OXO, Spacewar!, Adventure – Ein handlungsorientierter Ausflug in die Geschichte der Computerspiele

In diesem Beitrag werde ich dafür argumentieren, dass zentrale Konzepte, Methoden und Bildungsstandards der Informatik im Unterricht aus Computerspielen heraus entwickelt und dargestellt werden können. Darüber hinaus aber stehen Computerspiele als Kontext der Lebenswelt Jugendlicher den Schülerinnen und Schülern nahe, wodurch die Beschäftigung mit ihnen zu motivierenden Unterrichtsprojekten führen kann. Denn die Geschichte der Computerspiele führt uns zurück in die Geschichte der Computer, zu grundlegenden Herausforderungen und zu ihren Lösungen.
Drei Genres werden im Folgenden an drei bedeutenden Meilensteinen der Computerspielgeschichte beleuchtet: Strategie-, Action- und Adventurespiele.

OXO – Strategie auf dem EDSAC

1945 veröffentlichte John von Neumann die Ergebnisse der EDVAC-Forschungsgruppe unter dem Titel »First Draft of a Report on the EDVAC« (Neumann 1945). Dieses zunächst als internes Memo gedachte Papier verbreitete sich rasch und prägte den Begriff der *von-Neumann-Architektur* als Modell für zukünftige Computer-Generationen. Dass von Neumann als Berater erst relativ spät zum EEDVAC-Projekt stieß, wurde im Laufe der Zeit ebenso vergessen, wie die anderen Autoren, deren Ideen von Neumann so elegant zusammen fasste. Von Neumann wird später sagen, dass er nie beabsichtigt habe, seinen Artikel außerhalb der Moore School of Electrical Engineerung publik zu machen und dass er ansonsten aufmerksamer bei der Nennung der Autorenschaft gewesen wäre.
Wie auch immer die Umstände waren, die zur Veröffentlichung des Reports führten, er inspirierte eine Reihe von Computerentwicklungen, darunter die EDVAC (1949), JOHNNIAC (1953) in der RAND-Corporation, MANIAC (1952) in Los Alamos, ILLIAC (1951) an der Universität von Illinois, die SILLIAC (1953) in Sydney sowie die EDSAC (1949) in Cambridge, England.
Dieser von Maurice Wilkes und seinem Team gebaute *Electronic Delay Storage Automatic Calculator* gilt als erster frei programmierbarer Computer, der seine Programme im Speicher halten konnte. Die Beschäftigung mit diesem Gerät würde eine Reihe interessanter Hardware-Komponenten enthüllen, darunter Röhrenschaltungen, Lochkarten, Quecksilberverzögerungsleitungen als Arbeitsspeicher und Kathodenstrahlröhren als Display. Aber unser Schwerpunkt hier ist die Software oder besser: ganz spezielle Software.

1952 schrieb Alexander S. Douglas auf der EDSAC ein Programm zur Illustration seiner Doktorarbeit über Mensch-Maschine-Interaktion. Auf den Katho-

denstrahlröhren mit einer Auflösung von 35x16 Pixeln konnte der Nutzer gegen den Computer Tic-Tac-Toe spielen. Die Eingabe des Spielers erfolgte numerisch mit Hilfe einer Telefonwählscheibe, wobei die Zahlen 1-9 für je ein Kästchen standen. Sie wurde als Kreuz ‚X' auf dem Display ausgegeben. Der EDSAC reagierte auf die Eingaben mit einem Kreis ‚O' und spielte akkurat, was aus heutiger Sicht als ein Beispiele für Künstliche Intelligenz betrachtet wird. Die beiden Zeichen X und O bildeten den Namen des Spiels: ‚OXO', eines der ersten noch bekannten Computerspiele mit grafischem Output. Es ist immer noch spielbar im EDSAC-Emulator, den Martin Campbell-Kelly an der Universität Warwick geschrieben und veröffentlicht hat (Campbell-Kelly 1996).

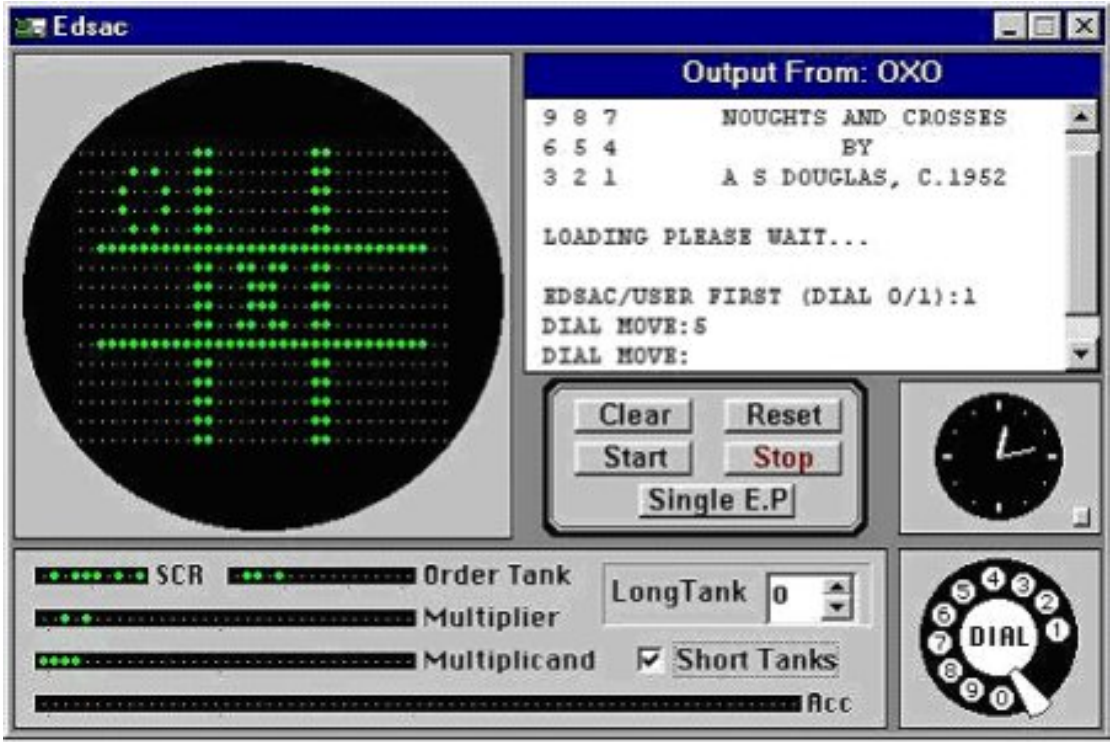

Abb.: OXO im Edsac-Emulator
Quelle: `http://img.index.hu/cikkepek/0506/tech//oxo.jpg`

Was es für den heutigen Informatikunterricht interessant macht, sind seine vielfältigen Verknüpfungsmöglichkeiten mit verschiedenen informatischen Themen:

1. Die EDSAC nimmt eine prominente Stelle in der Geschichte des Computers ein, ihre direkte Verwandtschaft mit der EDVAC und dadurch mit der ENIAC verbindet sie mit Namen wie John von Neumann, John Mauchly und Presper Eckert, aber auch mit den ENIAC-Girls und der Rolle der Frauen in der Informatik.
2. Die von-Neumann-Architektur hat viele Jahre die Computerhardware bestimmt und viele ihrer Merkmale finden sich auch in zeitgenössischen PCs, der gemeinsame Speicher von Programmen und Daten, das Rechenwerk, das Steuerwerk oder I/O-Werk.

3. Einen eigenen Algorithmus für Tic-Tac-Toe zu schreiben ist nicht trivial, aber auch nicht zu komplex für den (fortgeschrittenen) Unterricht. Vorbilder in zahlreichen Programmiersprachen sind im Netz verfügbar und können zur Analyse herangezogen werden. Grundgedanke ist die Berechnung eines Entscheidungsbaums, der keineswegs so groß ist, wie er zunächst erscheint. Denn für den ersten Zug gibt es nach Drehung und Spiegelung des Spielfelds lediglich drei Möglichkeiten, entsprechend gering sind die möglichen Antworten. Am Beispiel von Tic-Tac-Toe bzw. von OXO lassen sich grundlegende Verfahren der KI verdeutlichen, von Suchbaumbeschneidung und Knotenbewertung, von Eröffnungsbibliotheken und Minimax-Algorithmen. Seine überschaubare Komplexität macht Tic-Tac-Toe zum Musterbeispiel für die Modellierung kognitiver Prozesse im Computer. Dieser Umstand war der Grundlage des Plots für den Hackerfilm *Wargames* (1982) ist, der in den 80er Jahren eine ganze Generation von Jugendlichen and die Modems getrieben hat (Brown 2008).
   Von hier wäre problemlos ein Übergang in die Homecomputer-Szene der 70er und 80er möglich, in die Zeit von Apple, Commodore und Atari. Doch ist dies eine andere Geschichte, die ein anderes Mal erzählt werden soll.

Spacewar! – Action auf der PDP-1

Die ersten Computerspiele aus den 50er Jahren wie OXO, Bouncing Ball (s. Thielmann 2006, S. 19 f.; Webbox 2008) oder Nim (s. Godeve 2008) und mit Einschränkungen auch *Tennis for Two*, das weniger ein Computer- als ein Videospiel war, können auch im Rahmen eines Unterrichtprojekts noch relativ einfach nachprogrammiert werden, um den Blick in die Computer- und Spielgeschichte mit einer Einführung in grundlegende Spielmechanismen zu ergänzen. Spätestens mit dem 1962 am MIT geschriebenen Spiel *Spacewar!* wird dieses Vorgehen schwierig. Auf einem Vektorbildschirm werden zwei Raumschiffe von der Sonne in der Mitte angezogen und müssen sowohl versuchen, im Gravitationsfeld zu navigieren als auch sich gegenseitig abzuschießen. Ein Spieler gewinnt, wenn sein Gegner in die Sonne stürzt bzw. von einem Geschoss getroffen wird. Wie OXO kann auch Spacewar! in einem Emulator gespielt werden: `http://spacewar.oversigma.com/`

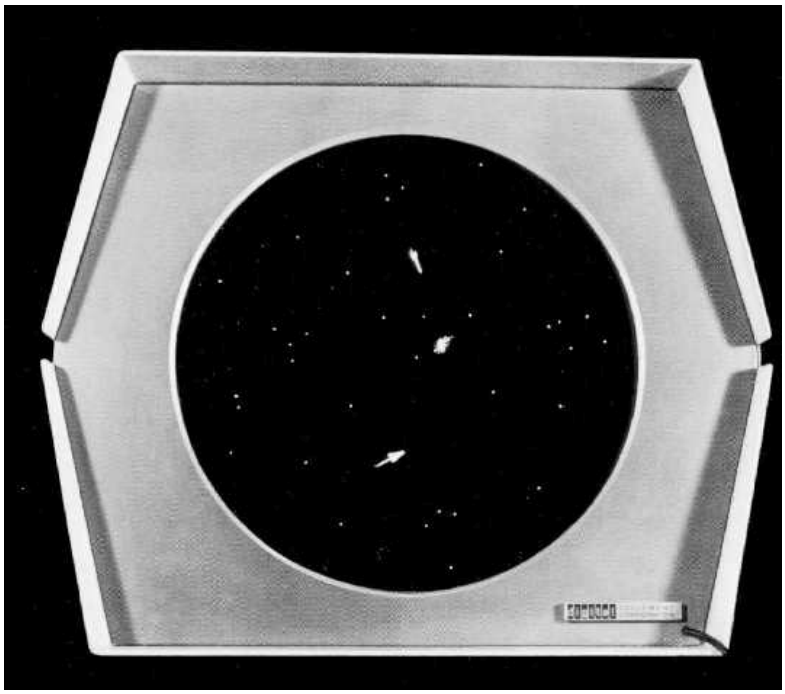

Abb. Spacewar!
Quelle:
http://sophia.javeriana.edu.co/%7Eochavarr/computer_graphics_history/historia/images/1961_SpaceWar.jpg

Geschrieben wurde dieses erste grafische Action-Spiel von einem Team um Stephen Russel an einer PDP-1 von der Firma DEC. Es konnte um verschiedene grafische Extras erweitert werden, z.B. um einen realistischer Sternenhimmel im Hintergrund. Dadurch eignete es sich zum Testen der Leistungskraft sowohl des Prozessors als auch des Bildschirms und wurde zu diesem Zweck von DEC an PDP-1 Kunden mitgeliefert. Anders als OXO oder Nim lief es auf einem Computer, der in Serie produziert wurde und war in den 60er Jahren sehr verbreitet. Viel ist über die PDP-1 erzählt worden, über ihre Ursprünge im legendären TX-0, der wiederum eine Transistorfassung des Whirlwinds war (Levy 1984; Schein 2003). Aber auch hier soll der Focus weiterhin auf den Spielen liegen. Wie lässt sich Spacewar! in den Unterricht einbinden?
Im Gegensatz zu OXO erweist sich eine Nachentwicklung als zu komplex. Neben Grafikroutinen sind hierfür auch Algorithmen zur Kollissionserkennung, zur Gravitation oder zur Echtzeitsteuerung der Raumschiffe notwendig. Eine aktuelle Java-Implementierung umfasst 3312 Zeilen, (Karumuru, McKenny 2000).
Dieses Los teilt Spacewar! mit den vielen Actionspielen. Auch einfache Genrevertreter dürften einen Programmieraufwand haben, der den durchschnittlichen Informatikunterricht überfordert.

Wer dennoch die grundlegenden Mechanismen dieser Spiele handlungsnah vermitteln möchte, kann auf ein Werkzeug zurück greifen, das die typischen Entwicklungsaufgaben vereinfacht. Prominent für den Einsatz im Schulunter-

richt ist Scratch, über das in der LOG IN bereits an anderer Stelle berichtet wurde (Stoll et al. 2008).

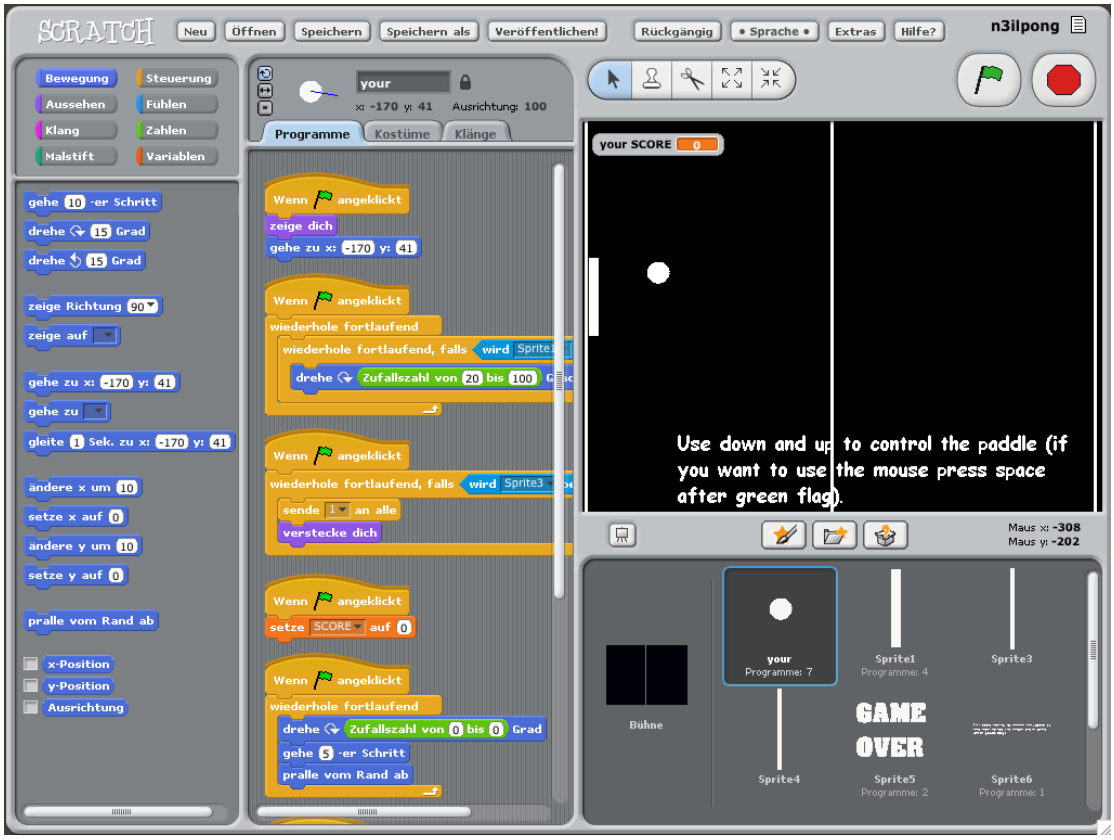

Abb. Scratch mit der Pong-Variante n3ilpong.

Scratch bietet eine leicht zu erlernende Umgebung an, in der selbst gezeichnete grafische Elemente (Sprites) mit vorgefertigten Bausteinen manipuliert werden können. Aussehen und Bewegung können dabei beeinflusst werden, aber auch ein Event-Modell ist integriert, mit dem Tastatur- und Mausevents behandelt werden oder die Position eines Sprites im Verhältnis zu einem anderen gesetzt werden kann. Zustände lassen sich in Variablen ablegen, womit den Programmen ein Gedächtnis zur Verfügung steht. Mir ist kein Nachweis bekannt, dass Scratch Turing-vollständig ist, aber es lassen sich eindrucksvolle Projekte auch nach kurzer Einarbeitungszeit und ohne Programmierkenntnisse bewältigen. Dennoch ist Scratch für ein Ballerspiel wie Spacewar kaum geeignet, weil es keine dynamischen Objekte erzeugen kann, was für die Schussfolgen der Raumschiffe unabdingbar ist. Dafür bieten sich andere Klassiker der Computerspielgeschichte an, in Scratch adaptiert zu werden. Allen voran der Initialzünder der Gaming-Industrie: Das unsterbliche Pong.
Nolan Bushnell versuchte sich 1971 an einer Adaption von Spacewar!, die er unter dem Titel *Computer Wars* mit mäßigem Erfolg veröffentlichte. Ein halbes Jahr später kam er mit Ralph Baers Spielekonsole *Odyssey* in Kontakt und adaptierte das darauf zu spielende Computertennis in einem eigenen Schaltkreis.

Ende 1972 veröffentlichte er *Pong* es als Arcade-Automat, der das erste Computerspiel im öffentlichen Raum wurde und Bushnells neu gegründeter Firma Atari riesige Gewinne einbrachte. Weitere Arcade-Spiele auch anderer Hersteller folgten, darunter Breakaout (1976), Space Invaders (1978), Galaxian (1979), Donkey Kong (1981) oder Pac Man (1981), die Atari von den Automaten in mehr oder weniger gelungenen Adaptionen auf seinem Home Entertainment System VCS 2600 holte.
Atari dominierte den Computerspielmarkt der frühen 80er Jahre bis zu seinem Zusammenbruch nach Weihnachten 1983. Spiele minderer Qualität wie die Adaption zum Film *E.T.* sowie der wachsende Homecomputer-Markt sorgten dafür, dass Kunden sich zunehmend gegen Spielekonsolen und für allgemeine Computer entschieden, wodurch der noch jungen Industrie der Markt wegbrach. Erst Ende der 80er Jahre gewannen reine Spielecomputer in Europa und USA wieder an Boden und sind seither mit Konsolen von Nintendo und Sega, später auch von Sony und Microsoft in Kinderzimmern präsent.

Während die einfachen Spiele wie Pong oder Breakout noch mit Scratch adaptierbar sind, leidet Space Invaders bereits deutlich unter den Einschränkungen der Plattform. Komplexere Spiele aus den 80er Jahren dürften den Unterrichtsrahmen sprengen. Dennoch müssen sie nicht gänzlich ausgeschlossen werden.
Die authentischste Rezeptionsform für historische Computerspiele im Unterrichtskontext ist der Gebrauch eines Emulators. Das Projekt *Multiple Arcade Machine Emulator* (M.A.M.E.) bietet seit 1997 unter `mamedev.org` eine Emulation der gängigen Arcade-Automaten an, die sich mit den Originalprogrammen in Form von ROM-Images ausführen lassen.

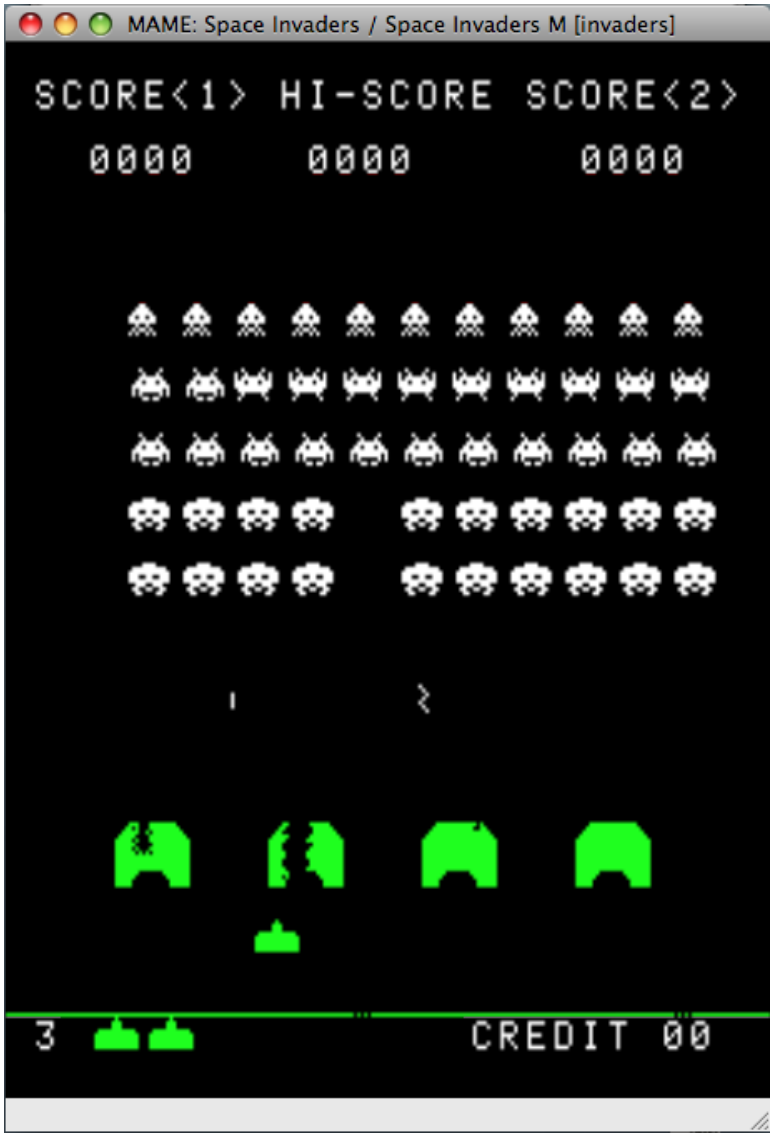

Abb. M.A.M.E mit Space Invaders. Aus lizenzrechtlichen Gründen müssen ROM-Images getrennt vom Emulator installiert werden.

Zwar sind auf einem Desktoprechner der Sound, die Bildschirmgröße, Schnittstellen und damit das Spielgefühl gänzlich anders als im Original, dennoch lässt sich auch an Emulationen noch die Faszination vermitteln, die von manchen Spielen ausging und damit verdeutlichen, dass ein gutes Spiel keineswegs von fotorealistischer 3D-Grafik und bombastischem Soundtrack bestimmt wird, sondern vielmehr von einer gut umgesetzten Spielidee. Derartige Ausflüge in die Geschichte eines Mediums können eine gelungene Auflockerung des Unterrichts darstellen oder Ausgangpunkt für die Besprechung von Computerhardware am Beispiel der Einstellungen eines Emulators sein. Der bereits erwähnten EDSAC-Emulator lädt den Programmcode in *Tubes*, Quecksilberverzögerungsleitungen, in denen 578 Bit/Röhre in Form von Schallwellen codiert wurden, die in einer Endlosschleife zwischen einem Lautsprecher auf der einen und einem Mikrofon auf der anderen Seite das Quecksilber durch-

rasten. Eine aus heutiger Sicht kaum vorstellbare Lösung für das Problem, wie man viele Bits in einem schnellen Arbeitsspeicher verfügbar halten kann.
Wie immer hilft der Blick in die Vergangenheit, die Errungenschaften der Gegenwart in einem neuen Licht zu sehen und nicht fraglos hinzunehmen, sondern als Prozess zu begreifen, bei dem verschiedene Lösungen angeboten, probiert und abgewogen werden.

ADVENTURE: Interaktive Geschichten unter UNIX

Neben Strategie- und Actionspielen stellen Adventures die dritte große Kategorie der Computerspiele dar. Das erste seiner Art wurde 1975 von William Crowther geschrieben, der in seiner Freizeit mit seiner Frau Patty Crowther Höhlen erforschte und nach seiner Scheidung die gemeinsamen Ausflüge in die Mammoth-Höhlensysteme für seine Töchter in einem Computerspiel aufarbeitete, in das er Elemente des Rollenspiels *Dungeons & Dragon* einfließen ließ (Nelson, 1998, S. 343 ff.). Er schrieb eine Textbeschreibung mit teilweise präzisen geologischen Angaben der verschiedenen Höhlenabschnitte, zwischen denen der Spieler sich durch Eingabe einfacher Textkommandos bewegen konnte. Das *Colossal Cave Adventure* oder kurz *Advent* begrüßte den Spieler mit den berühmten Sätzen:

```
YOU ARE STANDING AT THE END OF A ROAD BEFORE A SMALL
BRICK BUILDING. AROUND YOU IS A FOREST. A SMALL
STREAM FLOWS OUT OF THE BUILDING AND DOWN A GULLY.
```

Was durch die ausschließliche Verwendung von Versalien aus heutiger, durch Chat geprägten Sicht etwas `GESCHRIEHEN` aussieht, erklärt sich aus dem Umstand, dass die PDP-10 keine Kleinbuchstaben kannte.
Don Woods ergänzte 1976 den Programmcode um Rätsel, Gegenstände, Charaktere und Orte. Jim Gillogly portierte das Programm von Fortran auf C, so dass es für alle Unix-Rechner zur Verfügung stand. Und nicht zuletzt durch die Verbreitung im neuen Internet wurde Advent neben Spacewar! innerhalb kürzester Zeit zu einem der erfolgreichsten Computerspiele der späten 70er Jahre. Im Anschluss entwickelte sich das Spielgenre *interactive fiction* über wegweisende Titel wie Zork (1980), Mystery House (1980), King's Quest (1984), Manic Mansion (1987) bis zu Myst (1993) zunehmend vom Text- zum grafischen Adventure, das seine Geschichten inzwischen nahezu ausschließlich im Medium des Bildes und des Films erzählt. Durch die seit Manic Mansion etablierten Point-and-Click Interfaces wurden Texteingaben durch Auswahlmenüs und anklickbare Gegenstände ersetzt.
Zu erwähnen sei noch, dass Crowthers Originalcode im Jahr 2007 in einem Backup von Don Woods Schulaccount wieder gefunden wurde, was nicht nur für Adventure-Spieler ein echter Schatz ist, der dank seines digitalen Charakters mit jedem Interessierten geteilt werden kann: Der Fortran-Code steht zum freien Download zur Verfügung. Allerdings mangelt es inzwischen an

Compilern, die diesen Dialekt noch verstehen und verarbeiten können. Das ist das Drama digitaler Kulturgüter: hier sind Softwarearchäologen gefragt, die den historischen Quellcode wieder ein interaktives Erlebnis verwandeln können.
Die weitaus populärere Variante von Don Woods lässt sich hingegen seit 1977 in unzähligen Implementierungen spielen, darunter auch in einer deutschen Fassung bei
`http://www.ifiction.org/games/play.phpz?cat=&game=235&mode=html`

Empfehlenswerter als ein rein rezeptionsorientierter ist aber auch hier ein konstruktionsorientierter Umgang mit dem Genre. Denn Auch in diesem Fall lassen sich die grundlegenden Spielmechanismen am Besten durch einfache Projekte nachvollziehen, die neben der Förderung des historischen Bewusstseins im wahrsten Sinne des Wortes spielerisch zentrale informatische Prinzipien vermitteln. Der Vorteil von Adventures gegenüber anderen Genres ist ihre stufenlose Erweiterbarkeit. Zwar werden professionelle Spiele dieser Art von Anfang bis Ende durchgeplant, ehe sie implementiert werden, es spricht aber nichts gegen eine organische Entwicklung, die als Ergebnis vielleicht weniger rund ist, sich dafür aber am Kompetenzstand der Entwickler ausrichten lässt.

Das Projekt für den Informatikunterricht besteht also darin, ein eigenes Textadventure zu konzipieren und zu entwickeln. Ein solches Spiel besteht zumindest aus den folgenden Elementen:

1. Ein System von Räumen, die untereinander verbunden sind. Jeder Ort hat einen oder mehrere Ausgänge, die in einen anderen Raum führen.
2. Gegenstände, die an Räumen gefunden werden und teilweise nur betrachtet, teilweise mitgenommen und teilweise manipuliert werden können.
3. Ein Spieler, der durch das Spielfeld navigiert und dabei Gegenstände aufnehmen, bei sich tragen und benutzen kann.
4. Ein Vokabular, mit dem der Spieler Handlungen ausführen kann. Dazu gehören Bewegungen, Ansichten, aufheben und ablegen von Gegenständen, sowie Anwendungen von Gegenständen auf andere.
5. Eine Spielschleife, in der die Raum- und Objektbeschreibungen ausgegeben und die Spielereingaben verarbeitet werden. In dieser Schleife bleibt das Spiel bis zum Ende oder Abbruch.
6. Einen Parser, der Benutzereingaben verarbeitet und eine entsprechende Ausgabe produziert.

Während die prozeduralen Sprachen der 70er und 80er Jahre noch endlose READ und DATA Ketten zur Initialisierung der globalen Umgebungsvariablen benötigten, lassen sich in moderneren Ansätzen die Spielelemente in Objekten kapseln:

1. Ein Raumobjekt hat als Eigenschaften eine Beschreibung, eine Menge von möglichen Ausgängen, die zu anderen Räumen führen, eine Liste der Objekte, die sich an diesem Ort befinden. Als Methoden gibt es Antwort auf mögliche Aktionen, die an und mit diesem Ort ausgeführt werden können.
2. Gegenstände haben Namen, Kurz- und Langbeschreibungen, Zustände sowie Antworten auf Aktionen, die mit ihnen ausgeführt werden und die ihrerseits den Zustand dieses oder eines anderen Objekts ändern.
3. Das Spielerobjekt speichert den Aufenthaltsort und das gegenwärtige Inventar des Spielers sowie seinen Zustand, wenn der sich im Laufe des Spiels ändern sollte.
4. Das Vokabular bestimmt sich aus den Eigenschaften und Methoden, die von den Orten und Gegenständen exportiert werden.
5. Die Spielschleife ist die Hauptroutine der Form
   WHILE nicht_gewonnen DO {
   warte auf Spielereigabe
   verarbeite Spielereingabe
   gib Ergebnis aus
   }
   Ein Interaktionszyklus besteht aus einer Spielereingabe, ihrer Verarbeitung und der Ausgabe des Ergebnisses.
6. Der Parser ist Teil der Spielschleife. Je flexibler er programmiert ist, desto mehr Eingaben versteht das System und desto immersiver gestaltet sich die Interaktion.

Ein weiteres wichtiges Element ist die Speicherung des Spielstands, bei dem alle Objekte in ein persistentes Format überführt werden, sei es als Serialisierung in einer Datei oder als Abbildung in eine Datenbank. Dabei kommen aber fortgeschrittene Konzepte zum Einsatz, die erst eingeführt werden sollten, wenn sich der Bedarf nach Speicherung aus dem Spielspaß heraus geradezu aufdrängt.
Der stufenweise Aufbau besteht darin, dass zunächst nur die Spielschleife mit einigen Raum- und dem Spielerobjekt implementiert werden, so dass einfache Navigation aber noch keine Interaktion möglich ist. Dies entspricht in etwa der Intention von Crowthers *Colossal Cave Adventure*, die im Schwerpunkt eine Beschreibung für virtuelle Touristen war. Erst durch den Einsatz von Objekten können interessante Rätsel gestellt und echte interaktive Fiktionen erzählt werden, bei dem sich die Umgebung durch den Spieler verändern lässt.

Was hier nur in wenigen Sätzen angedeutet ist, gestaltet sich in der tatsächlichen Umsetzung als Folge kniffliger aber durchaus lösbarer Probleme.
Ein kurzes Tutorial für die Sprache Java findet sich bei javacoffeebreak:
http://www.javacoffeebreak.com/text-adventure/

Alternativ kann einer der zahlreichen Adventure-Generatoren verwendet werden, wie z.B. *inform* von Graham Nelson oder die deutsche Übersetzung *deform*, die bei textfire.de, dem deutschen e-zine für Textadventures, erhältlich ist. In einer Skriptsprache können Räume, Gegenstände, Nicht-Spieler-Charaktere und Eventtrigger definiert werden, die von einer generischen Spielschleife verarbeitet werden. Die Programmbibliothek schirmt den Entwickler vor den Implementierungsdetails ab, wodurch er mehr als Autor arbeiten und sich dabei auf das Spielkonzept, auf die Beschreibungen und Interaktionen konzentrieren kann. Zwar lassen sich damit deutlich schneller komplexere Textadventures entwickeln als mit Handarbeit, die informationstechnischen Grundlagen bleiben aber verborgen. Diese erschließen sich erst bei der eigenen Entwicklung. Dazu gehören:

- Schleifen und bedingte Anweisungen
- Variablen
- Text Eingabe, -verarbeitung und -ausgabe
- Objekte, Attribute, Attributwerte und Methoden
- Kapselung, Schnittstellen
- Zustandsmodellierung von Objekten als endliche Automaten
- Parsingmodelle mit Darstellungsmitteln formaler Sprachen
- Kartenmodellierung mit Hilfe von Graphen

Dies alles sind zentrale informationstechnische Konzepte, die zu vermitteln Ziel des Informatikunterrichts ist. Ohne Übertreibung lässt sich behaupten, dass wesentliche Bereiche der Grundsätze und Standards (Puhlmann 2008) bei der Entwicklung eines Textadventures gefordert und gefördert werden. Begleitet werden kann dies von einem Ausflug in die Geschichte der Computerspiele, die bis heute interaktive Fiktionen in verschiedenen literarischen Stilrichtungen produziert. Die Textbeschreibungen fördern auch die sprachliche Ausdrucksfähigkeit in deutscher und wahlweise auch englischer Sprache. Eine echte Herausforderung ist z.B. die Adaption einer literarischen Vorlage.
Wird das gesamte Projekt als Gruppenarbeit angelegt, kommen zusätzlich noch soziale Kompetenzen ins Spiel. Nicht zu vernachlässigen ist der Spielspaß, wenn die fertigen und halbfertigen Produkte getestet und miteinander verglichen werden.

Fazit

Die handlungsorientierte Beschäftigung mit der Geschichte der Computerspiele im Informatikunterricht kann zu motivierendem, vieldimensionalen, standardorientiertem und methodisch vielfältigen Unterrichtsprojekten führen, die darüber hinaus noch fächerübergreifend gestaltbar sind.

Computerspiele sind als Leitmedium (so Kulturstaatsminister Bernd Neumann bei der Vorstellung des Medien- und Kommunikationsberichts 2008) direkt mit der Lebenswelt der Jugendlichen verbunden, die erfahrungsgemäß sehr positiv darauf reagieren, wenn ihre Begeisterung für Spiele nicht als eine entweder sinnlose im Zweifelsfall aber verrohende Zeitverschwendung wahrgenommen, sondern in Form ernsthafter Auseinandersetzung akzeptiert wird.
Die Kombination verschiedener lebensweltlicher Aspekte, z.B. der technischen, historischen und ästhetischen Perspektive, kennzeichnet den vieldimensionalen Unterricht (Koubek 2005). Bei der Entwicklung von Textadventures werden grundlegende informationstechnische Konzepte gestärkt, die den Schwerpunkt der Bildungsstandards bilden. Aber auch der Blick in die Geschichte des Computers und in die Ästhetik monochroner Textabenteuer fördert Einsichten in andere Modi der Weltbegegnung.
Methodische Vielfalt kann in Form von Referaten, interaktiven Rezeptionsformen, Entwicklungsprojekten oder Recherchephasen gewährleistet werden. Ein handlungsorientierter Ausflug in die Geschichte der Computerspiele ist damit kontextorientierter Informatikunterricht (Koubek/Schulte/Schulze/Witten 2009), der großen Wert darauf legt, dass bei der Betrachtung der Kontexte die informationstechnischen Kerninhalte nicht zu kurz kommen. Diese werden im vorliegenden Kontext spielerisch in einem Projekt erarbeitet, das im besten Fall die Umsetzung einer eigenständigen Spielidee darstellt, wodurch der Computer als kreatives Ausdrucksmedium des individuellen Gestaltungswillens erfahrbar wird.

Quellen

Brown, Scott (2008): *WarGames: A Look Back at the Film That Turned Geeks and Phreaks Into Stars.* Wired Magazine 16.08, August 2008. Internet (Mai 2009):
http://www.wired.com/entertainment/hollywood/magazine/16-08/ff_wargames

Campbell-Kelly, Martin (1996): *The Edsac Simulator.* Internet (Mai 2009):
http://www.dcs.warwick.ac.uk/~edsac/

Cohen, D. S.: *Cathode-Ray Tube Amusement Device – The First Electronic Game.* Internet (Mai 2009):
http://classicgames.about.com/od/classicvideogames101/p/CathodeDevice.htm

Godeve, Pete: *NIMROD.* Internet (Mai 2009):
http://www.goodeveca.net/nimrod/

Karumuru, Venu; McKenny, Paul (2000): Spacewar Game. Internet (Mai 2009):
http://www.rdrop.com/users/paulmck/projects/spacewar/Project/SpaceWar.htm

Koubek, Jochen (2005): *Informatische Allgemeinbildung*. In: Friedrich, J. (Hrsg.): Unterrichtskonzepte für informatische Bildung. Dresden, 2005, S. 57-66.

Koubek/Schulte/Schulze/Witten (2009): *Informatik im Kontext. Ein integratives Unterrichtskonzept für den Informatikunterricht*. Unveröffentlicht.

Levy, Steven (1984): *Hackers: Heroes of the Computer Revolution*. Doubleday.

Nelson, Graham (2001): *The Craft of Adventure*. In: Nelson: *The Inform Designer's Manual*. Vierte Auflage, Kapitel 8. Internet (Mai 2009): http://www.inform-fiction.org/manual/Chapter8.pdf

Neuman, Johann von (1945): *First Draft on the Report on the EDVAC*. Internet (Mai 2009)
http://www.virtualtravelog.net/entries/2003-08-TheFirstDraft.pdf

Puhlmann, Hermmann (Hrsg.) (2008): Grundsätze und Standards für die Informatik in der Schule. In: LOG IN, 28. Jg. (2008), Heft Nr. 150/151.

Schein, Edgar (2003): *DEC is dead, long live DEC. The lasting legacy of Digital Equipment Corporation*. San Francisco: Berrett-Koehler.

Stoll, Thomas; Thalheim, Klaus; Timmermann, Bettina (2008): *Die Katze im Computer Das Programmierwerkzeug SCRATCH im Informatikunterricht der Sekundarstufe I*. In: LOG IN 154/155. S. 81 ff.

Thielmann, Tristan (2006): *Eine Mediengeschichte des Displays*. In: Thielmann; Schröter (Hrsg.): Navigationen. Display I, S. 13-30.

Webbox 2008: Whirlwind „bouncing Ball“ demo program by Charly Adama. http://www.webbox.org/cgi/1949-50%20Whirlwind%20applications.html

Prof. Dr. Jochen Koubek
Universität Bayreuth
Digitale Medien
jochen.koubek@uni-bayreuth.de