Universität Mannheim
Fakultät Mathematik und Informatik
Seminar: Algorithmen zur Unterstützung von Immersive Gaming
Dozent: Tonio Triebel
Herbst-Wintersemester 2008

Immersive Voice Communication:
Sprachübertragung in bevölkerten virtuellen Welten.

Verfasst von: Andreas Orzyszek
7. Semester Software und Internettechnologie
Matrikelnummer: 977939
E-Mail: aorzysze@rumms.uni-mannheim.de

Inhaltsverzeichnis

Abstract

Diese Arbeit liefert einen Überblick über die Bereitstellung und Verwendung von Immersive Voice Communication in großen skalierbaren Welten. Zu diesem Zweck vergleiche ich im ersten Teil verschiedene Möglichkeiten einen solchen Dienst überhaupt anzubieten. Zur Diskussion stehen dabei verschiedene Netzwerkstrukturen wie ein zentraler Server, lokal verteilte Server, Peer-to-Peer Netze mit und ohne Multicast Unterstützung sowie ein Hybridmodell. Der Vergleich wird auf Grundlage verschiedener Quantitäts- und Qualitätsmerkmale wie z. B. Anzahl der Avatare in Hörweite und Verzögerungsmetriken zwischen den einzelnen Avataren durchgeführt. Auf Grundlage dieses Wissen, beschäftigt sich der zweite Teil der Arbeit mit der konkreten Umsetzung einer Architektur anhand des DICE Projektes. Im speziellen gehe ich hierbei auf zwei Algorithmen, Angular Clustering und Grid Summarization, ein. Diese sollen vor allem 2 Probleme des Serveransatzes lösen. Erstens die große Anzahl an Avataren und den damit verbundenen Rechenaufwand, sowie zweitens die heutzutage immer noch vorhandene Bandbreitenbeschränkung bei den Nutzern. Zum Abschluss möchte ich noch einen kurzen Überblick über die zukünftigen Möglichkeiten für den Einsatz dieser Technologie geben und welche Probleme damit verbunden sind.

1. Einleitung

In den letzten Jahren gab es eine Vielzahl an Fortschritten in der Entwicklung von verteilten Virtuellen Welten, im Verlauf dieser Arbeit als DVEs (Distributed Virtual Enviroment) bezeichnet. Diese wurden vor allem durch den Boom von Spielen wie World of Warcraft und Everquest, aber auch von Gesellschaftssimulationen wie Second Life eingeläutet. Gleichzeitig wurde diese Entwicklung aber erst durch die rasanten Verbesserungen im Hardware Bereich ermöglicht. Die Leistungen und Geschwindigkeiten bei der Darstellung von 3D-Objekten, sowie die inzwischen zum Standard gehörende Präsenz von 3D Soundkarten lassen den Nutzer (Spieler) erst richtig in die Welt eintauchen. Doch ein wichtiger Aspekt der realen Welt wird bis heute, wenn überhaupt, nur sehr rudimentär behandelt. Die Rede ist von der zwischenmenschlichen Kommunikation. Wir sind es im Alltag gewohnt, dass wir z.B. bei der Arbeit unser Frühstücksradio hören können und uns zur gleichen Zeit mit einem Kollegen über den heutigen Tag abstimmen, während womöglich der Chef gerade seinen Frust an der Putzfrau entlädt. In einer DVE wäre so etwas heutzutage unmöglich, da wenn überhaupt eine Sprachkommunikation möglich ist, diese nur eine Art Funkverbindung ist. Die Nutzer verbinden sich hierbei zu einem Server der einerseits als Sammler und Verteiler der Audiostreams dient und andererseits Räume für eine bestimmte Anzahl Leute bietet. Das geschieht in der Regel durch extra Software wie Teamspeak oder Ventrilo, sowie selten auch durch im Spiel integrierte Sprachkommunikation wie z. B. bei Counterstrike oder World of Warcraft. Fängt jetzt einer der Teilnehmer im Raum an zu sprechen, wird sein Audiostream zum Server geschickt, dort kopiert und einfach an alle im Raum weitergeleitet. Das führt natürlich zu erheblichen Problemen bei der Kommunikation, da immer nur einer der Anwesenden verstanden wird. Sprechen mehrere Leute gleichzeitig, überlagern sich die Audiostreams und es kommt nur noch ein unverständliches Durcheinander an.
Das ist der Punkt, an dem Immersive Voice Communication ansetzt. Das Ziel ist es, ein realistisches Klangumfeld für alle Avatare in einer DVE zu erzeugen. Dafür muss jeder Avatar mit einer auf ihn persönlich zugeschnittenen Mischung von Sprache und Hintergrundgeräuschen versorgt werden. Zusätzlich müssen, um es realistisch klingen zu lassen, Informationen über die Richtung und den Abstand aus der Geräusche kommen, die Lautstärke sowie die Einflüsse der Umgebung auf den Klang geliefert werden. Dieser Service wird immersive audio communication Service [1] genannt.
Einen solchen IAC Service für DVEs anzubieten führt zu einem hohen Bedarf an Rechenleistung

und Netzwerkressourcen. Denn der Audiostream muss in Echtzeit für jeden Avatar aus allen Klangquellen im Bezug auf seine Umgebung, inklusive Sprache der anderen Avatare in seiner Reichweite, erzeugt werden. Außerdem sorgt die ständige Bewegung der Avatare in einem DVE für die Notwendigkeit zur Änderung der Audiostreams für eine Vielzahl von Avataren. Welche Netzwerkstrukturen für einen IAC Service in Frage kommen klären wir in Kapitel 2 dieser Arbeit.
Der Rest der Arbeit beschäftigt sich mit einem Dense Immersive Communication Enviroment (DICE) bei dem es sich um einen IAC Service, der auf dem Client-Server Prinzip aufbaut, handelt. Das System wurde an der Universität Wollongong in Australien entwickelt und die Testergebnisse in [2] veröffentlicht.

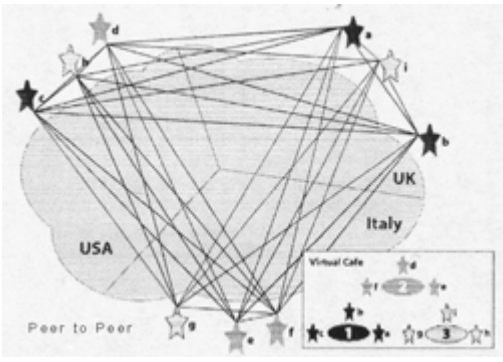
2 Netzwerkstrukturen

2.1 DVE Modell und Qualitätsmetriken

Um den Vergleich der verschiedenen Ansätze zu vereinfachen gehen wir von einem einfachen DVE Modell aus, einem virtuellen Cafe. Dieses besteht nur aus 3 Tischen, an denen Platz für jeweils 3 Gäste ist. Hierbei stellen die Tische immer die so genannte Interactive Zone der Gäste dar, also den Bereich in dem ein Gast aktiv zuhört und/oder spricht. Die 2 anderen Tische werden für diesen Gast als seine Background Zone bezeichnet. Das bedeutet, diese erzeugen zwar Geräusche, aber die Qualität spielt für den Gast hier eine untergeordnete Rolle, da es ihn nicht interessiert. Die 9 Gäste in diesem Cafe sind über verschiedene Länder verteilt und können sich hinsetzen wo sie wollen. Dies wird in der Simulation mit einer Korrelation zwischen 0 und 1 bezeichnet. Dabei bedeutet 1, dass die Leute an einem Tisch auch geographisch benachbart sind, und 0 bedeutet, dass sie aus verschiedenen Ländern kommen. Es gibt zwei wichtige Vergleichsmerkmale für die Qualität eines IACs. Erstens das Interactive Delay, das sich aus der durchschnittlichen Verzögerung zwischen dem Absenden und dem Eintreffen der Audiostreams aller Gäste an seinem Tisch ergibt. Zweitens die Scene Accuracy. Um diese zu bestimmen, muss man davon ausgehen, dass das Delay für Klangquellen die sich in großer Entfernung befinden, höher werden darf. Man berechnet also $A_{ij}=\epsilon+\frac{x_{ij}}{v}$, wobei ϵ eine Fehlerschranke ist, v die Schallgeschwindigkeit und x_{ij} der Abstand von i zu j in Metern ist. Die Scene Accuracy ist dann der prozentuale Anteil der Zuhörer/Quelle Paare (i,j) der virtuellen Welt, bei denen A_{ij} größer ist als das Delay zwischen Zuhörer und Quelle. Auf Grundlage dieser beiden Metriken vergleiche ich in den nächsten Kapiteln verschiedene Ansätze für einen IAC Service.

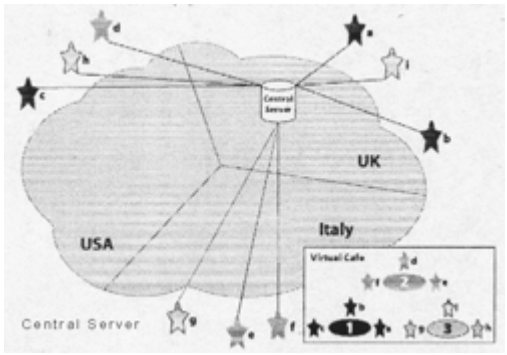
2.2 Peer-to-Peer

Bei einem Peer-to-Peer Ansatz sind alle Gäste direkt miteinander verbunden. Dadurch bekommt jeder Gast einen direkten Audiostream von allen anderen Gästen. Damit ist er in der Lage an seinem eigenen PC eine realistisches Klangfeld für seinen Avatar aus den Streams der anderen zu berechnen. Darin liegt sowohl der Vorteil, als auch der Nachteil dieses Ansatzes. Der Vorteil ist natürlich, dass man die „billige“ Rechenleistung der Clients nutzen kann. Je nach Ausprägung der virtuellen Welt kann das aber auch zu einer Überlastung eines oder mehrerer Clients führen. Wenn sich nämlich zu viele Avatare in seiner Umgebung aufhalten, steigt der



Rechenaufwand linear mit der Anzahl an Avataren in der Umgebung. Dies führt zu Verzögerungen in der Klangberechnung und somit wird der Nutzer aus der virtuellen Welt gerissen, da es sich nicht mehr real anfühlt. Betrachtet man allerdings das Interactive Delay kann sich der Peer-to-Peer Ansatz zusammen mit dem in 2.6 beschriebenen Hybridmodell an die Spitze setzen. Dabei lässt sich beobachten, dass der bei einer Korrelation von 0 schon sehr gute Wert von 125ms sich bei einer Korrelation von 1 auf 25ms senken lässt.
Einer weiterer wichtiger Punkte bei Peer-to-Peer Netzen ist die Bandbreite, da Clientrechner meistens nur mit ADSL angebunden sind. Für den Fall, dass wir Unicastverbindungen zwischen den Clients verwenden, skaliert der Bandbreitenverbrauch mit P * m, wobei P die Anzahl aller Nutzer ist und m der Durchschnitt der Avatare in der Interactive und Background Zone. Das wird vor allem für große P ein Problem, selbst wenn die Bandbreitennutzung für eine Verbindung minimal ist. Man geht bei heutigen Spielen von bis zu 10.000 Teilnehmern und mehr aus. Und selbst eine Umstellung von Unicast auf Multicast sorgt hierbei für eine geringe Reduzierung des Verbrauchs auf $P * m^{0.8}$. Allerdings kommt gleichzeitig ein Zusatzaufwand für die Verwaltung der Multicast Bäume hinzu. Die Simulationen haben gezeigt, dass sich, gerade wenn sich viele Background Zonen der Avatare überschneiden, der Zusatzaufwand beim Aktualisieren der Bäume bei $m^{0.2}$ bewegt. Ein weiteres Problem beim Peer-to-Peer Ansatz, bezieht sich eher auf die Art des DVE. Da diese Arbeit aber über Immersive Gameing geht führe ich es hier noch auf. Durch die Funktionsweise von Peer-to-Peer Netzen ist es schwierig die Privatsphäre der Spieler zu bewahren und zu verhindern, dass Leute die Daten zu ihrem Vorteil manipulieren (cheaten).

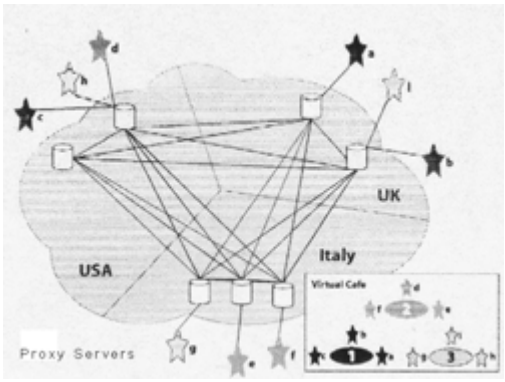
2.3 Central Server



Bei diesem Ansatz sind natürlich alle Clients mit dem zentralen Server verbunden. Dadurch kann man die Bandbreite in der Regel außen vor lassen, da die Server meist über eine schnelle Anbindung verfügen und die Clients ja nur wenig Bandbreite für das Senden ihrer Daten brauchen. Der Empfang von Daten ist bei ADSL ja wesentlich besser möglich, sollte also auch keine Probleme aufwerfen. Auch dem am Ende von 2.2 erwähnten Sicherheitsproblem kann man bei einem zentralen Server leicht entgegenwirken. Zusätzlich lässt sich auch sehr leicht ein Zahlungssystem einführen, was vor allem für kommerzielle Produkte wie Spiele wichtig ist. Die Problemstellen ergeben sich hier aus der Positionierung des Servers und dem Fakt, dass die Skalierbarkeit durch die Leistung des Servers begrenzt wird. Der Rechenaufwand wächst hierbei mit P^2 . Es gibt zwar Möglichkeiten, ein paar Berechnungen wiederzuverwenden und Avatare, die sich auf engem Raum befinden, zusammen zu berechnen, aber insgesamt bleibt der Server ein Flaschenhals für die Skalierbarkeit. Die Position des Servers in der realen Welt hingegen wirkt sich natürlich direkt auf das Interactive Delay aus. Dies geschieht gleich auf doppelte Weise: 1. Muss der Audiostream, nachdem er beim Server angekommen ist, ja erst für die Berechnung verwendet werden, was natürlich auch Zeit kostet. 2. Müssen die Informationen den doppelten Weg zurücklegen, da sie erst zum Server und dann wieder zurück geschickt werden. Man kann diesem Problem dadurch begegnen, indem man Server optimal positioniert. Dafür platziert man den Server in der Nähe der meistens Clients. Hierbei gilt es zu beachten, dass dieser optimale Platz sich zur Laufzeit ändern kann. Nämlich dadurch, dass sich meistens nicht alle Nutzer in der selben Zeitzone befinden und daher zu verschiedenen Zeiten in der virtuellen Welt aktiv sind. Dagegen haben Ortswechsel in der virtuellen Welt keinen Einfluss auf diese Position. Diese Ergebnisse konnte man auch aus den durchgeführten

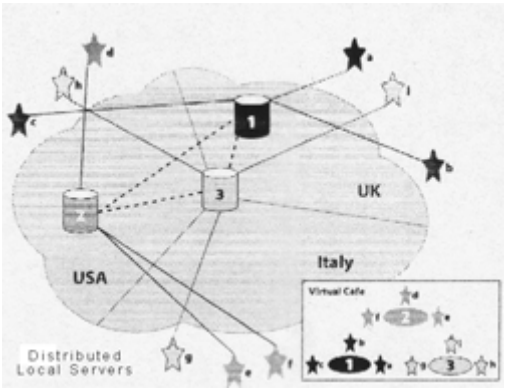
Simulationen ableiten. Für das Interactive Delay wurden 300ms für einen zentralen Server gemessen und nur 150ms wenn man diesen optimal positioniert. Desweiteren konnte man feststellen, dass natürlich die Korrelation keine Rolle spielt, da sich ja alle Nutzer zum Server verbinden müssen. Betrachtet man jetzt die Scene Accuracy, erkennt man sofort, dass bei diesem Ansatz die Positionierung der ausschlaggebende Faktor ist. Während man für den optimalen Server ca. 40% Genauigkeit hat, schafft es ein zufällig platzierter Server nur auf schlechte 10%.

2.4 Distributed Proxies



Bei diesem Ansatz werden die Proxie Server in der Nähe großer Nutzergruppen aufgestellt. Sie dienen hierbei dazu, die Audiostreams für die bei ihnen angebundenen Clients zu empfangen und die Berechnungen für das Klangbild an ihrer Stelle durchzuführen. Gleichzeitig leiten sie natürlich die von ihren Clients erzeugten Audiostreams zu den anderen Proxies weiter. Damit handelt es sich im Grunde nur um einen Peer-to-Peer Ansatz, bei dem die Proxies die Rolle der Clients übernehmen. Der grundlegende Unterschied hierbei ist dass man das Bandbreitenproblem eliminiert hat, da die Proxieserver untereinander viel schneller angebunden werden können wie es bei Heimrechnern als Client der Fall wäre. Auch die Möglichkeiten, Manipulationen zu überwachen und ein Zahlungssystem einzuführen sind durch die Kontrolle über die Proxieserver gegeben. Im Gegenzug hat man allerdings so gut wie keine Möglichkeit Streams zusammenzufassen, wenn es keine oder nur geringe Korrelation gibt. Im Bezug auf die beiden Metriken Interactive Delay und Scene Accuracy verhält sich ein solcher Ansatz wie reines Peer-to-Peer.

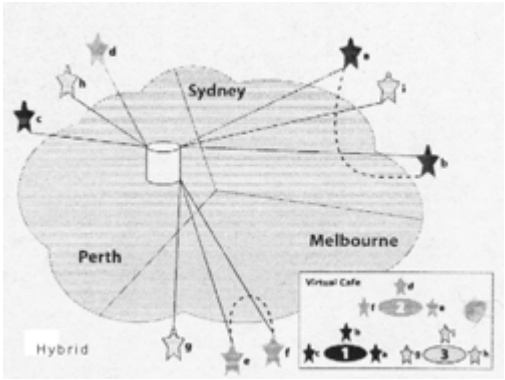
2.5 Distributed Local Server



Für diesen Ansatz muss man die virtuelle Welt in einzelne Bereiche unterteilen. In unserem Beispiel könnte man also jeden Tisch als einen Bereich wählen. Diese Bereiche werden dann von jeweils einem Server verwaltet, die sich beispielsweise in jedem Land befinden. Ein solcher Server kann natürlich auch mehrere Bereiche verwalten wenn es notwendig ist. Damit lässt sich der Ansatz wieder auf eine Art zentralen Server Ansatz zurückführen, bei dem die Bereichsserver nur einen Teil der virtuellen Welt verwalten. Jeder der Bereichsserver berechnet hierbei die Audiostreams für die angeschlossenen Clients und kann gleichzeitig aus den ankommenden Audiostreams den Background für die anderen Bereichsserver erzeugen. Damit lässt sich die Bandbreitennutzung verringern. Die kritischen Überlegungen für diesen Ansatz ergeben sich hierbei zu einem Teil aus der Position der Bereichsserver und zum anderen Teil aus Veränderungen in der virtuellen Welt. Wenn zum Beispiel einer der Gäste von Tisch 2 zu Tisch 1 wechselt, muss sein Audiostream von Bereichsserver 2 zu Bereichsserver 1 umgeleitet werden. Das kann erhebliche Auswirkungen auf das Delay haben, je nachdem wie weit der Bereichsserver 1 vom Clientrechner des Gastes entfernt ist. Der Aufwand für das Umrouten ist aber nicht so hoch wie beim Multicastproblem des Peer-to-Peer Ansatzes, da ja

nur ein Client oder Multicastbaum betroffen ist. Das Problem mit der richtigen Position kann man wie in 2.3 lösen, allerdings haben hier die Bewegungen in der virtuellen Welt Einfluss auf die optimale Position der Bereichsserver. Um diesen Änderungen nachzukommen, muss man sich Gedanken über die Neuverteilung der Bereiche auf die Bereichsserver während der Laufzeit machen. Betrachtet man die Simulationsergebnisse für das Interactive Delay und die Scene Accuracy, befindet sich dieser Ansatz mit 175ms und 40% bei 0 Korrelation auf gleicher Höhe wie ein zentraler Server. Doch während die Korrelation auf den zentralen Server keinen Einfluss hatte, sinkt das Delay auf 25ms und die Genauigkeit steigt auf nahezu 100% bei einer Korrelation von 1.

2.6 Hybrid Modell



Eine Möglichkeit für einen solchen Hybrid Ansatz ist es, sowohl Peer-to-Peer als auch Central Server Mechanismen zu verwenden. Das Bild zeigt ein mögliches Beispiel solch einer Architektur. Der zentrale Server berechnet hierbei alle Audiostreams der Background Zonen und so viele der Interaktive Zone wie möglich. Peer-to-Peer Mechanismen werden verwendet um Audiostreams der Interaktiven Zone direkt zu verteilen, wenn durch den zentralen Server ein zu großes Delay erzeugt würde. Im Blick auf die Metriken verhält sich dieser Ansatz beim Interative Delay daher wie reines Peer-to-Peer mit 125ms bei 0 Korrelation und nur 25ms bei einer Korrelation von 1. Die Scene Accuracy richtet sich nach dem Zentralen Server der zur Berechnung der Hintergrund Zonen verwendet wird, also ca. 40%.

2.7 Schlussfolgerung

Farzard Safei hat mehrere Ansätze um einen IAC Service in einer DVE anzubieten, vorgestellt und sie anhand von Metriken für Klangqualität sowie ihres Bedarfs an Rechenaufwand und Bandbreitenbedarf analysiert. Die Ergebnisse haben gezeigt, dass unterschiedliche Faktoren, die sich auch zwischen den Ansätzen unterscheiden, wie zum Beispiel: Art der virtuellen Welt, Häufigkeit der Bewegungen in der Welt, Position der Server sowie Anzahl der beteiligten Avatare diese stark beeinflussen. Besonders Peer-to-Peer hat große Probleme zu skalieren, selbst wenn eine Unterstützung für Multicast besteht. Beim Ansatz mit einem zentralen Server oder verteilten Bereichsservern kommt es auf die Position dieser an. Dieses Problem scheint aber leichter zu lösen, daher beschäftige ich mich im nächsten Kapitel dieser Arbeit mit der Umsetzung des Distributed Local Server Ansatzes, da dieser noch den Vorteil der hohen Scene Accuracy von fast 100% bei einer Korrelation von 1 gegenüber eines zentralen Servers hat.

3 DICE

3.1 Anforderungen und Systemarchitektur

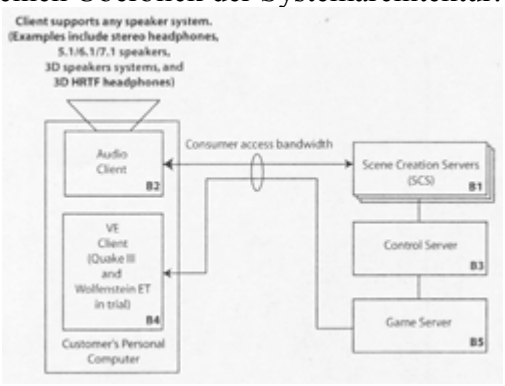
Der DICE Service sollte in erster Linie mit dem Ziel entwickelt werden, natürliche Kommunikation in großen virtuellen Welten zu ermöglichen. Die Nutzer verbinden sich dabei von überall auf der Welt per Internet zum DICE.

Dieses Ziel stellt eine Reihe von Designanforderungen auf, die bei der Entwicklung beachtet werden mussten.

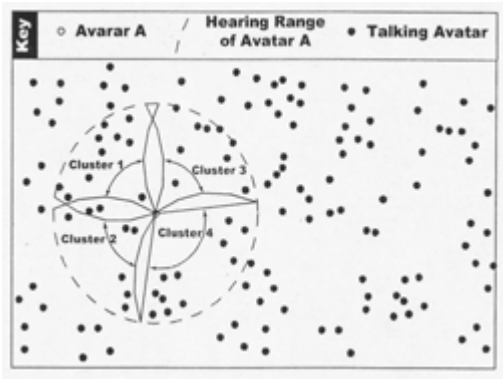
1. Skalierbarkeit auf große virtuelle Welten mit einer Vielzahl an Avataren
2. Skalierbarkeit auch in dicht bevölkerten Bereichen der virtuellen Welt
3. Funktionsfähigkeit auch bei großer geographischer Streuung der Nutzer
4. Behandlung des Bandbreitenproblems

Zusätzlich wurde bei der Entwicklung wieder davon ausgegangen, dass eine gewisse Toleranz in der Genauigkeit des Klangbildes herrscht, wenn die Klangquellen sich nicht in der unmittelbaren Umgebung des Avatares befinden.

Die folgende Abbildung zeigt einen Überblick der Systemarchitektur:



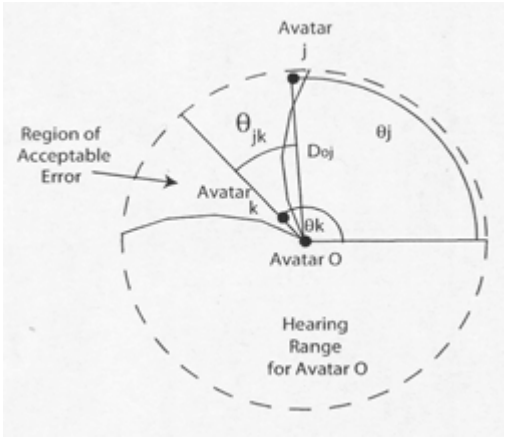
Wie man sofort erkennt, wird die Arbeit zwischen dem Client PC und einem oder mehrerer Scene Creation Servers (SCS) aufgeteilt. Als Szene ist bei den SCS nur der für die Kommunikation wichtige Teil der Umgebung gemeint. Die optische Darstellung wird komplett vom VE Client/Server übernommen, bei dem es sich in diesem Fall um die Spiele Quake 3 oder Wolfenstein ET handelt. Einfach ausgedrückt, werden in den SCS Vereinfachungen der Audioszene berechnet und das Ergebnis dann mit niedrigem Bandbreitenbedarf an die Clients weitergeleitet.
Im Detail bedeutet das: Der Audioclient (B2) des Nutzers sendet einen einzelnen Audiostream, in der Regel eine Mikrofonaufnahme, zu einem oder mehreren SCS. Im Gegenzug erhält der Audioclient eine feste Anzahl k an Audiostreams vom SCS. Durch k und dem im nächsten Abschnitt beschriebenen Angular Clustering Algorithm, lässt sich Punkt 4 in den Griff bekommen. Zusätzlich enthalten die Audiostreams der SCS Meta-Daten der VE Server (B5), mit deren Hilfe die Clients den Raumklang auf ihrer Hardware erzeugen können und ihn über die angeschlossene Peripherie ausgeben. Man überlegt sich leicht, dass wenn k genügend groß ist, die SCS nur die Aufgabe haben die Streams der anderen weiterzuleiten, entweder per Uni- oder Multicast. In der konkreten Implementierung wurde k=4 gewählt, um eine gute Antwortzeit der Anwendung zu ermöglichen, auch wenn der Großteil der Nutzer nur über ADSL verfügt. Dabei sollte man immer im Hinterkopf haben dass ein Großteil der vorhandenen Bandbreite schon von der VE verbraucht wird. Diese Audiostreams werden aufgrund des in 3.1 vorgestellten Algorithmus auch Angular Cluster Summaries genannt. Jedes dieser Cluster stellt dabei eine gewichtete Mischung aller Audiosignale aller Avatare eines Segmentes in Hörweite des Clienten dar (siehe Abbildung 2). Die Gewichtung der Audiodaten ist dabei extrem wichtig, da so die



unterschiedlichen Lautstärken für die Klangquellen berechnet werden können. Über die genaue Zusammensetzung entscheidet der Kontrollserver (B3). Erstens nutzt er die Positionsangaben aus dem VE Server um den zuständigen SCS für diesen Avatar zu zuweisen und zweitens berechnet er aus diesen Daten das Massezentrum für jeden der Streams an diesen Nutzer. Im nächsten Abschnitt wird klar was dieses Massezentrum ist und wofür es verwendet wird.

3.2 Angular Clustering

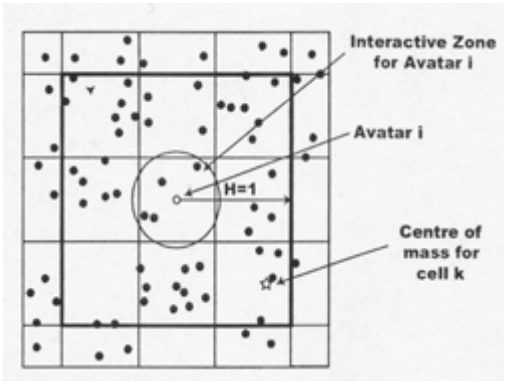
Wie schon im letzten Abschnitt beschrieben, senden die SCS k, im konkreten Fall also 4, unabhängige Audiostreams an die Clients. Jeder der Audiostreams ist eine gewichtete Mischung aller Klangquellen um den Avatar in diesem Segment. Dabei werden die Klangquellen ausgehend von dem als Massezentrum bezeichneten Punkt im Raum platziert. Hierbei kommt die am Anfang des Kapitels angesprochene Toleranz zum tragen. Da man von einem Massezentrum ausgeht und nicht von den genauen Positionen der Avatare, führt man sowohl Fehler in der Entfernung der Klangquellen wie auch der Richtung aus der sie kommen ein. Um der Annahme, dass entfernte Klangquellen weniger genau dargestellt werden müssen gerecht zu werden, wird eine Technik verwendet, die sich Distance Weighted Clustering nennt. Das funktioniert auf folgende Weise: Im ersten Schritt werden m Interactive Zone Cluster erstellt, und zwar für die m nächstgelegenen Avatare in diesem Segment. Dabei gilt immer $m \leqslant k$. In einem zweiten Schritt werden dann $k-m$ Background Cluster Zones erstellt, um die entstandenen Lücken im Segment zu füllen. Eine Interactive Zone Cluster, wird dabei ganz einfach erzeugt. Der Algorithmus wählt sich den Avatar mit geringstem Abstand zum Avatar des Clients aus, und macht ihn zum Massezentrum dieses Clusters. Die Abbildung 1 verdeutlicht das weitere Vorgehen. Sei Avatar A_k der mit dem geringsten Abstand zum Avatar des Clients, also wird ein Cluster mit ihm als Massezentrum erzeugt. Als nächster Schritt wird geschaut, welche anderen Avatare man diesem Cluster noch zuordnen kann. Betrachten wir also den Avatar A_j . Wie man sieht, hat dieser Avatar einen gewissen Abstand D_{oj} von unserem Client Avatar und befindet sich innerhalb eines Winkels θ_{jk} ausgehend von der Geraden durch A_j und unserem Client-Avatar. Der Avatar A_j gehört jetzt zu diesem Cluster wenn der Winkel θ_{jk} kleiner ist als der Schwellenwert ε_{oj} . Dieser Schwellenwert ist eine mathematische Übersetzung der Toleranzannahme. Es gilt $\varepsilon_{oj} = E_{min} + E_{max} \cdot \frac{D_{oj}}{D_{max}}$, wobei E_{min} und E_{max} auf den gewünschten Detailgrad justiert werden können und D_{max} die Hörweite des Clientavatares ist. Mit dem Schwellenwert wird also das mit Region of acceptable Error bezeichnete Gebiet abgedeckt und alle Avatare, die sich in diesem Gebiet befinden werden diesem Cluster zugeordnet. Nachdem man diesen Schritt m Mal durchgeführt hat, wählt man für die restlichen k-m Cluster den Punkt als Massezentrum für ein neues Cluster, der den größten Abstand zu den Region of acceptable Error der Interactive Zone Cluster hat. Diesen Background Clustern werden diejenigen Avatare zugeordnet, deren Winkel zur Geraden durch Massezentrum und Clientavatar am kleinsten ist. Nach Abschluss dieses Teilschrittes, hat der Algorithmus also genau k Cluster erzeugt, die dann an den Client übertragen werden. Allen die sich Gedanken über die Ausrichtung des Clientavatares im Spiel machen, sei gesagt, da es sich um eine punktförmige Audioquelle handelt, kann man das Ergebnis einfach im Client in die richtige Richtung bringen. Ein Problem kann allerdings durch den



proportional zu der Anzahl der Avatare ansteigenden Rechenaufwand in den SCS auftauchen. Um auch diesem Problem entgegen zu wirken nutzt das DICE einen weiteren Algorithmus, der sich Grid Summarization nennt, ein. Auf diesen werde ich im nächsten Abschnitt nochmal genauer eingehen, bevor ich am Ende dieser Arbeit noch kurz auf erste Tests und Reaktionen des Systems eingehe.

3.3 Grid Summarization

Dieses Verfahren dient im Wesentlichen dazu, die möglicherweise riesige Anzahl an Avataren um einen Nutzer zu reduzieren. Damit müssen die SCS weniger Avatare beim Angular Clustering berücksichtigen. Das Vorgehen ist hierbei wieder ganz einfach. Man teilt die virtuelle Welt in Zellen ein, wobei man durch die Größe der Zellen direkt die Genauigkeit und verbrauchte Rechenleistung beeinflussen kann.



Das Bild zeigt eine Aufteilung der virtuellen Welt in der jeder Avatar genau 9 Zellen hören kann. Dabei hat jeder Avatar eine Heimatzelle, in der er sich gerade befindet. Für jede Zelle der Welt berechnen die SCS jetzt ein weiteres Massezentrum, dass sich aus dem Durchschnitt der Avatarpositionen innerhalb der Zelle ergibt. Wenn jetzt die Cluster für den Avatar i berechnet werden sollen, wird in seiner Heimatzelle das Angular Clustering wie in 3.2 beschrieben durchgeführt. Für die anderen acht Zellen, die sich in Hörweite befinden, wird anstatt den genauen Positionen das Massezentrum dieser Zellen als einziger Avatar verwendet. Als Audiostream wird dabei ein einfacher Mix aus allen Klangquellen dieser Zelle verwendet. Dadurch wird die Anzahl der am Clustering beteiligten Avatare teilweise drastisch reduziert. Dies findet natürlich auf Kosten der Genauigkeit in der räumlichen Darstellung der Klänge statt. Simulationen mit einer vereinfachten virtuellen Welt haben dabei folgende Ergebnisse gebracht. Bei einer Hörweite von 9 Feldern, konnte die Anzahl an notwendigen Operationen bei 100 Avataren die sich in Hörweite befinden auf nur 1/5 der Operationen die ohne Grid Summerization nötig wäre gedrückt werden. Allerdings hat das natürlich auch Auswirkung auf die richtige Position der Avatare im Bezug auf die im letzten Abschnitt beschriebene Region of acceptable Error. In Zahlen führte es dazu, dass 25% der Avatare nicht mehr einem der Interactive Zone Cluster angehörten.

3.3 Zahlen und Fakten zum realen System

Das DICE wurde als Distributed Local Server Ansatz innerhalb Australiens umgesetzt. Als SCS Server wurden 3, auf einzelnen 1.2GHZ Celeron CPUs basierende Rechner verwendet. Als VE wurde eine durch eine Modifikation veränderte Version von Quake 3 verwendet. Die Modifikation dient dabei nur dem Auslesen der Positionen der einzelnen Avatare innerhalb der Welt sowie ein paar Statusinformationen, wie z.B. ob der Avatar überhaupt am Leben ist. Die in den letzten beiden Abschnitten vorgestellten Techniken sind vollständig implementiert. Die verwendete virtuelle Welt ist durch die Beschränkung auf 32 gleichzeitige Nutzer nicht ganz ideal, aber es gibt zur Zeit leider keine Welten die mehr Spieler unterstützen und trotzdem kostenfrei sind. Interessant in diesem Zusammenhang ist, das Auswertungen der Serverbelastung andeuten, dass ein SCS Server leicht bis zu 100 Spieler unterstützen könnte. Vor allem bei dieser doch eher alten Hardware eine beachtliche Menge.

4 Ausblick und abschließende Worte

Ein weiteres interessantes Ergebnis brachten in diesem Zusammenhang die Eindrücke verschiedener Testgruppen am System ans Licht. Vor allem die Hardcore Gamer waren stark von solch einem System angetan, da sie natürliche Kommunikation zwischen den Spielern auch über lange Strecken ermöglicht. Außerdem wird der Realismus im Spiel gefördert. Schleicht sich zum Beispiel einer der Teammitglieder an einen Gegner heran, kann er die gegnerische Taktik belauschen, muss aber gleichzeitig darauf achten, dass er sich nicht durch seine Stimme verrät. In diesem Zusammenhang haben sich viele Spieler aber zusätzlich ein integriertes Walkie-Talkie gewünscht, dass auch weit entfernte Mitspieler informieren kann. Allerdings muss auch bei dessen Benutzung notfalls geflüstert werden, um einer Entdeckung durch den Feind zu entgehen.

Die Themen dieser Arbeit haben gezeigt, dass man auch heute schon, wenn man den richtigen Ansatz für die passende Situation wählt ein sehr viel realistischeres virtuelles Erlebnis erzeugen kann als es zur Zeit der Fall ist. Greift man dabei noch auf die in Kapitel 3 vorgestellten Algorithmen zurück, ist es sogar bei einer geringen Bandbreite möglich die Leute mit einer realistischen Klangumgebung zu beglücken. Doch wie jede neue Technologie wird sich auch IAC erst durchsetzen können, wenn es erste kommerzielle Erfolge gibt. Um dieses Ziel zu erreichen arbeiten die DICE Entwickler Paul Boustead, Farzad Safaei und Mehran Dowlatshahi mit einem australischen Telekommunikationsunternehmen zusammen um weitere Ergebnisse bei mehr als 32 Spielern zu sammeln. Und auch wenn sich diese Arbeit auf Onlinespiele bezieht, lässt sich dieser IAC Service auch für andere Szenarien wie zum Beispiel eine Online Universität einsetzen. Man muss ja nur die virtuelle Welt an die neuen Gegebenheiten anpassen.

Quellen

Sämtliche Grafiken sind stammen aus:

[1] Paul Boustead und Farzard Safei.
Comparison of delivery architectures for immersive Audio in Crowded Networked Games.
NOSSDAV'04, 16-19 Juni 2004, Cork Irland.

[2] Paul Boustead, Farzard Safei und Mehran Dowlatshahi.
DICE: Internet Delivery of Immersive Voice Communication for Crowded Virtual Spaces.
Veröffentlichungen zur IEEE Virtual Reality 12-16 März 2005, Bonn Deutschland