Video in Vorlesungsaufzeichnungen mit informatikfernen Inhalten am Beispiel Archäologie

Th. Ottmann*, R. Müller*, G. Seitz†, Ch. Steinert*

* Institut für Informatik
†Abteilung für Provinzialrömische Archäologie
Universität Freiburg

Zusammenfassung

Wir berichten über eine langjährige Zusammenarbeit zwischen Informatik und Provinzialrömischer Archäologie, die sowohl Rückwirkungen auf die archäologische Wahrheitsfindung hatte als auch zu neuen Herausforderungen an die Informatik geführt hat. Konkret wurden ein Ausgrabungsfund (die *Villa Urbana* in Heitersheim) mit Mitteln der Computergraphik modelliert, ein ca. 5–minütiger Film eines virtuellen Rundganges durch den Gebäudekomplex berechnet, Bilder und Film in eine Telepräsentation integriert und diese nach dem "Note–Taking–Verfahren" in ein multimediales Dokument verwandelt. Es zeigt sich, daß für eine sinnvolle Nutzung neuer Medien in der Archäologie nicht nur Standardtechniken der Informatik ausreichen, sondern insbesondere im Bereich der Synchronisation von Medienströmen neue Techniken benötigt werden.

1 Einleitung

Ziel der archäologischen Forschung ist es, uns heute ein möglichst umfassendes Bild vergangener Kulturepochen zu vermitteln. Wir zeigen in diesem Aufsatz an einem konkreten Beispiel (der *Villa Urbana* in Heitersheim bei Freiburg), was das bedeutet und wie mit Methoden und Werkzeugen der Informatik ein Beitrag zur Erfüllung dieses Zieles geleistet werden kann. Dabei wird einerseits auf bekannte Techniken zurückgegriffen. Aber es wird auch klar, daß neue Informatik-Lösungen gebraucht werden, wenn neue Medien in Forschung und Lehre in der Archäologie sinnvoll genutzt werden sollen.

Die traditionelle archäologische Arbeitsweise gliedert sich in die grundlegenden Bereiche Ausgrabung (Datengewinnung), Auswertung (Datenbewertung), Publikation (Datenaufbereitung) und Präsentation (Datendarstellung). Die Zusammenarbeit zwischen Archäologie und Informatik, über die hier berichtet wird, bezog sich fast ausschließlich auf den Bereich *Präsentation*. Anstelle einer Rekonstruktion des Fundobjektes durch eine technische Zeichnung sollte ein *realitätsnäheres* Computermodell erstellt werden.

Während es heute bei der Planung und beim Bau neuer Gebäude vielfach üblich ist, vorab ein computeranimiertes 3–d–Modell zu erstellen, das einen Blick in die später realisierte Zukunft erlaubt, gehört der entsprechende Blick in die Vergangenheit mit Hilfe virtueller Realität noch nicht zur allgemein üblichen Form der Darstellung archäologischer Forschungsergebnisse. Ausgehend von dem eher spielerischen Wunsch, ein für ein mächtiges 3–d–Modellierungswerkzeug geeignetes Übungsobjekt zu finden, sind so über mehrere Jahre hinweg ein immer wieder revidiertes und mit Hilfe eines professionellen Werkzeugs (Softimage, [13]) erstelltes 3–d–Modell der Villa Urbana in Heitersheim und zwei Filme von je etwa 5 Minuten Länge entstanden, die als virtuelle Rundgänge durch das Gebäude einen der historischen Realität möglichst nahe kommenden Eindruck vermitteln.

Die Filme ebenso wie eine Reihe von Einzelbildern und vielleicht sogar das 3–d–Modell selbst sollen in einer den Ausgrabungsbefund dokumentierenden, permanenten Ausstellung in einem Schutzhaus über der Ausgrabungsstätte integriert werden. Darüberhinaus wurden Bildmaterial und Film für eine im Rahmen des VIROR-Verbundprojekts [14] durchgeführte und nach dem AOF-Verfahren [8] aufgezeichnete Telepräsentation verwendet. Dabei zeigte sich, daß die Nutzung solcher neuen Medien in der archäologischen

Lehre von den zur Zeit verfügbaren Werkzeugen nur unzureichend unterstützt wird. Wir skizzieren im folgenden zunächst das Ausgrabungsprojekt selbst. Wir berichten dann über die Modellierung und Filmberechnung und berichten schließlich über die in einem konkreten „Lehrexperiment“ (Telepräsentation eines Archäologen) gewonnene Erfahrung. Wir gehen insbesondere auf die Frage ein, welche Anforderungen an die Annotierbarkeit von Video und die Synchronisation solcher oder ähnlicher Medienströme resultieren.

Abbildung 1: Grundriss und Sichten auf die Villa Urbana

2 Das Ausgrabungsprojekt

Als ein wichtiger Schritt zur Klärung des Wirkens der Römer am Oberrhein hat sich der Befund einer Villa Urbana in Heitersheim (ca. 20 km südlich von Freiburg gelegen) erwiesen. Dieses Ausgrabungsprojekt wird von der Abteilung für Provinzialrömische Archäologie der Universität Freiburg wissenschaftlich betreut.

Die Ausgrabungen in Heitersheim haben Reste eines bislang einzigartigen Villenkomplexes als Zeugnis römischer Besiedlung aus der Zeit zwischen 30 und 260 n. Chr. rechts des Rheins zutage gefördert. Leider sind nur Teile erhalten geblieben, die unter dem ursprünglichen ebenerdigen Geländeniveau liegen, also Grundmauern und Kellerräume. Hinzu kommen zahlreiche Kleinfunde unterschiedlicher Gattungen (Münzen, Schmuck etc.). Dennoch war es möglich, aus den Grabungsergebnissen recht genaue Pläne der verschiedenen Gebäude dieses Villenkomplexes zu erstellen und Rückschlüsse auf die Gebäudeart und Bebauung zu verschiedenen Zeitpunkten zu ziehen (vgl. [10] [11] [12] [6]).

Auf der Basis dieser Pläne und Funde wurde nun bereits vor mehreren Jahren damit begonnen, ein 3–d–Modell des Gebäudekomplexes mit Hilfe des Werkzeuges Softimage zu erstellen. Ziel war und ist es, eine möglichst realistische Vorstellung dieses einzigartigen Kulturdenkmals zu vermitteln. Dazu mußten natürlich viele nicht mehr vorhandene oder unbekannte Details durch Analogieschlüsse aus anderen Funden derselben Zeit ergänzt werden. Während konventionelle Methoden der Darstellung durch Modelle, Zeichnungen, Rekonstruktion im Maßtab 1:1 sehr material- und kostenintensiv sind und eine zu einem bestimmten Zeitpunkt gültige Forschungsmeinung festschreiben, liegt der Vorteil einer Modellierung im Rechner in der größeren Flexibilität. Weil es sich bei dem Heitersheimer Ausgrabungsprojekt um ein noch laufendes Forschungsprojekt handelt, war es immer wieder nötig, das Modell an neue Erkenntnisse anzupassen. Das war zu Beginn des Projektes (im Jahre 1995) so nicht vorhersehbar. Und so wundert es nicht, daß das im ersten Versuch entworfene 3–d–Modell als ad hoc Entwurf den späteren Anforderungen, also den durch Erkenntnisfortschritt notwendigen Anpassungen nicht gerecht wurde. Daher wurde die Modellierung wiederholt, aber von vornherein auf übersichtliche Struktur und Änderungsfreundlichkeit geachtet. Dieses Modell ist nun Schritt für Schritt erweitert und verbessert worden. Es war nicht nur ein interessantes Studienobjekt im Rahmen der Informatikausbildung, um interessierten Studenten ein mächtiges Werkzeug zur 3–d–Modellierung, Verfahren der photorealistischen Computergraphik, virtuelle Realität und außerordentlich rechenintensive Rendering–Verfahren nahe zu bringen. Es hat auch den Archäologen immer wieder neue Einsichten verschafft, weil durch die Teilmodellierung und Einarbeitung von aktuellen Erkenntnissen Widersprüche aufgedeckt oder neue Befunde erst richtig eingeordnet werden konnten. Mitte 1997 lag dann ein erster 5–minütiger Film vor, der bereits eine sehr anschauliche Vorstellung der Heitersheimer Siedlungstelle vermittelt. Wir beschreiben im nächsten Abschnitt den aktuellen Stand der Modellierung, der die Grundlage für die Berechnung eines zweiten Films war, die Festlegung der Kamerafahrt und Animation, das Rendering und den Export der Daten in ein für die Videovorführung geeignetes Format.

3 Modellierung und Filmberechnung

3.1 Modellierung

Als Basis für die Modellierung und für den geplanten virtuellen Besichtigungsrundgang dient ein Plan der älteren Steinbauphase des Hauptgebäudes und der weiteren Gebäude des Villenkomplexes. Ferner lag eine Rekonstruktionszeichnung herkömmlicher Art vor. Daraus wurde ein erstes grobes Modell erstellt, das nur die wichtigsten Maße der Gebäude berücksichtigte. Wände, Dächer, Böden, Säulen usw. wurden als einfache CSG–Modelle realisiert. Viele Einzelheiten blieben zunächst offen, die später, jeweils nach Rücksprache mit den Archäologen, ergänzt wurden. Das betraf beispielsweise die Position und Größe von Fenstern und Türen, Texturen für Wände, Dächer und Böden, die Modellierung der Außenanlagen und die Einbettung des Gesamtkomplexes in die Rheinebene vor dem Hintergrund des Schwarzwaldes. Neben der Objektmodellierung bestimmen auch andere Parameter wie Lichtquellen, Schattenwurf, Reflexionsverhalten und Transparenz von Flächen u. a. darüber, ob schließlich ein realitätsnaher Eindruck entsteht. Auch hier wurden immer wieder Korrekturen vorgenommen. Besonderes Augenmerk wurde auf die Modellierung eines vollständig erhaltenen und inzwischen freigelegten Kellerraumes gelegt. Denn hier besteht die Möglichkeit der Prüfung des Modells an der (noch vorhandenen) Realität.

Das Ergebnis ist ein 3–d–Modell, das aus insgesamt ca. 1.300 Elementen, die als CSG–Modelle erstellt wurden, 150.000 Polygonen, 475 verschiedenen Materialien und 100 Grundtexturen besteht.

Abbildung 2: Einzelbild der Villa Urbana als Ergebnis des gerenderten 3-d-Modells

3.2 Kamerapfad und Animation

Für den virtuellen Rundgang muß nach Abschluß der Modellierung ein “Kamerapfad” festgelegt werden. Er besteht aus einer Linie, auf der die Kamera bewegt werden soll, sowie einer zweiten Kurve, die angibt, worauf der Fokus der Kamera gerichtet werden soll. So wird also einerseits die Bewegung der Kamera im Raum festgelegt als auch bestimmt, wohin sie schaut. Dadurch können für den Betrachter ganz unterschiedliche optische Eindrücke erzeugt werden, die beispielsweise Ruhe oder Hektik im Filmgeschehen vermitteln.

Die Animationen werden mit dem Programm Softimage durch das Festlegen sogenannter Keyframes erstellt. Ein Frame beschreibt den Zustand der Szene zu einem bestimmten Zeitpunkt. Will man ein Objekt animieren, so weist man ihm zu bestimmten Zeitpunkten Keyframes, d.h. etwa die gewünschte Position oder Form an diesen Zeitpunkten zu. Die Berechnung zwischen den Keyframes übernimmt dann das Programm Softimage. Hier kann man dann mit Hilfe von sogenannten FCurves (function curves) vorgeben, wie der Übergang von der einen vorgegebenen Position/Form zur anderen auszusehen hat. Legt man die FCurve z.B. als eine Gerade fest, so geschieht die Transformation linear; das würde beispielsweise eine Bewegung mit einer konstanten Geschwindigkeit erzeugen. Man kann diese Kurven jedoch beliebig verändern, so daß man die Animation abbremsen oder beschleunigen bzw. ganz stoppen oder springen lassen kann.

3.3 Rendering

Für einen Film von 5 Minuten Länge, der mit 25 Bildern pro Sekunde abläuft, müssen 7.500 Einzelbilder berechnet werden. Weil als wichtigste Präsentationsformate für den zu berechnenden Film S-VHS und VHS gefordert waren, wurden die Einzelbilder im PAL–Format, d. h. im Verhältnis 4 zu 3 mit einer

Auflösung von 768x576 und einer Farbtiefe von 24 bit berechnet. Hierzu wurde das Programm Mental-Ray [7], das Teil des Softimage Paketes ist, benutzt. Das Programm Softimage produziert für jedes Bild (Frame) eine Szenenbeschreibung, aus der der Renderer das fertige Bild berechnet. Weil die Berechnung eines einzelnen Bildes ca. 3 bis 5 Minuten Zeit beansprucht, die Einzelbilder aber völlig unabhängig voneinander berechnet werden können, wurden gleichzeitig mehrere Rechner zur Berechnung eingesetzt. Die berechneten Originalbilder wurden im von Softimage gelieferten .pic–Format auf CDROMs archiviert und anschließend nach RGB konvertiert. Das ergab ein Datenvolunen von ca 9,7 GByte.

Es ist klar, daß eine direkte Wiedergabe der Folge der Originalbilder im .pic– oder RGB–Format mit einer Frequenz von 25 fps auch auf sehr leistungsstarken Rechnern mit großem Bild- und Plattenspeicher unmöglich ist. Denn das würde einen Datentransfer von ca. 25 bis 30 MByte pro Sekunde vom Speichermedium auf den Bildwiederholspeicher des Rechners erfordern. Es ist also notwendig, die Originaldaten zu komprimieren. Dabei ist ein angemessener Kompromiß anzustreben zwischen der Darstellungsqualität und dem dafür erforderlichen Datenvolumen. Der Export in das analoge S-VHS-Videoformat erfolgte über ein spezielles Bildplattenmedium. Das Ergebnis ist optisch zwar akzeptabel, erreicht aber nicht die Qualität der Rechnerdarstellung der Originalbildfolge. Es soll daher untersucht werden, ob ein besserer Kompromiß zwischen Datenvolumen und Qualität erreicht werden kann, wenn aus der Originalbildfolge einen MPEG-2-Film berechnet und dieser über einen DVD–Player (und Großbildprojektor) abgespielt wird.

Berechnet wurde bisher nur ein MPEG-1-Film mit einer Auflösung von 320x240 aus den Originalbildern. Das ergab ein Datenvolumen von ca. 40 MByte, ein für die Nutzung zu Lehrzwecken während einer Telepräsentation und zur Ablage auf einem ftp-Server akzeptabler Wert.

Die für die Filmberechnung gewählte Auflösung von 768x576 ist für große Ausdrucke von Standbildern (DIN A4 und größer) zu gering. Daher wurden einige Einzelbilder mit sehr hoher Auflösung (3000x2250) zusätzlich berechnet. Hier zeigte sich, daß für die Auswahl von Einzelbildern (Ansichten) die Möglichkeit des Zugriffs auf sämtliche Originalbilder hilfreich, die Möglichkeiten zur Navigation in der Bildfolge, also die Identifizierung einer Ansicht aber nur unzureichend unterstützt wurden.

4 Telepräsentation und Note–Taking

4.1 Live-Szenario

Am 03.05.99 fand im Rahmen der Ringvorlesung zum Verbundprojekt VIROR [14] ein Vortrag des das Ausgrabungsprojekt betreuenden Archäologen, Prof. H. Nuber, mit dem Titel „Die Römer am Oberrhein“ statt, der an die an VIROR beteiligten Standorte übertragen und aufgezeichnet wurde. Man kann diese Telepräsentation als durchaus typischen Fall eines vor allem auf Bildmaterial und ein Video abgestützten Vortrages ansehen. Im Vortrag wurde insbesondere auf die Heitersheimer Villa Urbana eingegangen. Der Vortrag wurde mit Hilfe eines SmartBoards (interaktiver Großbildschirm) mit Rückprojektion, der MBone Tools *vic* und *vat* [9] für die Bild- und Tonübertragung und des AOFwb [8] als Whiteboard für das Aufblättern und Annotieren der Bilder und Starten und Stoppen des MPEG-Films (s.o.) gehalten. Mit Hilfe der in das AOFwb integrierten Aufzeichnungsmöglichkeit wurde der Vortrag aufgezeichnet. Dieselben Werkzeuge wurden bisher in nahezu 100 ähnlich verlaufenen (Tele–)Präsentationen eingesetzt, die allerdings nahezu ausschließlich Informatikinhalte hatten. In den meisten Fällen wurde für den Vortrag auch nicht, wie in diesem Fall, das SmartBoard benutzt, sondern der Vortrag direkt am Rechner (mit Bildschirm, Tastatur und Maus) gehalten. Es handelte sich hier also um einen Testfall für einen sehr informatikfernen Inhalt, der von einem Geisteswissenschaftler ohne Routine in der Handhabung der für die Telepräsentation benutzten Werkzeuge gehalten wurde.

Wir erläutern nun, welche Besonderheiten gegenüber entsprechenden Telepräsentationen mit Informatikinhalten sichtbar wurden und welche Folgerungen daraus für die Weiterentwicklung der benutzten Werkzeuge zu ziehen sind.

4.2 Benutzungsschnittstelle für den Dozenten

Ein Vortrag, der vom gesprochenen Wort getragen, und von zahlreichen Bildern (wie Dias) unterstützt wird, führt zu sehr wenig Aktivität auf dem Whiteboard. Zwar wäre in vielen Fällen eine direkte Annotation (Beschriftung, Markierung) eines im AOFwb aufgeblätterten Bildes oder die Verwendung eines Zeigers sinnvoll. Jedoch haben ungeübte diese Vortragsart nicht gewohnte Benutzer ganz offensichtliche Schwierigkeiten, die zahlreichen vom AOFwb (ebenso wie von anderen Whiteboards) angebotenen Annotationswerkzeuge zu gebrauchen. Zur Verfügung stehen neben einem Telepointer, Linien, Pfeile, Freihandlinien, Rechtecke, Kreise, Text usw. Die Konzentration auf den verbal vorgetragenen Inhalt macht es aber offensichtlich unmöglich, diese Vielfalt zu nutzen. Gebraucht wurde ausschließlich der Telepointer. Doch selbst dessen Einsatz setzt voraus, daß ein hinreichender Druck auf das Whiteboard ausgeübt wird, um ihn zu aktivieren, was ebenfalls leicht vergessen wird. Auch die Auswahl und das Aufblättern der Bilder von der mit einem Scroll–Balken versehenen Thumbnail-Übersicht am linken Whiteboardrand ist offenbar nicht so intuitiv, wie etwas das Drücken eines „Weiter/Zurück"-Knopfes eines Dia-Projektors. Obwohl die Verwendung des SmartBoards für eine Rechnerpräsentation gegenüber einer Rechnerbedienung mit Bildschirm, Tastatur und Maus dem in diesem Gebiet üblichen und gewohnten Lehrszenario deutlich eher entspricht, zeigte sich doch, daß auch das SmardBoard ohne besondere Anpassung der verwendeten Software noch nicht ideal ist: Die Übertragung der Desktop–Metapher auf das SmartBoard liefert eine Benutzerschnittstelle, die diesem Interaktionsmedium nicht angemessen ist, d. h. Funktionen, die bei Interaktion mit Bildschirm, Tastatur und Maus intuitiv und einfach zu bedienen sind, müssen in anderer Weise realisiert werden. Denn bei einer auf Bildmaterial abgestützten Rechnerpräsentation sind nur ganz wenige, aber einfach und intuitiv nutzbare Annotationsmöglichkeiten (nur Annotation mit Freihandlinien und ggfs. zusätzlicher Telepointer) sinnvoll. Wichtig sind gute Thumbnail-Übersichten und Blättermöglichekeiten, die nicht durch Whiteboard–Tools, sondern vermutlich besser durch einen getrennt bedienbaren „Weiter/Zurück"-Schalter realisiert werden, der ständig in einer Hand verbleibt.

Bei der Vorführung und Diskussion des MPEG-1-Filmes (s.o.) gegen Ende des Vortrags wurde allerdings deutlich, daß hier eine andere Annotationsmöglichkeit sinnvoll einsetzbar gewesen wäre: Zur Erläuterung hätte der Dozent gern auf einer dem laufenden Film überlagerten, transparenten Ebene Annotationen und Markierungen (mit Freihandlinien) vorgenommen. Ferner würde man es sich wünschen, den im Film gezeigten (virtuellen) Rundgang durch die Villa Urbana gleichzeitig auf einer im AOFwb aufgeblätterten Grundrisskarte markieren und verfolgen zu können. Alles das sollte natürlich synchron mit dem Audiostrom (verbale Erläuterungen des Archäologen) erfolgen.

Es sollten also neben den bei Telepräsentationen üblichen Datenströmen (Audio, Video, Whiteboardaktionen) noch der MPEG-Strom und ein Strom von Annotationen dieses Stroms aufgezeichnet werden. Ferner sollte im aufgezeichneten Dokument eine synchrone Echtzeit–Wiedergabe möglich sein und das alles noch möglichst unabhängig von der verwendeten Rechnerplattform und der aktuellen Auslastung des Rechners.

4.3 Bedeutung des Videobildes des Dozenten

Bei der (Tele–)Präsentation von Informatikinhalten kommt es vor allem darauf an, die Whiteboardaktionen und den Audiostrom des Dozenten zu synchronisieren. Das Videobild ("Talking Head") des Dozenten spielt - das zeigen zahlreiche Versuche - für das Verständnis kaum eine Rolle, wenn der Dozent seinen Vortrag direkt am Rechner mit Bildschirm, Tastatur und Maus hält. Denn dann sind Bewegungsfreiheit und Gestik des Dozenten ohnehin stark eingeschränkt. Die Situation ändert sich bereits, wenn der Dozent das SmartBoard benutzt. Ob in diesem Fall der mit dem Whiteboard– und Audiostrom synchronisierte Videostrom einen wesentlichen Beitrag zu Verständnis von Informatikinhalten liefert, ist bisher allerdings noch ungeklärt. Im Falle des Archäologen kann das Video des Vortragenden offensichtlich durch seine statt der Whiteboard-Werkzeuge zur Erläuterung von aufgeblätterten Bildern intuitiv benutzte Gestik sehr wichtig sein. Sie sollte daher bei der Übertragung an entfernte Standorte sehr gut sichtbar sein und auch in der Aufzeichnung, also beim Replay, zur Verfügung stehen. Dazu reicht ein H.261-Videostrom (vic [5]) bei geringer Framerate nicht aus. Die Übertragung des Videobildes des Sprechers sollte wenigstens PAL–Qualität haben. Wir haben das experimentell bei diesem Vortrag dadurch überprüft, daß eine PAL–Übertragung des Videostromes zusätzlich zur Übertragung des Whiteboard-Stromes in einem dem Vortragsraum benachbarten Hörsaal in Freiburg realisiert wurde. Das Ergebnis ist signifikant besser als eine Übertragung ausschließlich mit Hilfe der MBone–Tools. Zu beachten ist, daß dabei die getrennten Projektionen des Whiteboards und des Dozentenbildes beide in derselben Anordnung erscheinen wie im Vortragssaal, d.h. wenn der Dozent in der Regel rechts neben dem SmartBoard steht und nach links auf Bilder zeigt, sollte die Videoausgabe auch

Abbildung 3: SmartBoard: Ein interaktiver Großbildschirm mit Rückprojektion, der die Integration des Videobilds in Vorlesungsaufzeichnungen notwendig macht.

rechts von der Whiteboard-Ausgabe an den Empfängerstationen angeordnet werden.

Man sieht sofort, daß hieraus zahlreiche Anforderungen an die Übertragungsqualität und die Synchronisierung verschiedener Datenströme resultieren, die jedenfalls für ein skalierbares Multicast–Szenario im Internet gegenwärtig nicht realisierbar sind. Wir verweisen in diesem Zusammenhang auch auf die an der Universität Erlangen–Nürnberg gemachten Erfahrungen [4].

4.4 Replay

Auch für die Aufzeichnung und das Replay einer solchen (Tele-)Präsentation durch ein Note-Taking System, wie das AOF-System, ergeben sich zusätzliche Anforderungen: Nicht nur der Videostrom des Dozenten muss als ein weiterer *Slave Stream* neben dem Whiteboard-Strom und dem Applikationsstrom mit dem Audiostrom als *Master* (vgl. [8]) synchronisiert werden. Auch annotierte Video-Clips (hier: ein MPEG-1-Film) sollten bei der Wiedergabe der Aufzeichnung synchron dargestellt werden.

Bisher liegen keine Erfahrunge darüber vor, welches das sinnvollste Datenformat für derartige Videos (Bildgröße, Framerate, MPEG-1, MPEG-2, usw.) ist, welche Datenvolumina geliefert werden und wie man sichern kann, daß die für das Replay wesentliche Eigenschaften des direkten Zugriffs und der schnellen Navigation (Random Visible Scrolling) im durch Note-Taking erzeugten Dokument nicht verlorengehen.

Wir erläutern im nächsten Abschnitt ein Verfahren für die synchrone Wiedergabe von Video in Vorlesungsaufzeichnungen unter Beibehaltung des beliebigen Zugriffs. Wir beschreiben, wie die Implementierung des Verfahrens in ein Applikationsmodell eingebettet werden kann, das bereits routinemässig zur Wiedergabe von AOF-Dokumenten, also Vorlesungsaufzeichnungen nach dem AOF-Prinzip, verwendet wird.

5 Videosynchronisierung unter beliebigem Zugriff

Die Integration von Video in Vorlesungsaufzeichnungen, sei es das Videobild des Vortragenden oder einzeln eingestreute Video-Clips wie etwa der vorher beschriebene MPEG-Film über die Villa Urbana, machen aufgrund der anfallenden Datenmengen eine Kompression des Videostroms unerläßlich. So erzeugt beispielsweise ein unkomprimiertes Video in einer Auflösung von 352x288 (*CIF/SIF*), einer Farbtiefe von 24 bit und 25 fps eine Datenrate von ca. 60,8 Mbit/s. Selbst ein MPEG-1-Strom mit einer Datenrate zwischen 400 kbit/s und 1,2 Mbit/s erhöht die übliche Datenrate für AOF-Dokumente ohne Video (ca. 1 MByte/min) auf das bis zu Zehnfache. Im folgenden soll nun am Beispiel MPEG-1 erläutert werden, wie man den natürlichen Zwiespalt zwischen beliebigen Zugriff auf einen Videostrom und dessen Delta-Komprimierung oder Inter-Frame-Kodierung auflöst.

5.1 Beliebiger Zugriff

Um die einzelnen bei der Aufzeichnung einer Vorlesung entstehenden Medienströme (z.B. Audio, Video, Whiteboard) in Echtzeit und synchron zueinander wiedergeben zu können, müssen entweder alle Ströme in einem *Multiplexing*-Schritt in einem gemeinsamen Strom zusammengeführt werden (z.B. *MPEG-1 System*) oder aber jeder einzelne Strom während der Aufzeichnung mit zeitrelevanter Information (Zeitstempeln) versehen werden. Bei kontinuierlichen Datenströmen, wie unkomprimiertes Audio und Video, sind diese Informationen implizit durch Parameter wie Abtastrate und Größe der Abtastwerte (Frames/Samples) gegeben. Andere Datenströme wie etwa der Whiteboard-Strom müssen explizit mit solchen Zeitstempeln versehen werden [3]. Ist die Granularität dieser Zeitstempel fein genug, dann sind die grundsätzlichen Voraussetzungen für eine synchrone Wiedergabe der aufgezeichneten Datenströme gegeben.

Abgesehen von der einfachen Wiedergabe (Start, Stop, ggf. sequentielles Vorspulen) einer aufgezeichneten Vorlesung sind aber komfortablere Zugriffs- und Navigationsmöglichkeiten für einen weiten Einsatzbereich solcher Aufzeichnungen unerlässlich. Der *beliebige Zugriff* auf eine zusammengehörige Menge von Medienströmen - also die Verfügbarkeit der zugehörigen Daten aus allen Strömen für einen gegebenen beliebigen Zeitpunkt, sowie deren Darstellung auf dem Bildschirm bzw. hörbare Ausgabe über Lautsprecher innerhalb einer sehr kurzen medienabhängen Zeitspanne (vgl. [8] für eine genaue Definition) - ermöglicht viele dieser Mechanismen.

Random Visible Scrolling zum Beispiel ist ein komfortabler Navigationsmechanismus für AOF-Dokumente, der auf beliebigem Zugriff basiert. Der Benutzer hat hierbei die Möglichkeit, sich in jeder Geschwindigkeit vorwärts und rückwärts durch ein solches Dokument zu bewegen während gleichzeitig alle visuellen Daten angezeigt werden. Auf diese Weise kann man sehr schnell einen Überblick über den präsentierten Inhalt bekommen oder gesuchte Stellen leichter lokalisieren. Es erinnert in gewisser Weise an das schnelle Durchblättern eines Buches. Aber auch der direkte Zugriff auf signifikante Stellen der Aufzeichnung wie Folienwechsel, Startpunkte von Video-Clips oder der Beginn eines neuen Abschnitts sind für das Studium einer aufgezeichneten Vorlesung durchaus akzeptanzfördernd und für die Verbindung dieser Dokumente mit anderem relevantem Unterrichtsmaterial in hypermedialen Lehrumgebungen sogar zwingend.

Strom-inhärente Kriterien, wie die durchschnittliche Datenrate (z.B. 1,2 Mbit/s bei MPEG) oder die Länge der Historie (z.B. Distanz zwischen zwei aufeinander folgenden I-Frames) können den beliebigen Zugriff selbst beim einfachen Abspielen notwendig machen, wie im folgenden Abschnitt beschrieben wird. Hierbei tritt der klassische Zwiespalt zwischen delta-komprimierten Datenströmen und dem beliebigen Zugriff zu Tage. Auf den ersten Blick erfordert dieser direkte Zugriff auf die Daten in einem Medienstrom, daß für jeden vorgebenen Zeitpunkt die vollständige Information, um zum Beispiel den zugehörigen Videoframe darstellen zu können, an einer eindeutig und schnell zu bestimmenden Position im Strom verfügbar ist. Um einen schnellen Zugriff zu gewährleisten sollte die Darstellung der Daten an dieser Position dabei möglichst ohne Rückgriff auf andere vorhergehende oder nachfolgende Daten erfolgen. Das führt zwangsläufig zu einer Redundanz, da Informationen, die über mehrere Abtastpunkte gleich bleiben jeweils in allen zugehörigen Werten vollständig abgespeichert werden müssen. Dies gilt insbesondere für Video, da hier die Abstände zwischen zwei Werten weniger als 1/25 Sekunde betragen. Man stelle sich als Extrembeispiel ein unkomprimiertes Video vor, bei dem sich über mehrere Sekunden einfach nichts verändert. Jeder Frame entspricht dann dem vorhergehenden. Der Sinn einer Delta-Komprimierung liegt nun gerade darin, diese Redundanzen zu beseitigen. Diese Problematik tritt auch bei nicht kontinuierlichen Datenströmen wie zum Beispiel dem Whiteboard-Strom (vgl. [8]) auf. Im folgenden Abschnitt soll nun gezeigt werden, wie man die Trägheit der menschlichen Wahrnehmung ausnutzen kann, um diesen Zwiespalt bei MPEG zu umgehen.

5.2 Synchronisierung von MPEG

Eine Inter-Frame-Kodierung eines Videostroms wie bei MPEG hat zwangsläufig eine variable Bitrate zur Folge. Die Position zu einem gegebenen Zeitpunkt im Strom läßt sich also nicht einfach aus der Framerate und der Größe der Frames berechnen. Ein naive Lösung besteht nun darin, in einem Postprocessing-Schritt nach der Aufzeichnung des MPEG-Videos einen Index zu den Frame-Positionen zu erstellen, indem man den Strom von vorne bis hinten durchläuft und die Byte-Positionen in einer separaten Datei abspeichert. Indexwerte mit einer Größe von 4 Byte können MPEG-Videos bis zu einer Länge von ca. 4,2 GByte adressieren und sind damit für die meisten Anwendungszwecke vollkommen ausreichend. Das Datenvolumen wird dann durch die Indexierung etwa bei 25 fps um vernachlässigbare 200 bit/s erhöht. Um bei jedem beliebigen Zugriff auf den MPEG-Strom einen weiteren Dateizugriff zu vermeiden, kann man bei diesen geringen Datenmengen den Index vor Beginn der Wiedergabe in ein Array einlesen und bekommt so verzögerungsfrei die zu einem Frame gehörende Byte-Position.

Bei der Implementierung unseres MPEG-Players verwenden wir jedoch ein anderes Verfahrens. Die zugrundeliegende Idee ist die Ausnutzung der durchschnittlichen Bitrate auf dem *Sequence Layer*, um die Position b_0 des gesuchten Frames auf dem *GOP Layer* zu schätzen. Wenn die Schätzung falsch ist, dann kann die verwendete nicht die bis zu b_0 passende Bitrate gewesen sein. Die vor b_0 liegenden Frames waren also im Schnitt größer oder eben kleiner, also es die ursprüngliche Bitrate prognostiziert hat. Im nächsten Schritt wird nun die zu b_0 passende Bitrate für eine weitere Schätzung verwendet und so weiter. Genauer: Anhand einer *GOP* (*Group of pictures*) kann erkannt werden, ob der gesuchte Frame enthalten ist. Da stets wahlfreier Zugriff auf die in einer GOP enthaltenen Frames möglich ist, besteht die Suche nach einem Frame zu einem gegebenen Zeitpunkt in der Suche nach der zugehörigen GOP. Die durchschnittliche Bitrate br_0 aus dem Sequenz-Header liefert nun die vermutliche Byte-Position b_0 des gesuchten Frames zu einem vorgegebenen Zeitpunkt. Wenn die zu b_0 gehörende nicht die gesuchte GOP ist, dann wird dasselbe Verfahren mit der neuen durchschnittlichen Bitrate br_1 rekursiv fortgesetzt. Diese berechnet sich aus dem Zeitstempel (*time code*) der GOP, sowie seiner Byte-Position und muß von br_0 verschieden sein.

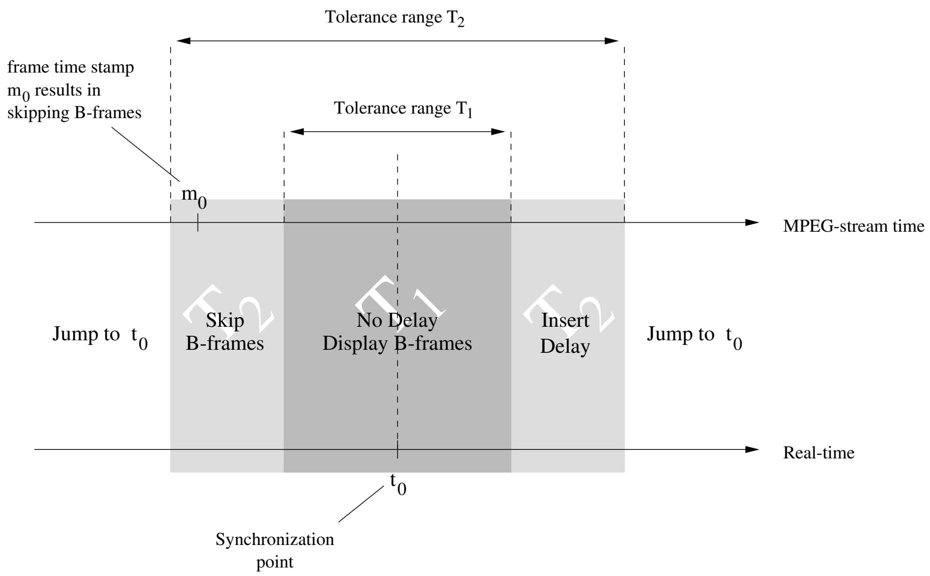
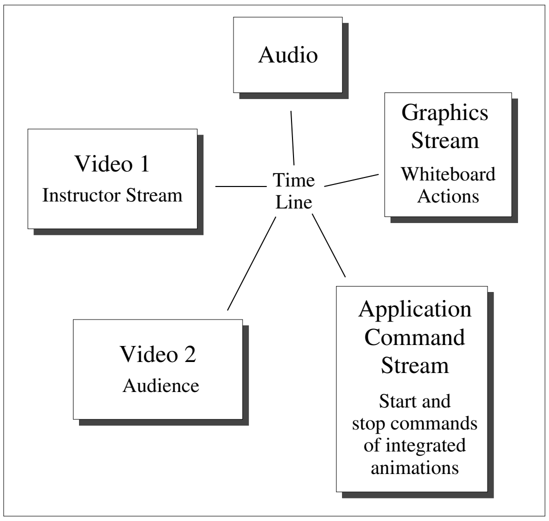


Abbildung 4: Wiedergabe-Modi in Abhängigkeit von der Echtzeit-Abweichung

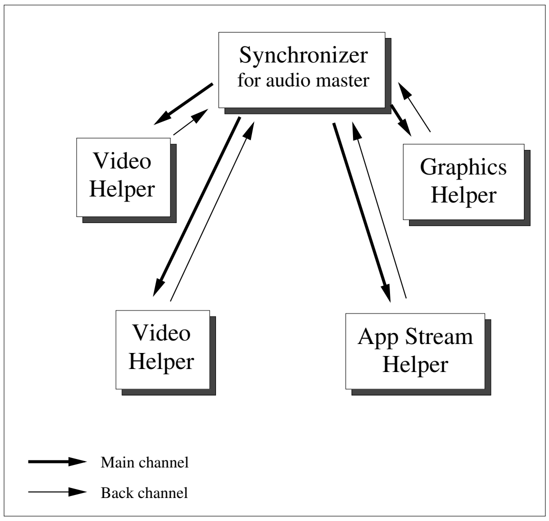
Offensichtlich kann die Bitrate nicht endlos steigen oder fallen, da dies durch die maximale Bitrate bei MPEG beschränkt ist und die Differenz zwischen zwei aufeinanderfolgenden Bitraten nicht beliebig klein werden kann. Dennoch ist diese Vorgehensweise in dem Moment kein gangbarer Weg, in dem nach dem Steigen ein Fallen der Bitrate folgt. Man sieht schnell an einfachen Beispielen, daß dann das Verfahren nicht zwingend terminiert. In dem Fall, daß dem Steigen ein Fallen der Bitrate folgt (oder umgekehrt) wird für die neue Bitrate schlicht das arithmetische Mittel der letzten beiden Bitraten verwendet. Das Verfahren ähnelt dann einer Mischung aus binärer Suche und Interpolationssuche [15].

Die Komplexität dieses Algorithmus, d.h die maximale Rekursionstiefe, ist bis jetzt nicht formal berechnet worden. Alle bisher durchgeführten Experimente mit unserem existierenden Prototyp haben jedoch gezeigt, daß die gesuchte Byte-Position im Stream in einer Zeit gefunden wird, die deutlich innerhalb des gesetzten Toleranzbereichs liegt.

Für eine sofortige Darstellung auf dem Bildschirm reicht der direkte Zugriff auf einen Frame natürlich nicht aus. Wenn es sich nicht zufällig um eine I-Frame (*Intra-coded frame*) handelt, muß für die Dekodierung des Frames auf davor und dahinter liegende Frames zurückgegriffen werden. Unsere Experimente haben jedoch gezeigt, daß der gesuchte Frame auf heute gängigen PCs bei der üblichen Frame-Reihenfolge *IBBPBBPBBI...*, bei den gängigen Auflösungen und Bitraten stets innerhalb der Toleranz decodiert zur Verfügung steht. Da dies aber nicht auf allen Rechner immer gewährleistet werden kann und die Zugriffzeiten auch durch strom-unabhängige, nicht verhersehbare Faktoren, wie beispielweise LAN-Last, *Process Scheduling* und Auslastung der Maschine beeinflußt werden kann, muß auch die einfache Wiedergabe des MPEG-Stroms für eine gesicherte Synchronität angepasst werden. In unserem Verfahren wird vor der Darstellung jedes Frames dessen implizit gegebener Zeitstempel mit der Echtzeit vergleichen. Solange die Abweichung innerhalb eines ersten vernachlässigbaren Toleranzbereichs T_1 liegt (s. Abb. 4), wird dies ignoriert. Für den Fall, daß die Abweichung aus T_1 herausfällt und die Filmzeit der Echtzeit hinterher hinkt, wird die Darstellung der *B-Frames* (*Bi-directionally predictive-coded frames*) abgeschaltet. Es werden also keine B-Frames mehr dargestellt in der Hoffnung, daß die Filmzeit gegenüber der Echtzeit aufholt. Entsprechend wird die Darstellung der B-Frames wieder eingeschaltet, wenn die Abweichung wieder innerhalb T_1 liegt. Für den Fall, daß die Filmzeit der Echtzeit vorauseilt - der Film wird also vermutlich zu schnell abgespielt - wird eine geeignete Verzögerung von einigen Millisekunden zwischen den Frames eingefügt, solange bis die Abweichung wieder in T_1 hereinfällt. Wenn die Abweichung auch aus einem zweiten T_1 umfassenden Toleranzbereich T_2 herausfällt, wird der beliebige Zugriff ausgenutzt und an die gesuchte Stelle t_0 (Echtzeit) gesprungen. Geeignete Größen für T_1 und T_2 haben sich bei uns im Laufe einiger Experimente herauskristallisiert.



Example document



Dynamically generated application structure

Abbildung 5: Ein Beispiel für ein AOF-Dokument und der zugehörige Aufbau der wiedergebenden Applikation [8].

Da bei dieser Vorgehensweise im Extremfall (z.B. bei starker Systemauslastung oder schwacher Rechnerleistung) nur von Frame zu Frame gesprungen wird, werden also nur soviele Frames angezeigt, wie es die beschriebenen Faktoren eben zulassen. D.h. die Framerate passt sich automatisch diesen Faktoren an und führt damit zu einem dynamischen Ausgleich zwischen der Anzahl dargestellter Frames und der

Systemauslastung. Wir bezeichnen dies mit *Dynamischer Frameraten-Anpassung* (*Dynamic Frame Rate Adaption*). Zugunsten möglichst hoher Synchronität werden hier also einzelne Informationen ausgelassen und nicht angezeigt. Dies Prinzip entspricht der Zielrichtung einer unsicheren Multicast-Übertragung (z.B. MBone), bei der der Verlust von Daten zugunsten einer möglichst verzögerungsfreien Übertragung akzeptiert wird.

Das zuvor beschriebene Verfahren der MPEG-Wiedergabe unter beliebigem Zugriff haben wir, wie oben bereits erwähnt, in einem Prototyp eines MPEG-Players implementiert, der bereits für verschiedene Plattformen existiert (Windows95/98, Unix-Derivate). Der Prototyp fungiert als *Helper*-Applikation für das in [8] genauer beschriebene Applikationsmodell, daß ebenfalls bereits implementiert ist und routinemässig für die Wiedergabe von AOF-Dokumenten genutzt wird (vgl. [1],[2]). Dem Applikationsmodell liegt eine Aufsplittung der Wiedergabe auf mehrere nahezu unabhängige Präsentationsmodule (ein *Synchronizer* und verschiedene *Helper*) gemäß der Zusammensetzung der einzelnen Medienströme im AOF-Dokument zugrunde (siehe Abb. 5). Der Synchronizer gibt dabei den Audio-Strom wieder und sorgt für die Synchronisierung der einzelnen Ströme. Dazu werden zwischen dem Synchronizer und den einzelnen Helper-Applikationen Kommunikationskanäle geschaltet, über die insbesondere die Zeitstempel für die Synchronisierung übertragen werden (siehe Abb. 6). Da die Helper-Applikation die ihnen zugeordenten Medienströme selbständig wiedergeben und die Wiedergabe anhand der empfangenen Zeitstempel synchronisieren können, ermöglicht das Applikationsmodell damit die oben genannte dynamische Frameraten-Anpassung.

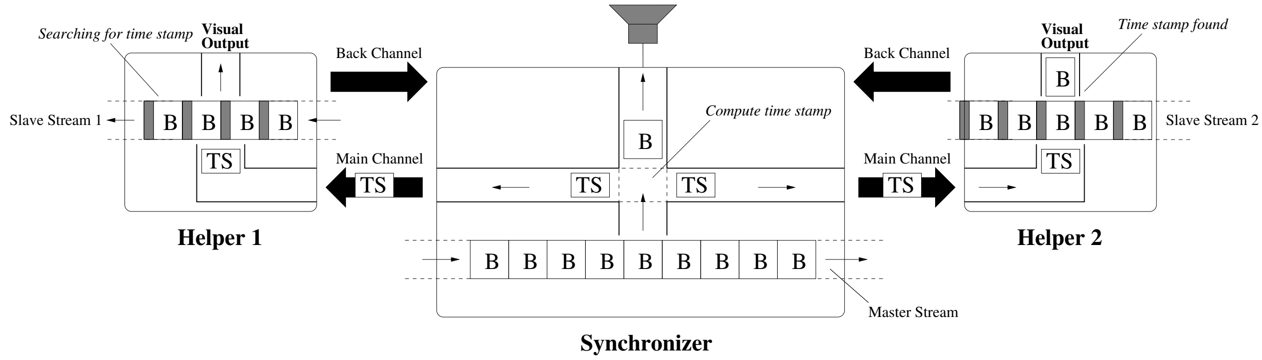


Abbildung 6: Konzeptuelle Darstellung der Funktionsweise des Applikationsmodell für die Wiedergabe von AOF-Dokumenten [8].

Mit Hilfe der Helper-Applikation für MPEG-1 ist also die Integration von Video-Clips, wie auch vom Videobild des Vortragenden in AOF-Dokumente unter Beibehaltung des beliebigen Zugriff und der Synchronität zu den verbleibenden Medienströmen der aufgezeichneten Vorlesung realisierbar.

5.3 Video-Annotation

Bei der allgemeinen Fragestellung der grafischen Annotation von in die Vorlesung integrierten Fremdapplikationen (z.B. Animationen, Simulationen, Filme) soll der Vortragende die Möglichkeit bekommen auf der grafischen Ausgabe der Anwendung (d.h. den Fenstern) zu zeichnen und dort dargestellte Objekte zu zeigen. Ähnlich den grafischen Fähigkeiten eines Whiteboards sollen hier also Freihand-Linien, einfache grafische Objekte wie Linien, Rechtecke, Ellipsen und eben auch ein Tele/Online-Pointer bereitgestellt werden. Die Annotationen, die der Vortragende während der Aufzeichnung auf den Fenstern der Applikation ausgeführt hat, sollen dann bei der Wiedergabe synchron zum erneuten Ausführen der Applikationen dargestellt werden.

Eine allgemeine Lösung dieser Aufgabe ist schwer zu realisieren, insbesondere wenn der Sourcecode der Applikation nicht verfügbar ist. Die Realisierung hängt stark vom unterliegenden Fenstersystem (z.B. Windows, X-Windows) ab und ist damit auf jeden Fall plattformabhängig. Eine plattformunabhängige Variante über Java würde das Problem nur für Java-Applikationen lösen und wäre für Anwendungen mit hohen Frameraten wie Animationen und Filme sehr wahrscheinlich viel zu langsam.

Prinzipiell erfordert das Problem die Realisierung eines transparenten Fensters, welches über einen Event/Request-Mechanismus über Veränderungen des dahinter liegenden Bildschirminhalts informiert wird, um die eigenen Annotation entsprechend aktualisieren zu können. Systemerweiterungen wie beispielsweise die *Shape Extension* unter X-Windows erlauben das Setzen von beliebigen, nicht rechteckigen Clipping-Bereichen und damit die automatische Restriktion des Zeichenbereichs eines Fensters auf etwa eine Freihand-Linie. Alles außerhalb dieser durch Bit-Masken festgelegten Clipping-Bereiche ist nicht Bestandteil des Fensters und damit transparent. Allerdings bringt die Komplexität und wohl auch Zweckentfremdung dieses Mechanismus Geschwindigkeitsprobleme mit sich, insbesondere dann, wenn Videos oder Animationen mit hohen Frameraten annotiert werden sollen. Ähnliches gilt auch für den obengenannten Event-/Request-Mechanismus, da dieser bei hohen Frameraten nicht schnell genug und zudem asynchron ist.

In dem hier betrachteten Spezialfall, bei dem es sich um einen eigens implementierten MPEG-Player handelt - der Sourcecode also zur Verfügung steht - ist die Annotation eines Videos recht leicht realisierbar. Ein besonderes Augenmerk der Diskussion im letzten Abschnitt war der Zeitpunkt, zu dem ein Frame dargestellt wird. Nachdem also der aktuelle decodierte Frame zur Darstellung in einen Puffer geschrieben worden ist, werden die Annotationen des Vortragenden hinzugefügt und erscheinen damit über dem Frame des Videos. Danach wird der gesamte Puffer gemäß dem *Double-Buffering* o.ä. für den Benutzer sichtbar in ein Fenster kopiert. Der Aufwand für die zusätzlich hinzugefügten Annotationen ist dann im Vergleich zum Decodieren und Zeichnen des Frames selbst minimal.

Eine interessante Fragestellung in Bezug auf Annotation bei frame-basierter grafischer Ausgabe ist derer Eignung. Insbesondere in Animationen und Filmen, bei denen die Sichtbarkeitsverhältnisse von Objekten sehr schnell wechseln - Objekte also oft nur sehr kurz sichtbar sind oder sich sehr schnell bewegen - wird Annotation im laufenden Zustand sinnlos. Hier wäre es sicher besser, den Videofilm kurzzeitig für die Annotation abzustoppen und nach Beendigung mit der Wiedergabe fortzufahren. Man kann sich jedoch auch Beispiele vorstellen, in denen grafisches Annotieren im laufenden Zustand hilfreich ist. Ein Beispiel ist der Film über die Villa Urbana, wenn etwa der zukünftige Pfad der Kamera durch die Szene beschrieben oder allgemein die anschließend folgende Bewegung von Objekten vorgezeichnet werden soll. Es ist also sicherlich sinnvoll beide Modi - Annotation im laufenden und im gestoppten Zustand - zur Verfügung zu stellen.

6 Zusammenarbeit mit der Archäologie

Charakteristisch für dieses Projekt war die intensive Zusammenarbeit zwischen Systemfachleuten aus der Informatik und den das Ausgrabungsprojekt betreuenden Wissenschaftlern der Archäologie über einen langen Zeitraum. Die Archäologen gaben den Modellierern die Vorstellung, wie das Gebäude wohl einmal ausgesehen haben könnte. Weil nur die Grundmauern durch die Grabung verläßlich rekonstruierbar waren, waren für alle weiteren Teile des Gebäudes plausible Vorgaben der Archäologie nötig. Daß die Umsetzung in das Modell wiederum Rückwirkungen auf die archäologischen Vorstellungen hatte, mag folgendes Beispiel belegen. Aus der Freilegung eines Kellerraumes konnte geschlossen werden, daß dieser Raum an den Seiten jeweils zwei ebenerdige Fenster hatte. Nun stelle man sich vor, wie eine Person aufrecht im Keller steht und aus einem Fenster hinaus in den Garten schaut. Da die Fenster ebenerdig sind, kann der Raum darüber nicht ebenfalls ebenerdig auf gleicher Höhe mit allen anderen Räumen gewesen sein. Die Konsistenz des Modells erzwang eine besondere Erklärung. Archäologische Überlegungen ergaben, daß es sich bei diesem über dem Keller liegenden Raum wohl um eine Art Festsaal gehandelt haben muß, also ein Ort für Vergnügungen und Feierlichkeiten des Hausherrn. Dieser Raum enthielt vermutlich ein sogenanntes Triclinium, auf dem Gastgeber und Geladene zu Tische lagen. Dieser Raum war daher gegenüber den anderen um einige Eintrittsstufen höher gelegt.

Auch bei der Modellierung der Umfassungsmauer um den Siedlungskomplex stellte sich heraus, daß die ursprüngliche Vermutung über ihre Höhe revidiert werden mußte: Die Modellierung ergab, daß nur eine die Geländestruktur berücksichtigende und nicht zu hohe Mauer mit der Vorstellung eines Villenkomplexes verträglich ist.

Insgesamt zeigte sich an vielen Details, daß auch die Archäologen ihre durch die Ausgrabung begründeten Vorstellungen durch die Computer-Modellierung häufig wesentlich präzisieren und insgesamt plausibler zusammenfügen konnten. Und wie man anhand des vorgestellten Filmes sieht, war die Zusammenarbeit zwischen Informatik und Archäologie recht fruchtbar.

7 Beteiligte und Danksagung

Im Verlauf der Jahre haben an verschiedenen Stadien der Modellierung und Filmberechnung zahlreiche Personen mitgewirkt, deren Vorarbeiten in das nun vorliegende Ergebnis eingeflossen sind. Besonderer Dank geht an Steffen Bunzel, der bei allen Fragen, die das Programm betrafen, stets geholfen hat. Beteiligt waren ferner Jörg Finger, Christian Wetzel, Ronny Fehling, und Thomas Loop. Technische Hilfe leistete auch Frank Dal-Ri. Die Konvertierung des Filmes auf Videoband wurde mit Mitarbeitern am Lehrstuhl für Angewandte Mathematik durchgeführt. Großer Dank gilt auch Oliver Klerx für die Hilfe bei der Implementierung des MPEG-Players.

Literatur

[1] Authoring on the Fly - Document Download Area: http://ad. informatik.uni-freiburg.de/mmgroup.-aof.docdownload

[2] A Tool for the Synchronous Replay of AOF Documents: Institute for Computer Science, University of Freiburg, http://ad.informatik.uni-freiburg.de/mmgroup.tools.aofSync

[3] Bacher C, Müller R (1998) Generalized Replay of Multi-Streamed Authored Documents. Proceedings of ED-MEDIA ' 98 Conference of Educational Multimedia and Hypermedia, Freiburg, Germany, June 1998

[4] Bodendorf F, Grebner R, Langenbach C (1996) The Virtual Lecture Theatre - Practice and Experience. Proceedings of 2. Russian-German Symposium, New Media for Education and Training in Computer Science, Swiridow AP, Widmayer P, Oberhoff WD, Unger H (Eds.), Moscow, Nov. 1996.

[5] Floyd S, Jacobson V, Liu C, McCanne S, Zhang L (1997) A Reliable Multicast Framework for Light-weight Sessions and Application Level Framing. IEEE/ACM Transactions on Networking, December 1997, Vol. 5, No. 6

[6] Jährliche Grabungsberichte in: Archäologische Ausgrabungen in Baden Württemberg 1991 ff.

[7] Mental Ray: Mental images Gesellschaft für Computerfilm und Maschinenintelligenz mbH Co. KG, Berlin

[8] Müller R, Ottmann Th (1999) The “Authoring on the Fly” System for Automated Recording and Replay of (Tele)presentations, to appear in: Special Issue on “Multimedia Authoring and Presentation Techniques” of ACM/Springer Multimedia Systems Journal

[9] Macedonia MR, Brutzman DP (1994) MBone Provides Audio and Video Across the Internet. IEEE Computer, Vol. 27, No. 4, April 1994

[10] Nuber HU (1994) Villenforschung an der Universität Freiburg im Breisgau. In: S. Palágyi (Hrsg.), Forschungen und Ergebnisse. Internationale Tagung über römische Villen. Vesyprém 16.–20. Mai 1994. Balácai Közlemények III, 1994 (Veszprém 1995) 159. Die villa urbana von Heitersheim (D). - Ebd.

[11] Nuber HU (1997) Römische Antike am Oberrhein: Die villa urbana in Heitersheim. Archäologische Nachrichten aus Baden 57, 1997.

[12] Seitz G (1998) Archäologieprojekt Villa Urbana Heitersheim. In: Expressum. Informationen aus dem Freiburger Bibliothekssystem. Elektronische Dienste für Studium und Wissenschaft (Sonderausgabe 1998).

[13] Softimage Inc., a wholly owned subsidiary of Microsoft corporation. http://www.softimage.com

[14] VIROR - The Upper Rhine Virtual University (orig. Virtuelle Universität Oberrhein). http://www.-viror.de

[15] Yao AC, Yao FF (1976) The Complexity of Searching an Ordered Random Table. Proceedings of 17th Annual Symposium on Foundations of Computer Science, 1976