvent this, but it increases the load on the network and the amount of irrelevant information, because the source of the information is not located in the AOI of affected users.

In order to map the user's AOI to the NVE more efficiently, dynamic approaches can be used. In contrast to static approaches, the virtual world is not divided into areas of fixed size but in a set of dynamically changing regions. VON uses Voronoi diagrams to create a dynamic region for every user in the NVE. Based on the shape of the regions, every user dynamically decides to which other users he connects to cover his AOI. Therefore, every user maintains his own AOI that moves with his avatar through the virtual world. It simply can be modeled as a circle around the avatar, but it also can include more complex metrics like line-of-sight or history of user-interaction [2]. If there is no control structure upon the P2P-network like super-peers or DHTs, there has to be some mechanism to discover new neighbors in the virtual world. As users move their avatars through the NVE, virtual neighbors constantly change. In a fully-distributed network there exist two common ways to implement the neighbor discovery: Either users constantly exchange their neighborhood lists or dedicated users take over the role of *watchdogs*. These nodes look out for users that enter the AOI of the watched user and send him a notification in this case.

One reason for the diversity of the NVE systems is the variety of possible application scenarios. While some NVEs focus on persistency and comparatively slow game mechanisms, other systems require a very low latency. This also applies for the available bandwidth, which can be restricted in some scenarios (e.g. applications on mobile devices) whereas in other situations a higher amount of available bandwidth could allow a higher responsiveness. As a consequence, the NVE has to compromise on the various options to provide an optimal system depending on a specific application.

3. ARCHITECTURE

To reflect the special requirements of MMOFPS, our design of the NVE is optimized in terms of user dynamics and responsiveness. Static structures are avoided whenever possible to achieve a high overlap of the AOI and the user's real interests in the NVE. To minimize communication overhead, no overlying structures like DHTs or super-peers are used. Consequently, a fully-distributed P2P-network was chosen as the underlying communication layer and dynamic regions are used for the management of the the NVE.



(a) Sender oriented model (b) Publish/subscribe model

Figure 1: A publish/subscribe mechanism prevents interest conflicts of neighboring users.

As there is no overlying control structure in a fully-distributed network, the nodes also have to implement the management in a distributed manner. Our system follows the principle that every node should manage its own responsibilities. From this follows that each user has to take care of the information he wants to receive in order to cover his AOI. Existing systems often use a sender-oriented model for information exchange: Each user determines the set of users that are located in his AOI and subsequently sends messages to these users. Especially when users do not share a common AOI this can lead to an overhead of message exchange and also to inconsistencies. As seen in Figure 1(a), there are two problems that arise from the sender-oriented model. On the one hand, users get information they are not interested in. This increases the amount of sent messages without a benefit for the users. Moreover, users possibly do not get the whole information about their AOI, because interest in a specific area of the NVE is not bidirectional when users do not share a common AOI. This leads to problems that at worst can cause overlay inconsistencies.

In order to correct the problems that arise with a sender-oriented model we use a publish/subscribe system for information dissemination. Instead of sending messages to users in the own AOI, users get all information that originates in the own AOI by subscribing to users that are located in the AOI. Therefore, every user publishes all information that affects him to a group other users can subscribe to. Thus, users do not actively control the set of receivers of their messages, but they control a set of senders that publish their messages to all subscribed users. With this mechanism an optimal coverage of the user interest sets can be achieved (see Figure 1(b)).

4. NVE-COMPONENTS

4.1 Users of interest

When designing a NVE, scalability is one of the key features. To achieve an appropriate scalability, the implementation of the AOI has strong influence. The majority of existing NVE implementations use a circular area with a certain radius around the avatar of a user to determine the part of the virtual world the user currently is interested in. Consequently, connections are established with all users whose avatars are located within a radius of the own avatar. While this works well in sparsely populated areas of the NVE, the approach suffers when the population grows, e.g., when many users gather in a small area. Since users establish connections to all users in their AOI, the amount of connections can exceed their bandwidth in such situations. As these gathering events are a common event in MMOGs, the NVE system has to be able to deal with such a situation.

In a NVE that uses dynamic regions one way to prevent overloads caused by too many users in the AOI is to dynamically adapt the size of the AOI [7]. This can be done by a threshold of maximum allowed connections. If the threshold is reached, the radius of the AOI is reduced. When the gathering event is dissolved it can be increased to its original size. Another way to decrease the load on the users is to vary the frequency of message exchange based on the distance within the AOI [9]. As this increases the response times, it is not suitable for a MMOFPS scenario. Our approach generalizes the idea of an adapting AOI by operating with a threshold of maximum connections like in [10]. Instead of connecting to users that are located within a certain radius around the own avatar, connections are established to the nearest n neighbors, with n being a user-defined parameter that specifies the amount of desired connections. The real amount of connections can vary from that value, e.g., if there are not enough users in the NVE. We chose this approach because of two reasons:

Firstly, the adaption of a radius by limiting the amount of users is similar to directly defining a fixed amount of connections. Both approaches limit the maximum amount of connections. However, the radius-based approach reduces the amount of connections if the AOI is smaller than the distance to the n nearest neighbors and therefore limits the user's sight distance. Our approach lets the user get as much information about his neighborhood as he can get without exceeding his bandwidth constraint.

Secondly, the user-limited approach matches the human perception more appropriately. This idea is based on the following observation: Humans normally only focus a small amount of objects in their surrounding [14]. Thereby, the amount of objects that lie within the focus is more important than the distance to the objects. This circumstance can be transferred to the virtual world. By defining a smart metric to find few but important objects for a specific user, the AOI can be modeled to fit the user's real interest. Like in [10], we use a simple approach using the n nearest neighbors, but our system can easily be extended to support further metrics like the ones presented in [15].

4.2 Overlay consistency

In an evenly populated NVE, the previously presented neighborhood-based approach would suffice to connect all users in a consistent overlay network[1]. As users of MMOGs tend to build groups or gather in specific regions, this cannot be assumed. So the neighborhood-based approach can lead to overlay inconsistencies and even to a partitioning of the

[1]In this paper, we use the term consistency purely for overlay consistency, so we do not explicitly consider the consistency of the virtual world.

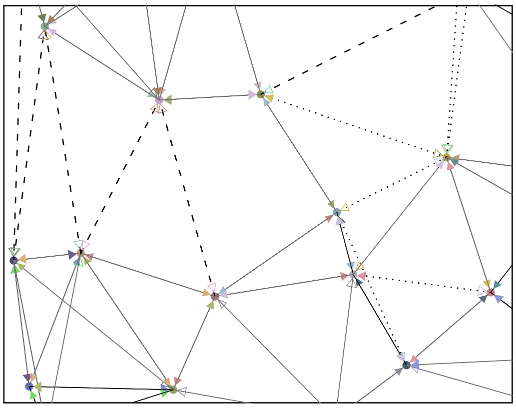


Figure 2: Pure NetConnector connections (sparsely dotted) ensure overlay consistency, whereas prevention of unidirectional connections (dotted) avoid unfair gaming situations.

NVE if the distance between groups of users is greater than the distance to the respective nearest neighbors. Since a partitioning cannot be repaired by the NVE itself but only by manual intervention of the users, it has to be prevented as effectively as possible.

To ensure consistency of the NVE we extended our purely neighborhood-based approach by introducing a new type of connection, so called *NetConnectors*. These connections take advantage from the fact that connections between users are built upon decisions made in the virtual world, e.g., the neighborhood between the virtual characters. Similarly, the structure of the virtual world can help to keep the overlay consistent. Based on the principle that each user in the NVE should keep at least one connection to a user in each cardinal direction, a set of consistency conserving connections can be determined. Those connections do not necessarily coincide with the set of nearest neighbors, so the amount of NetConnector connections is always a trade-off between user interest and overlay consistency. pSense uses a similar approach, but as its sensor nodes always are located outside of the users' AOI, they always pose an additional load on the users. To maximize the overlapping of the two sets, NetConnector nodes always are the nearest node in the corresponding area. Generally, the NetConnector nodes are determined by the following rules:

- Given an D dimensional virtual world, the number of designated NetConnector connections N has to satisfy $N \geq 2^D$.
- The space around an avatar is divided into N equal sized parts.
- In each of the resulting regions a user has to establish one connection to the nearest node whose avatar is located in this region if such a node exists.

In the case of $D = 2$, the minimal number of four regions can be obtained by dividing the virtual world at the axes of coordinates that originate from the position of the own avatar. While the minimal number of NetConnectors ensures a consistently connected overlay, a higher number can improve stability in the case of connection loss. This is further analyzed in the evaluation part in Section 5.

4.3 Neighbor discovery

To meet the requirements of a fast-paced MMOFPS we chose an unstructured P2P-overlay to connect users. One issue that arises with this decision is to solve the problem of neighbor discovery. As there is no coordinating central instance or super-peer functionality, users have to communicate with each other to find new neighbors within the virtual world. Therefore, two main approaches exist. Either users can periodically exchange neighbor lists or designated nodes – so-called *watchdogs* – inform users if new nodes enter their AOI. While the list-based approach does not require additional management for maintaining the set of watchdogs, it increases network load because the lists are exchanged even if neighborhoods are static.

We decided to use a dynamic watchdog approach that combines the properties of both techniques. Users do not maintain a set of designated watchdog nodes, but they periodically send a request including their own known nodes to all known neighbors. If a receiving node knows a node that is more appropriate in terms of neighborhood than the nodes the requesting node knows, it sends back a message to the requesting node including information about the new neighbor candidate. The requesting node then locally decides whether it wants to connect to the newly discovered node. This procedure has the advantage that users do not have to maintain any status information about neighbors of other users and they are able to locally initiate neighbor discovery by sending out a request. This follows the principle that each user should manage his responsibilities on his own.

4.4 State management

As our NVE system is intended for gaming environments, the interest management as it is described above has to implement another feature. Consider a scenario like shown in Figure 1. If neighboring users have a different AOI, connections are only build up in one direction. This corresponds to the different interest sets, but it also imposes an imbalance on the gaming experience. If users do not see each other in the virtual world, some users can gain an unfair advantage, e.g., by firing at users which themselves cannot see the other users. To correct that, connections should always be bidirectional even when users are not explicitly interested in these connections. These *secondary connections* are only kept as long as the connection partner is interested in them. When the primary connection partner is no longer interested in such a connection, the secondary connection partner ends the connection as well.

To manage all connections, each user maintains a table of connection states which can be used to identify the state of a specific connection. In particular, the connection state includes the following information:

Phase of connection Determines the general state of the connection. It can be *connecting*, *disconnecting* or *established*.

NetConnector state Determines whether the connection partner is a NetConnector or if the own node is a NetConnector to the connection partner.

Type of connection Determines if the connection is primary (user has interest in this connection) or secondary (connection is just kept to prevent a unidirectional connection).

As each of the values is separated into incoming and outgoing states, the table consists of six boolean values that completely characterize the state of a connection.

When only one connection partner is interested in a connection, he also has the full control of the connection. As soon as both partners have an interest, they need to communicate their connection states until one user loses interest in the connection. Thus, a pure local management of the connection would not be sufficient, because users need to know when they can end a bidirectional connection. This only occurs if there are no NetConnectors in the connection and none of the connection partners have interest in this connection, respectively the connection is secondary in both directions. To synchronize their connection state, users just have to send the connection state to the connection partner. Although connection states can change constantly before a connection is ended, it only needs one properly received state message to fully synchronize the connection state between two users. So the state management is resistant against occurring message loss. Furthermore, each user is able to locally manage all his connections.

4.5 Error recovery

Connection errors can be divided into two categories: Repairable errors are states that can be corrected by the network itself, whereas irreparable errors, e.g., a partitioning of the network, can only be corrected by a manual intervention of the user. From this follows that irreparable states should be prevented at all cost. We focus on two error sources: inconsistent connection states and complete connection losses.

A special case of inconsistent connections are purely unidirectional connections, because they can be treated locally. Those occur especially at connection setup. If an error occurs during sending of an initiating connect message, the sending user simply can resend the message if he did not receive an answer after a specified amount of time. However, inconsistent bidirectional connections have to be treated by both connection partners. While errors at connection setup can be treated locally, there is a possibility that connections are not closed even if none of the connection partners have interest in it any more. This can happen if a status message is lost that passes the control of the connection to the connection partner. Consequently, the partner is not able to end the connection, since he still considers the interest as bidirectional.

Not required connections do not imply a threat to overlay consistency, but they impose an unnecessary load on the involved users. To correct problems regarding these unnecessary connections, users periodically can notify their connection partners about the current connection state. To avoid further overhead, this information can be combined with neighbor discovery requests. This is sufficient, because one message is enough to completely synchronize the state of a connection.

As a MMOG is a highly dynamic environment, users constantly join and leave the system. Exits can happen intentionally if users decide to end the session, but they also can happen if the connection is lost due to network problems or software failures. While the former is a normal operation within the NVE, the latter can pose a serious danger to overlay consistency. If users exit the NVE unintentionally, a remaining user or even a group of users can loose the connection to the rest of the NVE resulting in a partitioning of the network. In any such event, neighboring users of the exiting user loose information about their neighborhood. To restore overlay consistency and users' view on the NVE, a repair mechanism has to be provided.

In order to perform a successful repair procedure for a specific user, the user must have at least one remaining connection to the rest of the NVE. If that is not the case, the error is irreparable by the NVE and requires a manual reconnect of the user. Otherwise, the following procedure can restore overlay consistency: After a user has detected a connection loss of a neighboring node via a message timeout, he tries to replace the no longer present neighbor with a new node. For this purpose, regular communication would not suffice to find appropriate neighbor candidates, because it requires a specific destination. As users do not have any information about the part of the virtual world they are trying to reach, there has to be another criterion how the message is forwarded. The general idea is to find the user whose avatar is located behind the lost node as this user can provide information about the surrounding area and new neighbor candidates. This search can be modeled as a minimization technique. The requesting user tries to find the user with the smallest angle between the own node and the node that lost the connection. Therefore, a message is repeatedly forwarded to the user with the smallest known angle. If the current user does not know a user with a smaller angle, the search is finished and the user sends a message containing his neighbors back to the requesting node. There is a possibility that the minimization ends up in a local minimum. As an approximate solution suffices to restore overlay consistency, this does not pose a problem.

4.6 Geocast

In normal operation, messages are just exchanged between neighboring nodes. While this is sufficient for position updates and direct interaction between users, there might be situations where messages have to be sent to users that are located beyond the own AOI. One example in MMOFPS could be a weapon with an area-based damage. To support such kind of messages we included a Geocast algorithm in our system that is able to send messages to an arbitrary set of users. The set is dynamically determined by a pre-defined region for which the Geocast message is addressed.

Geocast algorithms are often used in ad-hoc networks to send messages to a certain region within the network [13]. The key feature of those messages is that they are not explicitly addressed to specific users but to a certain region defined by its coordinates and its shape. Consequently, all users that are located in an addressed region receive the Geocast message. The mechanism normally consists of two steps: Firstly, a message is forwarded from the sender to the destination area without being processed by the forwarding nodes. Secondly, the message is distributed to all users in this region and then can be processed by its receivers. While the forwarding part can be implemented via simple unicast submission towards the destination, the dissemination of the Geocast is more difficult. A trivial solution is flooding of the destination area to reach all receivers. However, as this imposes a high load on the network more sophisticated methods should be used.

There is one substantial difference between standard Geocasts and Geocast algorithms for NVEs. As the overlay for NVEs is built based on the positions of the avatars, the

Geocast algorithms does not use the geographical position of the users to distribute its messages but the positions in the virtual world. With the use of an unstructured P2P-network, there is no global information on which users are located in a specific area. As a consequence, a flooding-based approach is used to reach all users in the destination area. To minimize the network load imposed by the flooding process we propose to make use of NetConnector nodes to distribute a Geocast in the destination area. This is appropriate because NetConnectors are explicitly designed to cover the whole area around a node.

In detail, the Geocast algorithm is designed as follows: A Geocast consists of a message containing destination and distribution radius, a unique identifier and a payload that is distributed in the destination area. Beginning with the sender, each receiving node forwards the message to the node that has the shortest distance to the destination of the Geocast. If no such node exists and the receiving node's position is in the destination are, the message has arrived its destination and the first phase ends. In the second phase, the Geocast is distributed in the destination area by spreading it from the center of the area to its boundaries. For our purposes we chose a circular distribution area. A node forwards a message to the groups of its NetConnectors if the following conditions are fulfilled:

- The NetConnector has to be further away from the origin of the Geocast than the own node but still inside of the destination area.
- The distance from the NetConnector to the origin has to be greater than the distance from the node that delivered the message to the origin.
- The own node must not be a direct receiver of the message but just a subscriber to the group of the addressed node.

By using this mechanism the growth of required messages for a Geocast can be kept linearly. Furthermore, an endless forwarding of messages, as it might happen in the case of flooding, can be prevented, because messages are always sent further away from the origin until they reach the boundary of the distribution area.

4.7 Implementation

The developed NVE system was implemented in two ways. To test the concepts in a real-world application, the algorithms were integrated into the existing MMOFPS PlanetΠ4. The game was developed in the style of the space shooter *Descent* and has high demands on the underlying network in terms of responsiveness due to its fast-paced game design. It uses the base overlay and further services of the SpoVNet project [17]. This allows connection between users across heterogeneous networks using the SpoVNet base *ariba* [4]. Implemented in C++, PlanetΠ4 uses the *Irrlicht* engine[2] as 3D engine. Tests with several players and computer-controlled bots showed promising results in terms of responsiveness and general user experience.

As the system should be tested for its scalability, it also was implemented in a simulation environment. In this way, it was possible to easily simulate a NVE with thousands of users. The simulation does not only simulate static connections between virtual nodes, but it also simulates different movement strategies of the virtual users that can be combined to get a realistic user behavior.

[2]http://irrlicht.sourceforge.net/

5. EVALUATION

In order to gain knowledge about the quality of the developed concepts, the implementation was tested with different parameters and configurations. The simulations were divided into static and dynamic runs. In static setups, the general capability of the NVE was evaluated, whereas dynamic simulations allowed statements about stability of the network and its properties under different user behaviors.

5.1 Static simulations

To gain basic knowledge about quality of the developed concepts, different simulations with a static node setup were evaluated. In these simulations, nodes joined the network, but they did not move or leave it. Although this setup does not model the full functionality of the NVE, it still allows statements about the connection setup.

A scalable system has to satisfy two requirements: Users that enter the system must not consume resources of a single resource limited component and the amount of available resources has to increase with a growing number of users in the system. As a consequence, the load on a single user must not increase when new users enter the NVE. To measure the load, a series of simulations was executed in which the amount of users varied from 500 to 5000. The results show that the number of connections per user does not increase with a growing number of total users, because the number of connections are limited by the design of the AOI. This is a basic requirement for a scalable system, as the load on a specific user is independent of the total amount of users in the system.

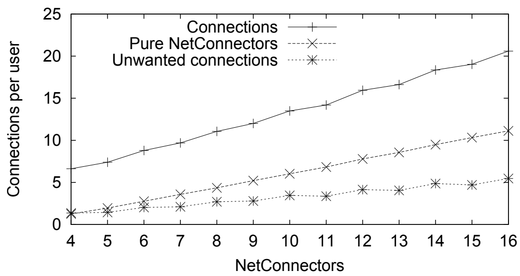


Figure 3: An increased number of NetConnector connections leads to an increased number of connections a user has no interest in.

Our concept was designed to achieve a maximal coverage of user interests and connections within the NVE. However, there are two situations that force users to establish connections they are not primarily interested in. Those are firstly connections to NetConnectors that do not coincide with the neighbor set and secondly connections that are only established to prevent unidirectional connections. To measure the influence of these secondary connections, several simulations with different parameter configurations were run. While the amount of users was fixed at 1000, the number of nearest neighbors and NetConnectors varied from four to 16. Results show that NetConnectors have a higher influence on

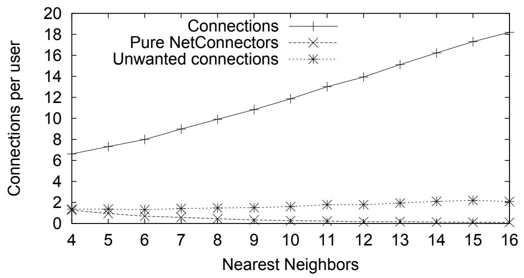
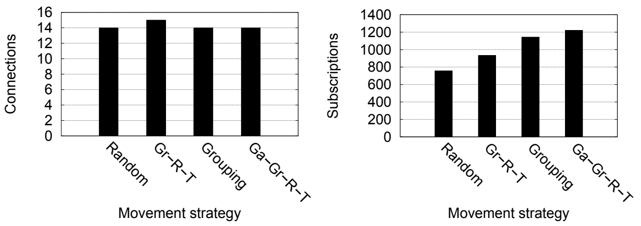


Figure 4: Connections to nearest neighbors do not influence the number of NetConnector connections.

secondary connections than neighbor connections (see Figure 3). With an increasing amount of neighbor connections, secondary connections nearly stay constant while pure NetConnector connections converge to zero (see Figure 4). With a high amount of NetConnectors however, both connection types increase with pure NetConnector connections being the biggest influence. From this follows that from the user's point of view the amount of nearest neighbors is more important than NetConnectors. This corresponds to the design of the different connection types: Nearest neighbors are directly connected with the user's interest set, whereas NetConnectors are important for the network's consistency and stability.

5.2 Movement



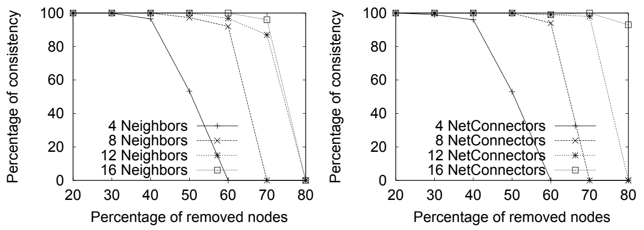
(a) Amount of connections per user (b) Amount of subscriptions per user

Figure 5: Movement strategies include random (R), tracking (T), grouping (Gr) and gathering (Ga) patterns.

As the movement of avatars in the virtual world is the most essential action in a NVE, it has to be considered when evaluating the scalability of the system. Therefore, several movement strategies were implemented in the simulation environment to measure their impact on the connection dynamics. A random waypoint pattern served as a basis, whereas a tracking pattern was added to simulate a common situation in action games where players chase each other in a fight. To investigate the network's stability against crowding situations, a group-based random waypoint pattern was added. Moreover, a gathering scenario was implemented where a specified amount of users gather in a small area of the NVE. This last pattern serves as a worst-case scenario, because it creates a high load on the involved users. All four patterns can be combined to create a realistic movement behavior of the virtual users.

Each of the simulation runs was executed with 500 nodes in the NVE, eight nearest neighbors and the minimum number of four NetConnectors. After a fixed amount of simulation steps the number of connections per user and the number of sent subscriptions for new connections were saved. Results show that the number of connections do not significantly differ among the different movement patterns (see Figure 5(a)). This is a direct consequence of the chosen design, because the number of established connections per user is fixed. Variations only occur due to unidirectional and pure NetConnector connections which can be higher if the users are not distributed uniformly. However, the number of established connections significantly differ between the random and the mixed pattern that includes a gathering situation (see Figure 5(b)). This also can be explained with the fixed number of connections: Since a crowding situation implies a low distance between users, the set of nearest neighbors changes frequently. A way to decrease connection dynamics would be to increase the subscription interval. This can be done locally in each node by defining the period of sent neighbor discovery requests. As this can lead to a temporal neighborhood inconsistency and decreased stability it should only be done if users experience a network load that could exceed their capabilities.

5.3 Stability



(a) Neighbors (b) NetConnectors

Figure 6: The amount of NetConnectors has a high influence on the possibility of overlay recovery in case of connection loss.

Results from static simulations showed that the number of nearest neighbors is directly connected to the users' interest in the virtual world, whereas NetConnector connections can increase the number of unwanted connections. From the network's point of view however, NetConnector connections are essential to ensure overlay consistency. Moreover, NetConnectors help to improve stability in case of connection losses. To examine this, several simulations with 250 nodes were run in which a varying amount of nodes was removed from the network. The number of nodes that lost the connection to the rest of the network after this operation determine the grade of inconsistency. Results show that NetConnectors have a higher influence on the ability of a successful consistency recovery (see Figure 6). This can be explained by the approximately uniform connection setup in terms of geographical coverage of the virtual world that results from NetConnector connections.

6. CONCLUSION

In this paper, we presented a NVE system that is optimized for massive fast-paced action games while focusing on scalability and responsiveness of the system. In order to reflect the high demand of responsiveness, a pure P2P-network with a dynamic connection scheme was chosen. Instead of dividing the virtual world into regions of fixed size, each user locally decides to which other users he directly connects. This decision is based on the design of the AOI: We decided to use an AOI that is not distance-based but based on the number of nearest neighbors, since this complies with human perception to only focus on a limited number of objects at one time. By introducing a publish/subscribe system for message exchange, an optimal coverage of users' interests in the virtual world and established connections can be achieved. To ensure global connectivity of the network, a special kind of connections was introduced. These NetConnectors prevent a partitioning of the network by connecting every user to another user in at least each of the virtual world's cardinal directions.

The algorithms were both implemented in the existing game PlanetΠ4 and a separate simulation environment. Results from various simulations show that the neighborhood-based scheme provides a scalable and consistent overlay by limiting the number of connections per user. In crowded user setups the number of connections remains stable, whereas the connection dynamics vary substantially to provide a consistent view on the NVE.

As a further step, the effect of heterogeneous connection parameters and additional neighborhood metrics on the system can be examined. While parameters were defined globally so far, an individual adaption based on every user's capabilities can lead to a more appropriate exploitation of available resources. By extending the purely distance-based neighborhood determination to further metrics like line-of-sight or interaction history, a more consistent neighborhood set can be achieved that increases user experience. Finally, a user evaluation with PlanetΠ4 can help to gather more information about user experience and the applicability of the system in real-world scenarios.

7. REFERENCES

[1] T. Beigbeder, R. Coughlan, C. Lusher, J. Plunkett, E. Agu, and M. Claypool. The effects of loss and latency on user performance in unreal tournament 2003. In *NetGames '04: Proceedings of 3rd ACM SIGCOMM workshop on Network and system support for games*, pages 144–151. ACM, 2004.

[2] A. Bharambe, J. R. Douceur, J. R. Lorch, T. Moscibroda, J. Pang, S. Seshan, and X. Zhuang. Donnybrook: Enabling Large-Scale, High-Speed, Peer-to-Peer Games. In *SIGCOMM '08: Proceedings of the 2008 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 389–400, Aug. 2008.

[3] A. Bharambe, J. Pang, and S. Seshan. Colyseus: A distributed architecture for online multiplayer games. In *NSDI'06: Proceedings of the 3rd conference on Networked Systems Design & Implementation*, page 12, Berkeley, CA, USA, 2006. USENIX Association.

[4] R. Bless, C. Hubsch, S. Mies, and O. P. Waldhorst. The Underlay Abstraction in the Spontaneous Virtual Networks (SpoVNet) Architecture. In *2008 Next Generation Internet Networks*, pages 115–122. IEEE, Apr. 2008.

[5] D. Frey, J. Royan, R. Piegay, A.-M. Kermarrec, E. Anceaume, and F. Le Fessant. Solipsis: A Decentralized Architecture for Virtual Environments. In *International Workshop on Massively Multiuser Virtual Environments (MMVE'08)*. IEEE, 2008.

[6] A. Goldin and C. Gotsman. Geometric message-filtering protocols for distributed multiagent environments. *Presence: Teleoperators and Virtual Environments*, 13(3):279–295, 2004.

[7] S.-Y. Hu, J.-F. Chen, and T.-H. Chen. VON: A scalable peer-to-peer network for virtual environments. *IEEE Network*, 20(4):22–31, July 2006.

[8] S.-Y. Hu and G.-M. Liao. Scalable peer-to-peer networked virtual environment. In *NetGames '04: Proceedings of 3rd ACM SIGCOMM workshop on Network and system support for games*, pages 129–133, New York, NY, USA, 2004. ACM.

[9] J.-R. Jiang, Y.-L. Huang, and S.-Y. Hu. Scalable AOI-Cast for Peer-to-Peer Networked Virtual Environments. In *ICDCSW '08: Proceedings of the 28th International Conference on Distributed Computing Systems Workshops*, pages 447–452, Washington, DC, USA, 2008. IEEE Computer Society.

[10] Y. Kawahara, H. Morikawa, and T. Aoyama. A peer-to-peer message exchange scheme for large scale networked virtual environments. In *ICCS '02: Proceedings of the 8th International Conference on Communication Systems*, pages 957–961, Washington, DC, USA, 2002. IEEE Computer Society.

[11] B. Knutsson, H. Lu, W. Xu, and B. Hopkins. Peer-to-peer support for massively multiplayer games. In *INFOCOM 2004. Twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies*, volume 1, pages 107–118, 2004.

[12] J. Liebeherr, M. Nahas, and W. Si. Application-layer multicasting with Delaunay triangulation overlays. *IEEE Journal on Selected Areas in Communications*, 20(8):1472–1488, Oct. 2002.

[13] C. Maihöfer. A survey of geocast routing protocols. *IEEE Communications Surveys and Tutorials*, 6(1-4):32–42, 2004.

[14] G. A. Miller. The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological Review*, 63(2):81–97, 1956.

[15] J. Pang, F. Uyeda, and J. R. Lorch. Scaling Peer-to-Peer Games in Low-Bandwidth Environments. In *IPTPS '07: Proceedings of the 6th International Workshop on Peer-to-Peer Systems*, Feb. 2007.

[16] A. Schmieg, M. Stieler, S. Jeckel, P. Kabus, B. Kemme, and A. Buchmann. pSense - Maintaining a dynamic localized peer-to-peer structure for position based multicast in games. In *IEEE International Conference on Peer-to-Peer Computing*, 2008.

[17] The SpoVNet Consortium, B. Koldehofe, and G. G. Koch. SpoVNet: An Architecture for Supporting Future Internet Applications. In *Proc. of the 7th Würzburg Workshop on IP: Joint EuroFGI and ITG Workshop on "Visions of Future Generation Networks"*, Würzburg, Germany, 2007.

[18] T. Triebel, B. Guthier, and W. Effelsberg. Skype4Games. In *Proceedings of the 6th ACM SIGCOMM workshop on Network and system support for games - NetGames '07*, pages 13–18, New York, USA, 2007. ACM Press.