Algorithmus-Animation in Lehr- und Lernsystemen der Computergraphik

Wolfram Luther

Fachgebiet Informatik, Fachbereich 11
Gerhard-Mercator-Universität Duisburg
Lotharstraße 65
D-47057 Duisburg

1 Einleitung

In fast allen Schulen und Hochschulen, aber auch im häuslichen Umfeld stehen heutzutage vernetzte Computer zur Verfügung, die dazu benutzt werden können, Lernenden den Umgang mit computer-basierten Medien zu vermitteln. Wir widmen uns einem speziellen Aspekt des Computereinsatzes im Unterricht, der Visualisierung von Datenstrukturen und Erklärung von Algorithmen in animierter Form. Gerade Algorithmen eignen sich durch ihren zumeist dynamischen Charakter sehr gut für die animierte Darstellung am Bildschirm, einer elektronischen Tafel oder Projektion und sind in einer klassischen Lehrveranstaltung an der Kreidetafel oder mit Hilfe von Overhead-Folien meist nur unzureichend zu vermitteln. Wir gehen dabei vor allem auch auf verschiedene Typen von Lehr- und Lernsystemen ein, definieren Anforderungen an die Animation von Algorithmen und zeigen Methoden auf, wie diese umgesetzt werden können. Gleichzeitig geben wir einen Überblick über den Stand unserer Projekte zur Erstellung von Lehr- und Lernumgebungen für Algorithmen aus dem Bereich der Computergraphik und Bildverarbeitung und ordnen sie in den Rahmen ähnlicher Aktivitäten in Deutschland und der Welt ein.

2 Eine Lernumgebung für Algorithmen der Computergraphik

Das Lehr- und Lernsystem ViACoBi (Visualisierung von Algorithmen der Computergraphik und Bildverarbeitung) [Janser 1997a, 1997b, 1998] widmet sich einem speziellen Aspekt des Computereinsatzes in der Lehre und ergänzt unmittelbar den klassischen Vorlesungsstoff zu 2D-Graphik und Grundlagen der Bildverarbeitung. Es wurde mit dem Multimedia-Autorensystem Authorware Professional für APPLE Macintosh entwickelt, auf Windows-Rechner portiert und baut auf einer Monographie zum selben Thema auf [Janser et al. 1996]. Weiterhin ist es durch eigenständige kleinere Systeme wie Grafik-Demo, HADES und MORPH, die in mehreren Seminar- und Abschlussarbeiten zu den Themen Low Level Routinen der Bildverarbeitung, fraktale Strukturen, Bintrees, Quadtrees und Octrees, Morphing sowie 3D-Rekonstruktion erstellt wurden, flankiert.

Letztere erlauben eher das freie Experimentieren mit Algorithmen und Datenstrukturen und verfügen über keinerlei Lernermodellierung oder Steuerung des Ablaufs durch das System. ViACoBi behandelt in acht Grundlagenkapiteln eine Darstellung des Arbeitsplatzes, Rasteralgorithmen für Strecke, Kreis und Ellipse und die Bearbeitung und Kompression von Grauwertbildern. Neben einer Aufbereitung des Lernstoffes in Hypertextform, der auch von einem Sprecher präsentiert wird, liegt das Schwergewicht auf einer Animation und Visualisierung der Wirkungsweise der Basis-Algorithmen. Zusätzlich enthält das System ein umfangreiches Glossar und eine Vielzahl von zum Teil automatisch generierten Aufgaben zum interaktiven Lösen. Folgende Aufgabentypen werden in ViACoBi unterstützt:

- Multiple Choice-Aufgaben,
- Aufgaben zum Erfassen visueller Zusammenhänge,
- Konstruktionsaufgaben,
- Aufgaben mit abrufbarer Musterlösung,
- Experimentelle Aufgaben.

Im Rahmen der Arbeiten am System und seiner Evaluation haben wir die traditionelle Unterrichtsorganisation modifiziert. Während statische Übersichten und beweisintensive Teile des Stoffes in der Vorlesung verbleiben und in einer Übung in Gruppen vertieft werden, sind wiederholende Zusammenfassungen, die Arbeit mit Graphikalgorithmen sowie eine Vielzahl verschiedener Aufgabentypen in eine *Praktische Übung* mit dem System ViACoBi verlagert worden.

Unsere Lehr- und Lernsysteme wurden auf verschiedenen Messen präsentiert, in Fortbildungen eingesetzt und parallel zur Entwicklung oder direkt im Anschluss an die Fertigstellung von kleineren heterogenen Lerngruppen bezüglich der folgenden Punkte in schriftlichen und mündlichen Befragungen mehrfach bewertet:

- Auswahl, Aufbereitung, inhaltliche Darstellung und Verständlichkeit des Lernstoffes
- Übersichtlichkeit und Folgerichtigkeit der Kursstruktur,
- adäquater Einsatz der Medien, Design, Benutzerführung,
- Verhältnis informierender, aktivierender und prüfender Inhalte,
- Bedienungsfreundlichkeit und Zuverlässigkeit.

3 Klassifizierung von Lehr- und Lernsystemen

Multimediale Lehr- und Lernsysteme können in mehrdimensionale Räume kognitiver Medien eingeordnet werden. Wesentliche Parameter sind

- der Grad der Lerneraktivität,
- der Anteil des Systems an Kontrolle und Steuerung,
- der Anteil von Präsentation von Lernstoffen durch das System im Vergleich zu kreativen Möglichkeiten des Lerners.

Die dreidimensionale Darstellung kognitiver Medien geht in dieser Form auf Klimsa [Klimsa 1995] zurück. Es ist allerdings zu bemerken, dass diese drei Parameter nicht unabhängig voneinander sind. Zudem sind auch weitere Parameter denkbar, wie der

Anteil oder das Gewicht verschiedener computerbasierter Medien oder die Vielfalt unterstützter Aufgabentypen.

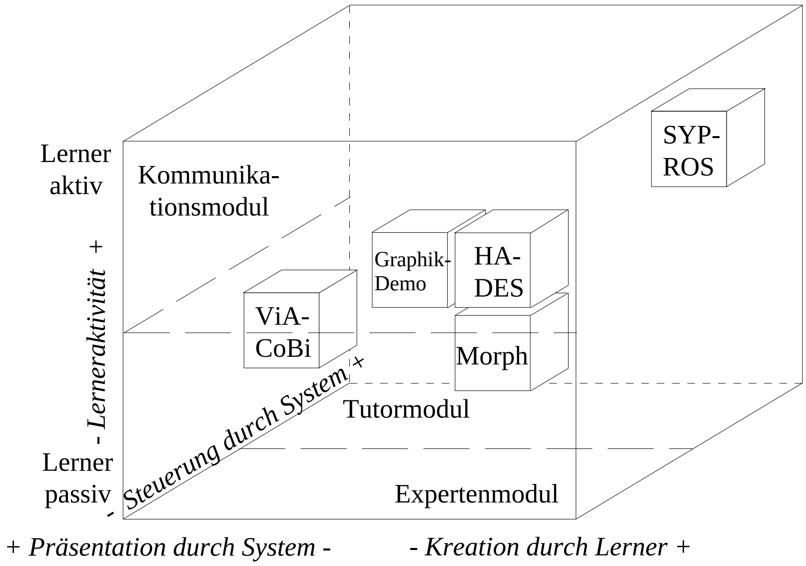

Abb. 1: Klassifizierung von Lehr- und Lernsystemen

Systeme zur Algorithmus-Animation wie ViACoBi sehen sich bezüglich der Lerneraktivität in einer mittleren Position. Innerhalb der Lektionen kann der Lerner frei navigieren und zu jeder Zeit Aufgaben lösen oder das Glossar konsultieren. Die Möglichkeiten der Interaktion werden im Rahmen eines *Kommunikationsmoduls* definiert. Es enthält Schnittstellen zur Steuerung des Lernverlaufs, zur Eingabe von Lösungen, Zugänge zu Hilfen. Glossaren und Ausgaben von Präsentationen und Animationen. Nutzer können im Rahmen von Aufgaben die Algorithmen in verschiedenen Parameterkonfigurationen austesten. Konstruktive Aufgaben sind didaktisch sinnvoll, setzen jedoch das Vorhandensein mächtiger Eingabeschnittstellen voraus, die mathematische Formeln parsen können. Auch ist dann die Anbindung von Computer-Algebrasystemen sinnvoll, die schrittweise Vorgehensweisen beim Lösen von Aufgaben unterstützen. Schließlich müssen 3D-Visualisierungswerkzeuge vorgesehen werden, da klassische Autorenprogramme im wesentlichen 2D-Graphik unterstützen.

Die Kreativität des Lerners ist somit recht eingeschränkt und kann nur in interessanten experimentellen Aufgaben mit einfachen Eingaben zur Geltung kommen. Hier entsteht jedoch ein neues Problem. Während bei Systemen, deren Lernstoffe aus großen oder verschiedenartigen Wissensdomänen stammen, Kontrolle und Steuerung eher beim Lerner verbleiben, müssen Systeme mit einer eigenen Kontrolle und Steuerung über ein *Expertenmodul* verfügen. In diesem ist das Wissen der vom System behandelten Domäne insoweit vollständig abgebildet, dass auf jede Eingabe des Lerners eine adäquate Reaktion erfolgt. Werden Aufgaben frei generiert, so muss das System jeden Schritt zu einer Lösung korrekt beurteilen und selbst eine korrekte

Lösung liefern. Daher haben wir in ViACoBi die Klasse aller korrekt abbildbaren Funktionen *KAF* der Menge der Farbbildmatrizen in sich oder auf einen *n*-stelligen Vektor von ganzen Maschinenzahlen definiert. Wir visualisieren und animieren die Klasse der korrekt abbildbaren Algorithmen *KAA*, die die Funktionen aus der Klasse *KAF* berechnen und dabei nur verifizierbare Operationen im Sinne des IEEE 754/854-Standards für die Gleitkommaoperationen ausführen.

Unter der semantischen Funktion *S*(*A*) eines Algorithmus *A* verstehen wir die Funktion *f*, die *A* auf seinen Eingabedaten berechnet. Dabei abstrahieren wir von der Programmsemantik eines Programms *P* und den beschriebenen Zustandstransformationen auf die Zustandsänderungen, die der Algorithmus auf den zugrunde liegenden Daten bewirkt. Wohl bemerkt machen wir keine Aussage zur Korrektheit der Algorithmen an sich und zum Problem, ob der Algorithmus die gewünschte Funktion berechnet, sondern nur zur Fragestellung, ob der Rechner auf einer realen Maschine das Ergebnis berechnet, das ein Rechner mit unendlicher Genauigkeit liefern würde. Solange sich die Rechnungen im Ganzzahlbereich bewegen und keine Bereichsüberschreitungen eintreten, können wir die Zugehörigkeit der Algorithmen zu unserer Klasse *KAA* garantieren. Im Bereich der Maschinenzahlen müssen wir kontrollierte Rundungen voraussetzen und Auslöschungen ausschließen. So sind alle in ViACoBi animierten Algorithmen aus *KAA*, und wir können selbst im Falle frei generierter Aufgaben zusichern, dass das System eine numerisch korrekte Lösung angibt.

Die Übernahme der Steuerung des Lernablaufs durch das System setzt das Vorhandensein eines *Tutormoduls* voraus, das grundlegende Strategien für das Vermitteln von Unterrichtsstoff nachbildet und in Abhängigkeit vom Kenntnisstand des Lerners das richtige Problem aussucht und adäquate Vorschläge zur Lösung macht. So zeichnet ein Tool zur Lernermodellierung Informationen darüber auf, in welchem Umfang der Lernende die Wissenselemente der Domäne bereits beherrscht, welche Arbeitsweisen er bevorzugt hat und welche Hilfestellungen er abruft.

Das intelligente Lehrsystem SYPROS, das derzeit unter der Leitung von Chr. Herzog [Rothenhöfer et al. 1993] an der TU-München entwickelt wird, ist für den Einsatz auf dem Lehrgebiet *Synchronisation paralleler Prozesse mit Semaphoren* konzipiert. Das für die Synchronisation grundlegende von Dijkstra eingeführte Konzept der Semaphore wird in SYPROS beispielhaft anhand von kooperierenden Prozessen eingeübt. Es ist ein herausragender Vertreter für ein intelligentes Lehr- und Lernsystem mit einer Lernermodellierung, einem Kommunikations-, Tutor- und Expertenmodul.

Zusammenfassend sind folgende Aspekte zu beachten:

- Interaktive multimediale Lehr- und Lernsysteme sind in der Entwicklung ausgesprochen zeitaufwändig (100 und mehr Arbeitsstunden sind für eine Kurseinheit von einer Stunde üblich).
- Systeme mit einer durchgehenden Lernermodellierung, Lernzielstrukturierung und intelligenten Lernerführung sind oft nur auf einen sehr engen Themenbereich begrenzt.
- Kurse zu umfangreichen Themenbereichen sind meist nur als freie Angebotssysteme mit eingeschränkter Lernermodellierung und Hilfestellung in Form von Anleitungen und Glossaren denkbar.

- Lehr- und Lernsoftware kann nur kostenrentabel erstellt werden, wenn sie in lernerintensiven Bereichen eventuell begleitend zu klassischer Lehre eingesetzt wird. Dabei muss der verantwortliche Dozent die Möglichkeit einer Aktualisierbarkeit und Erweiterbarkeit haben.
- Der erfolgreiche Einsatz multimedialer Lehr- und Lernsoftware ist auf eine moderne und leistungsfähige Infrastruktur an den Hochschulen angewiesen
- Die rechtlichen Probleme bei der Verwendung copyright-geschützter multimedialer Elemente müssen geklärt sein.

4 Animationsebenen

In [Janser, Luther 1994] haben wir wichtige Methoden und Anforderungen für die Algorithmus-Animation im Rahmen von Lehr- und Lernsystemen definiert, die in der Folgezeit in unseren Systemen umgesetzt wurden. Bei der visuellen Animation von Algorithmen unterscheiden wir zwei grundsätzliche Ebenen, die Animation von Daten und die Animation von Programmen.

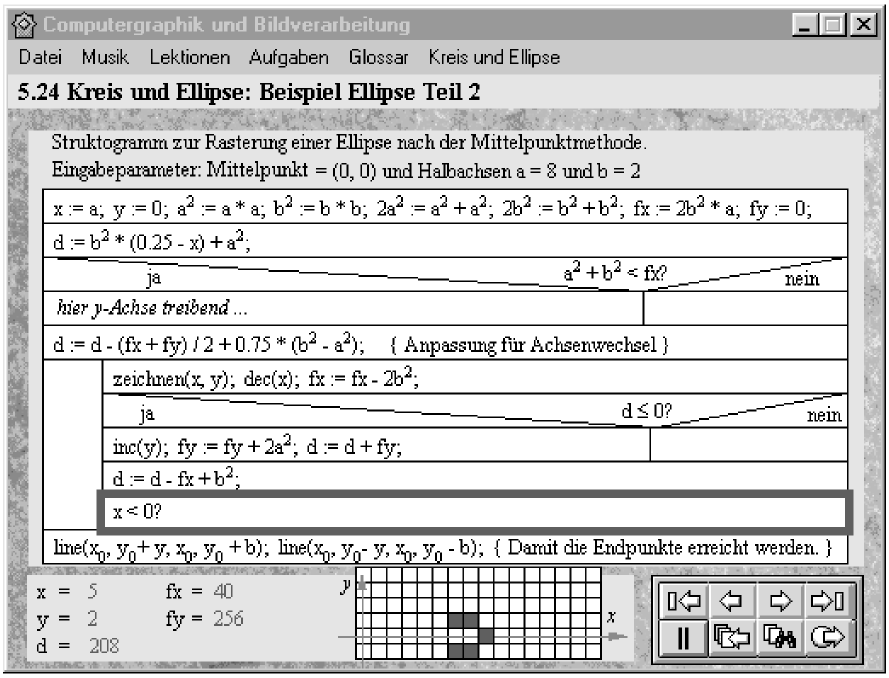

Abb. 2: Ellipsenrasterung in ViACoBi

Für die Animation eines Programms, das in einer Hochsprache geschrieben ist, bietet sich eine Darstellung in Diagrammform an, z. B. als Fluss- oder Nassi-Shneiderman-Diagramm. Ein animiertes rotes Fenster durchläuft das Struktogramm und zeigt dem Benutzer, an welcher Stelle und in welchem Zustand sich der Algorithmus gerade befindet. Die Darstellung wichtiger Daten, Variablen, Parameter, Zähler, Flags und des Stacks bei der Abarbeitung, in unserem Fall der Punkt (x,y), die momentane Richtung (fx,fy) und die Steuervariable d, dient dem Überblick und dem Verständnis des Benutzers. Zusätzlich sollte der Algorithmus synchron kommentiert werden.

Der Vorteil dieser Animationsform liegt in ihrer Allgemeingültigkeit. Alle Algorithmen einer Hochsprache können mit derselben Technik und den gleichen Darstellungsmethoden visualisiert werden. Dafür liefert das Flussdiagramm nur ein sehr abstraktes Bild der Funktionsweise des Algorithmus. Die Wirkung der einzelnen Programmteile auf die bearbeiteten Daten bleibt dem Benutzer weitgehend verborgen, da diese Animationsform nur ein logisches Bild des Algorithmus liefert. Die Darstellung ist von ihrer Natur her noch als statisch zu bezeichnen.

Erst bei der Animation der vom Algorithmus manipulierten Datenmenge wird dem Benutzer die Möglichkeit gegeben, den Algorithmus intuitiv zu begreifen. Die genaue Implementierung durch das zugrunde liegende Programm, die im allgemeinen nicht wesentlich ist, wird dem Benutzer hierbei in der Regel nicht offen gelegt.

Für die Animation von Daten unterscheiden wir drei Ebenen, die wir als *Standbild, Analogie* und *Bewegtbild* bezeichnen.

Standbilder als unterste Ebene der Visualisierung unterteilen wir nochmals in *Abbildungen* und *logische Bilder.* Eine *Abbildung* ist ein realitätsnahes, zumeist in digitaler Form vorliegendes Bild des dargestellten Objekts, das mit einem Scanner, einer Stillvideokamera oder einem Videorecorder und einer Framegrabberkarte digitalisiert wird. *Logische Bilder* bieten eine erste Form der Abstraktion; sie dienen der vereinfachten Darstellung komplexer Zusammenhänge in Form von Graphiken oder Diagrammen.

Eine *Analogie* als abstraktere Darstellungsform liegt vor, wenn die Wirkungsweise des Algorithmus auf seine Daten dargestellt, von der Beschaffenheit der Daten jedoch abstrahiert wird. Darüber hinaus werden Analogien in Form von Modellen verwendet, um z. B. nicht direkt beobachtbare Strukturen zu modellieren und so komplexe Sachverhalte darstellen zu können.

Die aufwändigste Form der Animation stellt das *bewegte Bild* dar; es zeigt die Wirkung des Algorithmus auf die Modelldatenmenge. In unserer Abbildung 2 wird der Ellipsenbogen in einer groben Pixelebene sukzessive erzeugt.

Bewegte Bilder entstehen, indem einzelne Objekte einer Szene animiert oder fertig digitalisierte Videofilmsequenzen eingebunden werden. Allerdings sind sowohl für die Erstellung von Animationen wie auch für die Produktion von Videosequenzen zunächst die Hardwarevoraussetzungen zu schaffen.

5 Hilfs- und Stilmittel

Ausgehend von einer Algorithmus-Beschreibung in Pseudocode werden Algorithmen normalerweise möglichst effizient programmiert, um die Laufzeit und ggf. den Speicherbedarf der Programme und damit die Kosten zu minimieren. Dabei werden spezielle Möglichkeiten der eingesetzten Programmiersprache oder der Hardware verwendet, die die Ideen des Algorithmus verwässern und für sein eigentliches

Verständnis nicht von Bedeutung sind. Um dem Lerner einen Algorithmus zu erläutern, ist jedoch Effizienz in dieser Form nicht notwendig. Die Algorithmen müssen nicht laufzeit- und aufgrund der zumeist kleinen Modelldatenmenge auch nicht speicheroptimiert werden.

Das bedeutet jedoch nicht, dass über Schwächen der Algorithmen für gewisse Parameter-Konfigurationen hinweggesehen werden sollte, sie sollten im Gegenteil in der Lehrsoftware herausgearbeitet werden. Als Beispiel sei hier der *Bresenham*-Algorithmus zur Kurvenrasterung genannt, der im Falle von Kreisen im allgemeinen eine bestmögliche digitale Approximation des reellen Kreises liefert, jedoch für einige wenige Radien einen Defekt an den Winkelhalbierenden aufweist.

Im Gegensatz zu der im Programmieralltag üblichen Zielrichtung, Programme möglichst schnell und effizient zu gestalten, ist es in Lehrprogrammen oft sogar notwendig, die Programme zu verlangsamen, damit jeder einzelne Schritt am Bildschirm nachvollzogen werden kann. Gegebenenfalls muss der Benutzer die Möglichkeit haben, im Algorithmus einen oder mehrere Rückschritte zu machen, um sich den Fortgang des Algorithmus nochmals zu verdeutlichen. Die Steuerung sollte hierbei immer dem Benutzer überlassen werden, damit dieser seine Arbeitsgeschwindigkeit frei wählen und beliebig innerhalb des Systems navigieren kann.

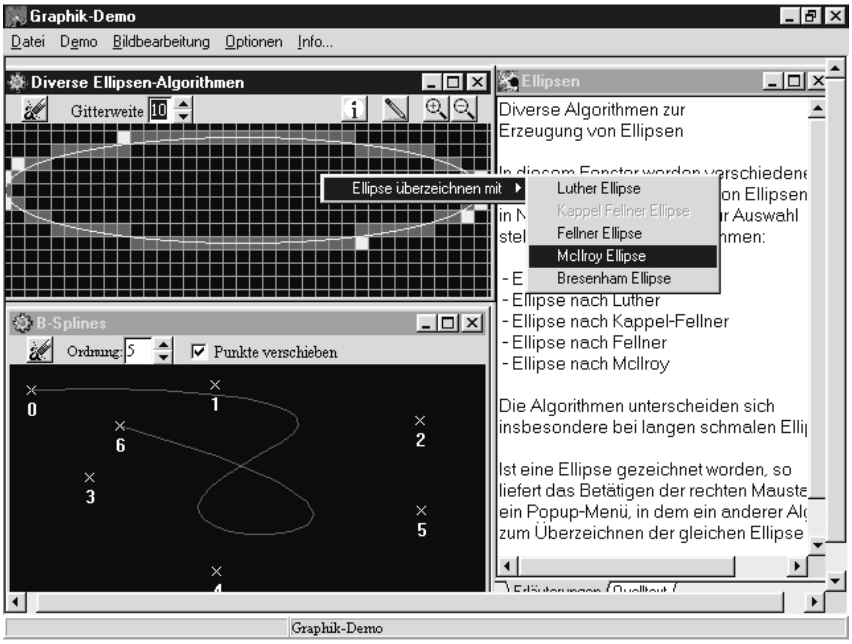

Abb. 3: Bildschirm aus dem Lerntool Graphik-Demo

Ein weiteres Stilmittel, das speziell auch bei der Animation graphischer Algorithmen eingesetzt wird, ist die Möglichkeit, Sachverhalte zu vergröbern. Bei der Animation eines Rastergraphik-basierten Algorithmus zum Zeichnen von Graphikprimitiven müssen die einzelnen Pixel stark vergrößert werden, damit der Benutzer überhaupt die

Möglichkeit hat zu sehen, wo die kritischen Punkte des Algorithmus liegen, wo der Programmierer Freiheiten nutzen kann und welche Pixel nach welchen Kriterien ausgewählt und gesetzt werden. Wichtige Details können zudem mit Mitteln der Form- und Farbgebung hervorgehoben werden.

Die einheitliche Färbung von wiederkehrenden Sachverhalten oder Objekten erleichtert dem Benutzer die Orientierung, signalisiert wichtige Zusammenhänge und löst Assoziationen aus. In dem mit Delphi entwickelten Tool Graphik-Demo (Abb. 3) ist die Rasterebene frei skalierbar. Durch verschiedene Farbgebung der Pixel können Ellipsen-Rasteralgorithmen in ihrer Funktionsweise durch Übereinanderlegung miteinander verglichen werden. Zugleich ist der Quelltext einsehbar. Eine andere Lektion betrifft B-Splines, die interaktiv durch Verschieben der Leitpunkte manipuliert werden können.

Einen ähnlichen Anspruch hat das OpenGL-basierte System MORPH [Ziegler 1998], das 2D-Morphing-Algorithmen unter einer intuitiven Benutzeroberfläche vorstellt und ihre Wirkungsweise mit den hier vorgestellten Methoden der Verlangsamung des Ablaufs, der Ausgabe und Visualisierung der Zwischenergebnisse und einer direkten Manipulierbarkeit der Parameter dem Lernenden näher bringt.

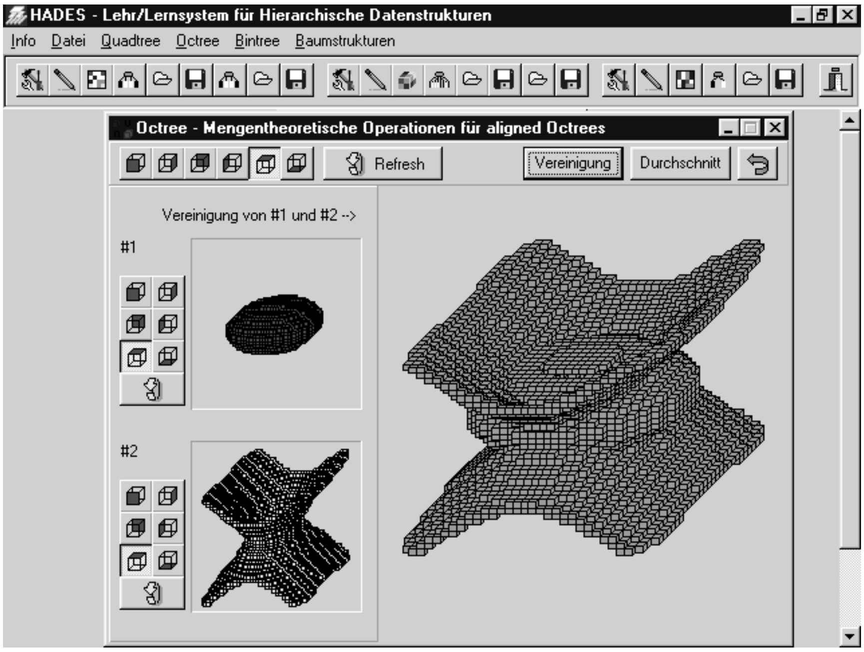

Abb. 4: Vereinigung zweier Octrees im System HADES

Bei dreidimensionalen Anwendungen stellt sich das Problem der Projektion. Hier ist eine geeignete Abbildung in die zweidimensionale Bildschirmebene zu wählen, so dass wichtige Inhalte nicht verloren gehen, die Darstellungsweise für den Betrachter aber auch nicht zu komplex wird. Falls notwendig, kann eine Darstellung aus verschiedenen Perspektiven in mehreren Ebenen (entweder nebeneinander oder bei kleineren Bildschirmen nacheinander bzw. überlagert) erfolgen. Davon wird in unserem mit Delphi entwickelten System HADES [Klüttermann 1997] Gebrauch gemacht, das die Datenstrukturen *Bintree*, *Quadtree* und *Octree* visualisiert und grundlegende Konstruktionen, wie Vereinigung, Schnitt, Translation und Rotation, Darstellung von Polygonen und Flächen zweiter Ordnung durch Quadtrees bzw. Octrees animiert und visualisiert.

Textrepräsentationen am Bildschirm sind für den Betrachter ermüdend zu lesen. Eine Auflockerung des Textes durch Bilder und Graphiken schafft hier bereits Abhilfe. Noch ausdrucksstärker ist die Einbindung von gesprochenem Text und ggf. Ton- und Musiksequenzen. Multimediale Technik kann auf den Einsatz von gesprochenem, erklärendem Text nicht verzichten.

Eine Animationssoftware wie ViACoBi stellt es dem Benutzer frei, mit gesprochenem oder geschriebenem Text oder mit beidem zu arbeiten. Bei der Animation wird gesprochener Text auf wichtige Stellen beim Ablauf des Algorithmus hinweisen und ergänzende Erläuterungen vorsehen. An geeigneten Stellen können auch Ton- und Musiksequenzen eingebunden werden, um die Aufmerksamkeit des Benutzers auf bestimmte Einzelheiten zu lenken, das Wiedererkennen gleicher Situationen zu fördern oder Abläufe leitmotivisch zu strukturieren und Eindrücke zu verstärken.

Eine noch bessere Akzeptanz wird erreicht, wenn digitales Video eingebunden wird, um Praxisbeispiele einzuspielen und den Lerner stärker zu motivieren. Videosequenzen sprechen mehrere Sinne gleichzeitig an und wirken auf das menschliche Auge nicht so ermüdend wie Fließtext, durch den sich der Betrachter durcharbeiten muss. Die Steuerung des Ablaufs einer Algorithmus-Animation sollte möglichst beim Benutzer selbst liegen. Er muss die Möglichkeit haben, Abschnitte zu wiederholen (Replay), Querverbindungen zum Quelltext, zu weiteren Erläuterungen, einem Glossar oder zum Aufgabenteil zu nutzen (Hypertext) und in Interaktion mit dem Rechner sein erarbeitetes Wissen zu evaluieren.

6 Andere Systeme zur Algorithmus-Animation

Zu den Systemen, die im Bereich Algorithmus-Animation oder Computergraphik und Bildverarbeitung existieren, gehören die Systeme BALSA [Brown 1988] zur Algorithmus-Animation, LINK [Berry 1997] und ein WWW-basiertes System [Lüssem, Sünderkamp 1997] zum Arbeiten mit Baum- und Graph-Algorithmen. Eine ähnliche Thematik verfolgt das GAIGS-System [Naps 1997], das Algorithmen mittels einer Sequenz graphischer Schnappschüsse erklärt, die benutzergesteuert angezeigt werden. GRAPHICA ist ein System, das von Shabo, Guzdial und Stasko zeitlich parallel zu ViACoBi entwickelt und evaluiert worden ist. Es ist ebenfalls Web-basiert und unterstützt kollaboratives Lernen an ausgewählten Beispielen.

An der Universität Freiburg wird unter der Leitung von Th. Ottmann seit 1995 an Systemen zum Thema *Authoring on the Fly* gearbeitet [Bacher et al. 1997]. In diesem Rahmen sind auch CD-basierte Kurse zu Algorithmen und Datenstrukturen entstanden, die Sortier- und Baumalgorithmen animieren. Ziel weiterer Forschungen ist es, die Animation nutzergesteuert ablaufen zu lassen und in jeder Phase persistente Annotationen zu erlauben.

Am Lehrstuhl von R.Wilhelm an der Universität Saarbrücken werden in Abschlussarbeiten, Dissertationen und Forschungsprojekten seit mehreren Jahren Lehr- und Lernsysteme mit Algorithmus-Visualisierung und -Animation zu den Themen Compilerbau und Programmiersprachen entwickelt und eingesetzt [Diehl et al. 1997, Kerren 1999]. Wesentliche Bemühungen gehen dahin, die Erstellung dieser Systeme zu automatisieren. Dazu wurde eine semi-automatische, Shape-basierte Programmanalyse eingeführt, die eine neue kondensierte Darstellung des Algorithmus in geclusterten Datenstrukturen eines Shape-Graphen mit aktiven und passiven Knoten ermöglicht. Dieser Ansatz erleichtert die Animation und Visualisierung klassischer Baum-basierter Algorithmen erheblich.

Unter der Leitung von J. Stasko sind die Systeme XTANGO, TANGO, POLKA und das Frontend SAMBA, das jetzt auch in einer Java-Version vorliegt, zur Algorithmus-Animation entwickelt worden. Das Ausgangssystem XTANGO erlaubt, aus dem Hochsprachenprogramm des Algorithmus wichtige Ereignisse zu isolieren und den Zustand der vom Algorithmus manipulierten Datenmenge in abstrakter Form zu visualisieren. Durch Animation der Einzelbilder kann die Wirkungsweise des Algorithmus illustriert und zu Lernzwecken eingesetzt werden. Eine umfangreiche Bibliographie befindet sich unter http://www.cc.gatech.edu/gvu/softviz/algoanim/.

MAPLE, Mathematica und MuPAD [Oevel et al. 1998] sind bekannte Computer-Algebrasysteme, die die Ausgabe von Hypermedia-Dokumenten mit Text, Graphiken, Formeln und Animationen ermöglichen. In den jüngsten Versionen können Funktionen und 3D-Objekte animiert werden, um Parameterabhängigkeiten zu visualisieren. Die verschiedenen Algorithmen sind unter Effizienzaspekten implementiert und besitzen Black-Box Charakter. Inkrementelle Vorgehensweisen beim Lösen von Aufgaben, Verlangsamung des Ablaufs und Skalierungen der Datenmengen sind nicht vorgesehen. Diese Programme sind daher nicht geeignet, die Wirkungsweise von Algorithmen zu vermitteln.

7 Ausblick auf Weiterentwicklungen und Web-basierte Lösungen

Wir haben Lehr- und Lernsysteme zur Computergraphik vorgestellt und zu ihrer Erstellung moderne Programmiersprachen, aber auch Autorensysteme wie Authorware oder Toolbook als Werkzeuge eingesetzt, die eine Aktualisierung und Erweiterung auch durch den Dozenten erlauben, der das Material verwendet. Allerdings bietet keines dieser Autorensysteme eine zufrieden stellende Integration leistungsfähiger Editoren, mächtiger Programmiersprachen oder formelmanipulierender Computer-Algebrasysteme. Daher müssen Hersteller solcher multimedialen Präsentationen auf die teure und ineffiziente Eigenentwicklung einzelner Komponenten zurückgreifen. So ist für eine Weiterentwicklung von ViACoBi die Integration einer leistungsfähigen Eingabeschnittstelle und des Computer-Algebrasystems MuPAD vorgesehen.

In [Hoppe et al. 1999] zeigen wir Möglichkeiten auf, wie Dozenten ihre Manuskripte unter Einsatz von Standard-Tools ohne großen technischen Aufwand in WWW-basierte Hypermediadokumente umsetzen können. Die Bereitstellung eines

Freihandtools auf einer elektronischen Tafel mit Stifteingabe ermöglicht im Gegensatz zu mit einer Präsentationssoftware erstellten elektronischen Folien einen schnellen Wechsel zwischen Präsentation und handschriftlicher Darstellung. Spontane Entwicklungen in der Vorlesung können automatisch in die vorbereiteten Dokumente eingebracht und später weiterverarbeitet werden. Ziel ist eine dynamische Darstellung der vom Dozenten entwickelten Herleitungen oder Algorithmen.

Dabei grenzen wir uns deutlich von CBT-Ansätzen, aufwändigen Multimedia- oder Authoring on the Fly-Produktionen mit der Betonung der Videokomponente auf Unix/MBone-Plattformen wie auch von Fernlernmaterialien ab. Den Lehrenden eröffnet sich ein Zugang mit niedriger Einstiegsschwelle zu moderner Informationstechnologie. Lernende erhalten aktuelle, mit multimedialen, interaktiven und dynamischen Anteilen angereicherte Hypermediadokumente zur Bearbeitung in der Präsenzphase und Weiterverwendung in einer individuellen Softwareumgebung.

Zur Anzeige von dynamischen Darstellungen bevorzugen wir Anzeigemodule (Viewer, Plug-Ins), z. B. für die dynamische XML-Sprache SMIL oder Java-Applets und Servlets, die Freihandeingaben direkt in das Skriptum integrieren.

So haben wir in Duisburg die Grundvorlesungen, Übungen und Praktika im Hauptstudiengang Lehramt Informatik und den Nebenfachstudiengängen zu den Informatischen Grundlagen Neuer Medien und Kommunikationstechniken, zur Informatik I mit Rechneraufbau, Rechnerstrukturen, imperativen, logischen und funktionalen Programmierparadigmen, zu den begleitenden Programmierkursen, zur Informatik II mit Algorithmen und Datenstrukturen, zur Theoretischen Informatik sowie einige weitere Vorlesungen aus dem Hauptstudium auf die neue Form umgestellt.

Beispielsweise werden die wichtigsten Sortieralgorithmen mit einem Applet visualisiert. Dabei sind sowohl die Anzahl der Elemente als auch die Geschwindigkeit parametrisiert, und es kann zwischen einem Fluss- und Schrittmodus hin- und hergeschaltet werden.

Dieser Ansatz wird zur Zeit im Projekt *InterActive Multimedia-Supported authoring for an Electronic Lecture hall* (AMSEL) weiterentwickelt.

8 Literatur

Bacher, Chr., Müller, R., Ottmann, Th., Will, M.: Authoring on the fly. In Hoppe, H. U. und Luther, W. (Hrsg.) 1997, 14-18.

Berry, J.: Improving Discrete Mathematics and Algorithms Curricula with LINK. ITiCSE’97, Uppsala, Sweden. SIGCSE/SIGCUE Proc. New York: ACM Press (1997), 14-20.

Brown, M. H.: Algorithm Animation. An ACM Distinguished Dissertation 1987. Massachusetts: The MIT Press (1988).

Diehl, S., Kunze, T., Placzek, A.: GANIMAM. Generierung interaktiver Animationen von abstrakten Maschinen. In Proc. Smalltalk und Java in Industrie und Ausbildung STJA’97, Erfurt 1997, 185-190.

GraphicaWebSide: http://www.cc.gatech.edu/gvu/multimedia/nsfmmedia/graphics/edulib/CSA.html

Hoppe, H.U., Luther, W. (Hrsg.): Informatik und Lernen in der Informationsgesellschaft. 7. GI-Fachtagung Informatik und Schule; Duisburg, September 1997. Reihe Informatik aktuell. Berlin: Springer (1997).

Hoppe, H.U., Luther, W., Mühlenbrock, M., Otten, W., Tewissen, F.: Interactive Presentation Support for an Electronic Lecture Hall - a practice report. ICCE'99, Chiba (Japan), November 1999. Advanced Research in Computers and Communications in Education. G. Cumming et al. (Eds.), IOS Press (1999), 923-930.

Janser, A., Luther, W.: Algorithmus-Animation als Hilfsmittel der Informatikausbildung. In Hoppe, H.U. und Luther, W. (Hrsg.): Neue Lehr-/Lernsysteme aus informatischer und fachdidaktischer Sicht. Duisburg: SM-DU-266 (1994), 20 S.

Janser, A., Luther, W., Otten, W.: Computergraphik und Bildverarbeitung. Braunschweig/Wiesbaden: Vieweg (1996), 360 S. mit Hybrid-CD.

Janser, A.: An interactive learning system visualizing computer graphics algorithms. ITICSE'97, Uppsala, Schweden 1997. SIGCSE/SIGCUE Proceedings New York: ACM Press (1997), 21-23.

Janser, A.: Ein Lehr-/Lernsystem zur Computergraphik und Bildverarbeitung mit unterschiedlichen Aufgabentypen. In Hoppe, H.U., Luther, W. (Hrsg.) 1997, 77-88.

Janser, A.: Entwurf, Implementierung und Evaluierung des interaktiven Lehr- und Lernsystems ViACoBi für die Visualisierung von Algorithmen der Computergraphik und Bildverarbeitung. Dissertation, Gerhard-Mercator-Universität Duisburg, Gesamthochschule. Berlin: Logos (1998), ISBM 3-89722-065-2.

Kerren, A.: Animation der semantischen Analyse. In A. Schwill (Hrsg.) Informatik und Schule. Fachspezifische und fachübergreifende didaktische Konzepte. 8. GI-Fachtagung Informatik und Schule; Potsdam, September 1999. Reihe Informatik aktuell. Berlin: Springer (1999), 108-119.

Klimsa, P.: Multimedia aus psychologischer und didaktischer Sicht. In Issing, L. und Klimsa, P. (Hrsg.): Information und Lernen mit Multimedia. Weinheim: Beltz Psychologie-Verlags-Union (1995), 7-24.

Klüttermann, K.: Grundlegende Algorithmen von Quadtrees und Octrees implementiert unter Delphi in Form eines Lehr-/Lernsystems. Duisburg: Diplomarbeit (1997).

Lüssem, J., Sünderkamp, St.: Einsatz von multimedialen Lernsystemen im Informatikunterricht. In Hoppe, H.U., Luther, W. (Hrsg.) 1997, 188-196.

Naps, T.L.: Algorithm Visualization on The World Wide Web - The Difference Java Makes! ITiCSE'97, Uppsala, Sweden. SIGCSE/SIGCUE Proc. New York: ACM Press (1997), 59-61.

Oevel, W., Postel, F., Rüscher, G., Wehmeier, S.: Das MuPAD Tutorium. Paderborn (Germany): SciFace Software (October 1998), ISBN 3-933764-00-9.

Rothenhöfer, D., Herzog, Chr.: Sypros - an intelligent tutoring system for parallel programming. In Chan, Tak-Wai (ed.). Proc. 1993 Int. Conf. on Computers in Education, Taiwan, Dec. 1993, 300-305.

XTANGO, TANGO, POLKA, SAMBA:
http://www.cc.gatech.edu/gvu/softviz/algoanim/.

Ziegler, K.: Ein Demonstrationssystem für verschiedene Morphing Algorithmen unter OpenGL. Duisburg: Diplomarbeit (1998).