III. SEMIOTISCHE ANALYSE – SIMULIERBARKEIT

1. Digitalisierte Zeichen

1.1 Maschinelle Zeichenverarbeitung

Der Zusammenhang zwischen Formalisierung und Mechanisierung, der bereits von Gottfried W. Leibniz erkannt wurde und ihn zur Konstruktion von Rechenmaschinen veranlaßte, ist die Voraussetzung der Computerentwicklung. Leibniz hat nicht nur ein Modell einer Rechenmaschine zur Ausführung aller vier Grundrechenarten entworfen, die eine *Lebendige Rechenbanck* sein sollte,[1] er hat sich vor allem Gedanken über die zu verarbeitenden Zeichen gemacht. Sein Vorschlag einer dualen Darstellung der Zahlen und dementsprechend formulierten Regeln zur Ausführung der Grundrechenarten sind das Ergebnis seiner Bemühungen.[2] Der Nachteil der Maschine von Leibniz wie auch anderer besteht in der Spezifizierung für einen definierten Bereich von Operationen, indem der Mechanismus zur Ausführung der Operationen die Maschine selbst ist. Oder entsprechend dem Computerjargon gesprochen: Es handelt sich um *festverdrahtete* Operationen, wie sie noch für die unter Mitarbeit von Alan Turing gebauten Rechner *Robinson* und *Colossus* typisch sind.[3] Die Abstraktionsleistung, die vollbracht werden muß, um einen allgemein programmierbaren Computer zu entwerfen, besteht darin, den Vorgang der Zeichenverwendung selbst zu mechanisieren, um dann jedes effektive Rechenverfahren simulieren zu können. Erst dann stellt sich auch die Frage, welche Operationen überhaupt berechenbar sind. Um also eine universelle Maschine zu entwerfen, bedarf es einer Strukturierung der Zeichenverwendung, die elementar, determiniert und endlich ist - so wie Turing es mit seinem Konzept einer automatischen Maschine vorschlägt, für welche die Zeichenverwendung im Beschriften, Löschen oder Überschreiben eines Feldes und in der Bewegung nach links oder rechts gemäß vorge-

---

[1] „... *so ich eine Lebendige Rechenbanck nenne, dieweil ein Wort dadurch zu wege gebracht wird, daß alle Zahlen sich selbst rechnen, addiren subtrahiren multipliciren dividiren*... „ Leibniz, G.W.: Sämtliche Schriften und Briefe, 1923ff, II/1, S. 160. Diese Maschine besteht aus einem Zählwerk aus Staffelwalzen mit achsenparallelen Zähnen. Das Rechnen vollzieht sich dann als mechanischer Vorgang, der durch eine Kurbel angetrieben wird. Vrgl. Mainzer, 1995, S. 32ff. Vor Leibniz haben sich vor allem Wilhelm Schickard und Blaise Pascal mit dem Bau von Rechenmaschinen befaßt. Allerdings gelingt es erst Leibniz alle vier Grundrechenarten zu mechanisieren, indem er die Multiplikation auf die Addition und die Division auf die Subtraktion zurückführt. Seine Rechenmaschine basiert jedoch auf dem Zehnersystem, da das Dualsystem die feinmechanischen Fähigkeiten seiner Zeit überforderte.

[2] „*Ebenso sei erwähnt F. Bacon ... der 1623 ein „Zwei-Buchstaben-Alphabet" für Geheimschriften vorschlägt und T. Hariot (1560 - 1621), in dessen Nachlaß bereits vier Grundoperationen des dualen Zeichensystems behandelt sind*." Mainzer, 1995, S. 38

[3] Colossus hatte eine klar umgrenzte Aufgabe, er diente der Entschlüsselung des Enigma-Kodes.

gebener Instruktionen besteht, sowie eines Mediums, welches das entsprechende Symbolsystem zur Verfügung stellt. Spricht Turing in seinem logisch-mathematischen Konzept von Zeichen eines Alphabets, so müssen diese Zeichen derart spezifiziert sein, daß eine Maschine diese verwenden und umformen kann. Diese Art von Zeichen können jedoch nur Zustände der Maschine sein, die ein geeignetes syntaktisches wie semantisches Schema aufweisen.[4] Da eine Maschine lediglich interpretationsfrei mit den Symbolen bzw. Zuständen agieren kann, muß das semantische Schema entsprechenden Kriterien genügen, wie sie typisch für Werte sind, d.h. es muß sich um ein Schema handeln, das formal-operativ erzeugbar ist und im Grunde den syntaktischen Anforderungen der Differenziertheit Rechnung trägt, die ansonsten in der visuellen Gestalt der Zeichen kodiert ist. Der Begriff semantisch ist hier also in einem anderen, intrasymbolischen Sinne zu verstehen als in Bezug auf sinnhafte Zeichen mit extrasymbolischer Bezugnahme.

Wie sieht nun die tatsächliche Umsetzung der Mechanisierbarkeit des Vorgangs des Rechnens und Schreibens aus? Die physikalische Implementierung basiert auf der Diskretisierung elektrischer Zustände mit Hilfe von Schaltungen sowie der Realisierung elementarer Operationen mit diesen Schaltungen. Liegt an einem Eingang eines Schalters t elektrische Spannung an, so leitet der Schalter den Strom: am Ausgang liegt Spannung vor (t = L). Liegt keine Spannung an, so sperrt der Schalter (t = Ø).[5] Durch die Kopplung von Schaltungen lassen sich Gatter bilden, welche entsprechend den aussagelogischen Operationen ein Muster an Eingangs- und Ausgangszuständen bilden. Negationselement, UND- sowie ODER-Gatter sind die Grundbausteine der Schaltalgebra, aus welchen sich kompliziertere Schaltungen bauen lassen. Die Gatter verarbeiten die Operationen gemäß den Wahrheitstafeln der Konjunktion, Disjunktion und Negation. „*Mathematisch gesehen wird durch die Wahrheitstabelle eine Funktion y definiert, die in Abhängigkeit von ihren beiden Variablen a und b die Werte Ø und L annehmen kann*."[6] Die Funktionen lassen sich symbolisch schreiben: $f_K = a \wedge b$ für die Konjunktion, $f_D = a \vee b$ für die Disjunktion und $f_N = \neg a$ für die Negation. Die Rechenregeln für die Schaltalgebra bilden eine Boolesche Algebra, welche ein abgeschlossenes System zweier Operationen darstellt, für welches kommutatives, assoziatives, distributives und Verschmelzungsgesetz gilt und in welchem ein Nullelement, ein Einselement und zu jedem Element ein Komplement existiert.[7]

[4] In dem Modell von Leibniz waren die Zustände Zähne der Staffelwalzen, die entsprechend positioniert und von daher syntaktisch disjunkt und differenziert waren.

[5] Zu Beginn der Computerentwicklung nutzte man elektromagnetische Schalter (Relais), später Röhrenschaltungen und Transistoren und schließlich integrierte Schaltkreise auf Siliziumkristallplättchen (Chips). LSI-Chips weisen auf wenigen mm^2 tausende Gatter, VLSI-Chips über 100.000 Gatter auf. Damit sind Schaltungsgeschwindigkeiten im Nanosekundenbereich möglich.

[6] Schauer, H.: Computersysteme – Aufbau und Funktionsweise, 1976, S. 12

[7] Schaltalgebra:

	Rechenregeln der Konjunktion		Rechenregeln der Disjunktion	
	$Ø \wedge Ø = Ø$	$a \wedge L = a$	$Ø \vee Ø = Ø$	$a \vee L = L$
	$Ø \wedge L = Ø$	$a \wedge Ø = Ø$	$Ø \vee L = L$	$a \vee Ø = a$
	$L \wedge Ø = Ø$	$a \wedge a = a$	$L \vee Ø = L$	$a \vee a = a$

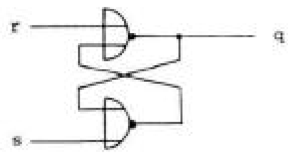

Abb. 6: Speicherelement

Schaltungen lassen sich mit Hilfe von Schaltfunktionen schreiben, und insgesamt gibt es sechzehn mögliche Schaltfunktionen für zwei Eingangsvariablen, wobei die NOR- und die NAND-Schaltung von besonderem Interesse sind. Beide Schaltungen sind zueinander dual und haben die Eigenschaft, jeweils alle anderen Schaltfunktionen darzustellen. Auf Basis dieser Grundelemente lassen sich komplexere Schaltungen bauen. Schaltet man beispielsweise zwei NOR-Gatter gegenseitig rückgekoppelt, so läßt sich auf diese Weise ein Speicherelement konstruieren. Liegt am set-Eingang des Speicherelements kurzzeitig Spannung an, so wird der Ausgang q = L. Fällt die Eingangsspannung auf Null zurück, bleibt die Spannung durch die Rückkopplung in der Schaltung erhalten. Liegt am reset-Eingang Spannung an, wird die Spannung am Ausgang Null q = Ø und zwar solange, bis an den set-Eingang Spannung angelegt wird. Ein solches Speicherelement heißt RS-Flip-Flop, und mit ihm läßt sich eine Dualziffer speichern.[8] Um die Zustandsänderungen in Flip-Flops zu koordinieren, wird deren Verhalten mit einem gemeinsamen Taktimpuls gesteuert, so daß taktgesteuerte Flip-Flops ein synchrones Netzwerk erzeugen. Mit Hilfe von Verzögerungselementen läßt sich ein Signal um den Abstand von zwei Taktimpulsen verzögern. Schaltet man eine Folge von Flip-Flops mit Verzögerungselementen aneinander, so erhält man ein Schieberegister, um eine n-stellige Dualzahl zu speichern.[9] Ein Register speichert Zahlwerte in Form von 0/1-Folgen. Die kleinste Einheit ist eine binäre Stelle (bit), die zwei binäre Werte darstellen kann. Mehrere Werte lassen sich speichern und nach rechts oder links verschieben.

	$L \wedge L = L$	$a \wedge \neg a = Ø$	$L \vee L = L$	$a \vee \neg a = L$
Kommutatives Gesetz	$a \wedge b = b \wedge a$		$a \vee b = b \vee a$	
Assoziatives Gesetz	$(a \wedge b) \wedge c = a \wedge (b \wedge c)$		$(a \vee b) \vee c = a \vee (b \vee c)$	
Distributives Gesetz	$(a \vee b) \wedge (a \vee c) = a \vee (b \wedge c)$		$(a \wedge b) \vee (a \wedge c) = a \wedge (b \vee c)$	
Verschmelzung	$a \vee (a \wedge b) = a$		$a \wedge (b \vee c) = a$	

In der Schaltalgebra gilt das Prinzip der Dualität, d.h. „*zwei Funktionen sind zueinander dual, wenn nach Komplementbildung der Variablen der einen Funktion das Ergebnis gleich dem Komplement der anderen Funktion ist.*" Schauer, 1976, S. 21
Konjunktion und Disjunktion sind zueinander dual und es gilt:

	$\neg(a \wedge b) = \neg a \vee \neg b$	$\neg(a \vee b) = \neg a \wedge \neg b$
De Morgan Theorem	$\neg(a \wedge b \wedge c \ldots) = \neg a \vee \neg b \vee \neg c \ldots$	$\neg(a \vee b \vee c \ldots) = \neg a \wedge \neg b \wedge \neg c \ldots$
Shannon Theorem	Es erlaubt das Komplement von Schaltfunktionen zu bilden, in welchen Konjunktion und Disjunktion gemischt auftreten. Die Klammerung legt die Reihenfolge der Operationen fest.	

[8] „*Der Ausgang q des RS-Flip-Flops ist nicht nur von den beiden Eingangsgrößen r und s, sondern auch vom momentanen Zustand q = Ø oder q = L des Flip-Flops abhängig.*" Schauer, 1976, S. 37. Wird an den Eingängen des RS-Flip-Flops gleichzeitig Spannung angelegt, ist der Folgezustand nicht definiert. Dies kann in einer Schaltung vermieden werden, die zwei UND-Gatter aneinander koppelt (JK-Flip-Flop) oder indem bei einem JK-Flip-Flop beide Eingänge kurzgeschlossen werden (T-Flip-Flop). Das T-Flip-Flop ist ein multistabiler, endlicher und determinsitischer Automat, mit einem Eingang, der alle Zustände in sich selbst überführt. Und multistabile Automaten sind grundsätzlich als Speicher geeignet. Unter einem Automaten versteht man einen Apparat oder eine Schaltung, der abhängig von den Eingangsgrößen von seinem aktuellen Zustand in einen anderen Zustand übergeht. „*Ein Automat heißt endlich, wenn er endlich viele Zustände annehmen kann, und deterministisch, wenn jeder Zustand durch den vorhergehenden Zustand und die Eingangsgrößen bestimmt ist.*" Schauer, 1976, S. 54

[9] „*Schaltet man den Ausgang des Schieberegisters in den Eingang zurück, so kann eine gespeicherte Dualzahl innerhalb des Registers rotieren. Schieberegister werden unter anderem häufig benutzt, um die Operanden und das Resultat arithmetischer Funktionen zu speichern*." Schauer, 1976, S. 42

Damit kommt ein Register der Vorstellung Turings von einem Band mit Feldern (bit) zur Notierung von je einem Zeichen (L) bzw. als Leerstelle (Ø) nahe. Mit den Gattern lassen sich weitere Schaltungen wie verschiedene Addierwerke (Akkumulatoren), Subtrahierwerke, Zähler, Vergleichswerke und mehr bauen. Die Operationen der Multiplikation und Division werden auf Addition und Subtraktion zurückgeführt, wie bereits von Leibniz vorgeschlagen.[10] Die aus den Gattern erzeugten Bauteile ergeben ein einfaches Rechnermodell: Mehrere Register werden als Speicher für Operanden genutzt, ein Rechenwerk (+, -, x, ÷) erlaubt die Realisierung der Grundrechenarten und weitere Register speichern die Ergebnisse. Um den Datentransfer zu regulieren, ist jedes Register mit einer Adresse zu versehen, und zu jeder Operation ist diese Adresse mit anzugeben. Die gewünschten Operationen sind in Einzelschritte (Befehle oder Instruktionen) zu zerlegen, die als Operationskodes geschrieben werden. *ADD R a* beispielsweise gibt an, den Inhalt des Registers R um den Inhalt der Speicherzelle a zu erhöhen (R := R + a), *STO R a* sorgt dafür, daß der Inhalt des Registers R auf die Speicherzelle a übertragen und damit gespeichert wird (a := R, store).[11] Um eine Berechnung auszuführen, bedarf es einer Folge von Anweisungen (Programm), welche die Operationen, Adressierungen und Registerinhalte (Daten) schrittweise in einer sinnvollen Reihenfolge koordinieren, sowie eines komplexeren Rechnermodells aus Programmspeicher, Steuerwerk, Rechenwerk und Arbeitsspeicher. Die Programme werden in einem separaten Programmspeicher in Form von Programmspeicherworten abgelegt, welche Informationen über die Art der Operation, die Registernummer sowie die Adresse enthalten. Ein Steuerwerk liest die Befehle aus dem Programmspeicher, schlüsselt diese zur Verarbeitung in den Operationsteil, die Registernummer und den Adressteil auf und leitet die Daten weiter.[12]

Alle aus mnemotechnischen Gründen verwendeten symbolischen Schreibweisen (Assemblerprogramm) müssen in eine binärverschlüsselte Form (Maschinenprogramm) transformiert werden. Dazu werden die symbolischen Operationskodes binärkodiert, Registerbezeichnungen und Dezimalzahlen als Dualzahlen dargestellt und Variablen an bestimmte Speicherzellen binärverschlüsselt adressiert. Da der Rechner permanent zwischen beiden Phasen wechselt, lassen sich Programm und Daten in

[10] Statt der Multiplikation einer p-stelligen Dualzahl wird eine p Addition durchgeführt. Die ganzzahlige Division (a ÷ b) mit Rest (a modulo b) wird in Einzelschritte zerlegt. Für die ersten Stellen des Dividenden wird geprüft wie oft der Divisor enthalten ist. Die ermittelte Anzahl liefert die erste Stelle des Ergebnisses. Der Rest des Dividenden wird um die nächste Stelle erweitert und geprüft, wie of der Divisor enthalten wird, usf. In einem binären System kann der Divisor nur Null-mal oder Ein-mal enthalten sein. Die Division ist also einfacher als im Dezimalsystem und kann durch einen Vergleich der Dualzahlen ermittelt werden. Zur Division benötigt man neben den Registern ein Subtrahierwerk und ein Vergleichswerk.

[11] Das Ergibtzeichen (:=) unterscheidet sich wesentlich von dem mathematischen Gleichheitszeichen (=). Das Ergibtzeichen drückt eine dynamische Operation aus und ist unsymmetrisch.

[12] „*Die automatische Abwicklung eines Befehls erfolgt somit in zwei Phasen. In der ersten Phase – der Instruktionsphase (engl. instruction cycle) – wird der Befehl aus dem Programmspeicher gelesen, und decodiert. ... In der zweiten Phase – der Ausführungsphase (engl. execution cycle) – wird der decodierte Befehl ausgeführt*." Schauer, 1976, S. 81. Es ist wichtig die Programmbefehle in aufeinanderfolgenden Speicherzellen abzulegen, denn dann können mit einem Befehlszähler die Adressen der Programmbefehle verwaltet werden, indem der Zähler jeweils um 1 erhöht wird. Damit ist das Programm vollautomatisch durchführbar und es bedarf lediglich eines Stop-Befehls, um das Programm abzubrechen.

einem Speicher ablegen und dadurch Bauteile einsparen. Damit ist die Hardware für unterschiedliche programmgesteuerte Aufgaben verwendbar. Die Effizienz von Programmen ergibt sich aus einer Strukturierung der Operationsabfolgen, die es ermöglicht, nicht nur Operationen linear nacheinander auszuführen, sondern mit Hilfe von Sprungbefehlen Teile des Programms oder Unterprogramme beliebig oft zu verwenden (Schleifen).[13] Damit erhöht sich jedoch auch die Komplexität der Programme, die dynamische, verschachtelte Operationsfolgen darstellen. Diese Dynamik läßt sich in Programmablaufplänen (flow-chart) darstellen. Während die Grundoperationen der Booleschen Algebra in Form elektronischer Schaltungen als Hardware (Gatter) implementiert sind, beschreiben Unterprogramme beliebige programmierbare Operationen wie mathematische Berechnungen von Quadratwurzeln, Winkelfunktionen etc.[14] Eine Sammlung von Standard-Unterprogrammen bildet eine Programmbibliothek.

1.2 Strom als fluides Trägermedium

Wie gestaltet sich die Realisierung der Zeichen in dem fluiden Medium Strom? Der Übergang von der skriptographischen zur typographischen Zeichenverwendung zeichnet sich durch eine Reduktion der bereits vorhandenen Zeichen aus. Hinzu kommt die Entwicklung neuer Prinzipien des Zeichengebrauchs: Diskretisierung der Zeichen, Sequenzierung der sprachlichen Einheiten, Entwicklung der grammatikalischen Normalform des Aussagesatzes und das Ausschreiben der Worte statt der Verwendung von Abkürzungen. Beide Technologien basieren jedoch auf demselben Erzeugungsprinzip, dem der Visualisierung mittels konventionalisierter Zeichen. Das Erzeugungsprinzip von Zeichen und Zeichensystemen mit Computern hingegen ist davon verschieden. Während im Falle der skriptographischen und typographischen Implementierung die Speicherung und die Präsentation zusammenfallen, sind beide im elektronischen Medium des Computers voneinander getrennt. Oder anders gesprochen: Das Mittel der graphisch realisierten Schriftzeichen wird nur noch für die Präsentation genutzt, nicht mehr zur Speicherung und Generierung der Zeichen auf der Maschinenebene. Das Erzeugungsprinzip von Schriftzeichen im Rechner nimmt auf das Prinzip der Verschriftung mittels diskreter Entitäten Bezug, transformiert dieses Prinzip jedoch in maschinell ausführbare Operationen.

[13] Ein Sonderfall ist ein Unterprogramm, das sich selbst aufruft (rekursives Unterprogramm).
[14] Die Hardware umfaßt als Kernstück eine Zentraleinheit (CPU central processing unit) mit Arbeitsspeicher, der bei Spannungsabfall gelöscht wird sowie Steuerwerk und Rechenwerk, die allesamt auf einem Chip integriert sind. Steuerwerk und Rechenwerk werden als Prozessor bezeichnet. Der interne Informationsaustausch erfolgt mit einem Bus-System zur Übertragung von Daten (Datenbus), zur Auswahl von Adressen (Adressbus) und zur Übermittlung von Signalen (Steuerbus). Peripheriegeräte werden über Ports angeschlossen, die an das Bus-System durch Interfaces angekoppelt sind.

Dies ist möglich, da Zeichensysteme wie die Alphabetschrift syntaktisch disjunkt und differenziert und deshalb digitalisierbar, d.h. eindeutig in einen binärkodierten, numerischen Wert überführbar sind (numerisches Prinzip).[15] Jedem gängigen Zeichen (Charakter) - Buchstaben des Alphabets, Zahlzeichen, Interpunktionszeichen und Sonderzeichen - wird mit den ASCII-Textkodes (American Standard Code for Information Interchange) ein numerischer Werte zwischen 0 und 255 zugewiesen.[16] Einem graphischen Zeichen *z* wird eindeutig ein numerischer ASCII-Kode *a für z* zugewiesen, der wiederum eindeutig als Binärzeichen *b für a* notiert ist. Die Kodierungsprozesse *a für z* und *b für a* sind programmgesteuerte Operationen. Während *a für z* konventionell festgesetzt ist, läßt sich *a für b* als Operation formalisieren. Mit den ASCII-Textkodes ist die Generierung von insgesamt 256 graphischen Zeichen möglich. Jedes ASCII-Zeichen wird durch eine 8-bit große Zahl identifiziert (2^8 = 256 Zeichen). Eine einstellige duale Zahl heißt ein bit (binary digit) und kann zwei verschiedene Zeichen darstellen: 0 oder 1. Mit 2 bit lassen sich vier Zeichen modulieren: 00, 01, 10, 11. Mit 8 bit, einem byte, lassen sich 256 Zeichen darstellen.[17]

Die computerbasierte Realisierung von Zeichen ist programmgesteuert. Programme werden mittels Programmiersprachen erstellt, die künstliche Sprachen sind. Als Maschinensprachen können sie direkt vom Computer ausgeführt werden und bestehen aus 8-bit langen 0-1-Folgen. Da sie jedoch für Menschen nicht verstehbar sind, werden Programme in Hochsprachen (C, C++, Kobol, Pascal, Prolog u.a.) geschrieben und mit Hilfe von Compilern in die Maschinensprache übersetzt. *Programme* sind Folgen von Anweisungen, also präzise artikulierte Vorgehensweisen zur Lösung spezifischer Aufgaben, die Computer verstehen können. Unter *verstehen* ist die Ausführbarkeit der in den Kodes enthaltenen Befehle zur Zeichenmanipulation gemeint. „*Programme notieren Algorithmen. Eine gutgewählte Notation muß eindeutig sein, da sie vom Rechner eindeutig in maschinelle Operationen umgesetzt werden muß*."[18] Die binärkodierte Notierung der Zeichen erfordert die Unterscheidung und Kennzeichnung von Zeichenklassen, die sich in ihrer operationalen Handhabung unterscheiden. Computer arbeiten mit verschiedenen Zeichenklassen, die in den Programmen über Variablentypen unterschieden sind: ASCII-Zeichen (char) und binary numbers: Byte-Zahlen (0 bis 255 oder –127 bis 127), Integer-Variablen (ganze Zahlen, int), Fließkomma-Variablen (Kommazahlen, float). Der Befehl *int=a;* bei-

[15] „*Die Schrift ist von Anfang an ein digitales Medium, da es sich auf einen endlichen Zeichenvorrat (Alphabet) beschränkt. Jedes endliche Alphabet läßt sich eindeutig in das Binäralphabet B={0, 1} abbilden. Dies ist die Basis der Digitalisierbarkeit aller schriftlichen Medien*." Coy, 1994, S. 71

[16] Die ASCII-Kodes zwischen 0 bis 127 sind genormt, 0 bis 31 und 127 sind Kontrollkodes, die Kodes zwischen 128 bis 255 können sich je nach gewähltem Zeichensatz unterscheiden. Beispielsweise ist der Zahl 70 der Buchstabe F zugewiesen.

[17] Mit n Bit lassen sich 2^n verschiedene Zeichen darstellen, und zwar die Zahlen von 0 bis $2^n – 1$. Ziffern sind ebenfalls mit ASCII-Kodes darstellbar, jedoch nur als konventionell vereinbarte, binärcodierte Charaktere, nicht als variabel handhabbare numerische Werte.

[18] Coy, 1994, S. 70

spielsweise in der Programmiersprache C gibt den Typ und den Namen der Variablen an und *a=1;* weist dann der Variable einen Wert zu.[19] Eine Deklaration für eine Variable kann über den Eingabekanal erfolgen, im Programm vorgegeben sein oder aktuell erzeugt werden. Der Standard-Eingabekanal ist die Tastatur, der Standard-Ausgabekanal der Bildschirm. Beide Kanäle werden automatisch durch das Betriebssystem mit dem Programm verbunden. Darüber hinaus sind eine Vielzahl weiterer Ein- und Ausgabekanäle wie Meßgeräte, Scanner oder Drucker möglich. Unabhängig vom Kanal wird jede Variable an einer definierten Stelle im Speicher abgelegt, um wieder auffindbar zu sein (Adresse). Adressen werden mit Hilfe eines einstelligen Adressoperators ermittelt, dessen Operand der Name der Variable und dessen Ergebnis die Adresse der Variable ist. Adressen lassen sich in speziellen Variablen sog. Zeigern (pointer) speichern: Mit dem Befehl *int i;* wird eine Integer-Variable deklariert, mit *int*z;* ein Zeiger auf eine Integer-Variable deklariert und schließlich mit *z=&i;* der Zeiger auf i gerichtet. Der Zeiger muß vom selben Datentyp wie die Variable sein, auf die er zeigt. Dementsprechend lauten für ein ASCII-Zeichen die Befehle: *char c; char*z; z=&c;.* Neben den Variablentypen char, int und float gibt es logische Werte, die entweder wahr oder falsch sein können und ebenfalls numerisch dargestellt sind. In der Programmiersprache C werden die logischen Werte als Integer-Variablen gespeichert: Für den logischen Wert *falsch* hat die Variable den Inhalt 0, für alle anderen von 0 verschiedenen Inhalte ist ihr logischer Wert *wahr*.[20]

1.3 Operieren, Speichern, Präsentieren

Mit der Tastatur lassen sich die Zeichen aktivieren, d.h. auf den Bildschirm schreiben. Die Zeichen, so wie wir sie sehen, werden von Zeichenfolgen (Programmen), die ebenfalls als elektrische Zustände realisiert sind, numerisch kodiert und in einem eindeutigen Binärkode notiert. Die Präsentation basiert auf graphischen Zeichen, die Inskriptionen von Schriftzeichen darstellen und als Gestalt aus Bildpunkten auf dem Monitor erscheinen. Das Erzeugungsprinzip unterscheidet sich von dem der Schriftlichkeit (phonographisches Prinzip). Die Zeichen werden weiter zerlegt und in eine binärkodierte, numerische Notation überführt (numerisches Prinzip). Es wird eine subsymbolische Ebene eingeführt, die aus einem diskreten Symbolschema und einem digitalen Schema besteht, insofern unter digital mit nume-

[19] Variablen können auch symbolische Namen sein wie *Name*, die durch eine individuelle Dateneingabe von ASCII-Zeichen mit *gets(name);* konkretisiert werden. Die Beispiele der Programmiersprache C sind dem Internet-Vorlesungsskript von Peter Kolb entnommen. [Quelle 1: Kolb, P.: C-Kurs, o. Dat.]

[20] In der Programmiersprache Pascal gibt es den Variablentyp Boolean, der die Werte TRUE oder FALSE annehmen kann. Logische Werte werden für Vergleichsoperationen (> *ist größer als*, < *ist kleiner als*, >= *ist größer oder gleich*, <= *ist kleiner oder gleich*, == *ist gleich*, != *ist ungleich*) und logische Operationen (*&& und*, *½½ oder* für zweistellige Operatoren und: *! nein* für den einstelligen Negationsoperator) verwendet. Zudem gibt es arithmetische Operationen für int- und float-Variablen.

rischen Werten darstellbar verstanden wird. Die Folge der Transformation der Zeichen in Maschinenzustände besteht in der Differenzierung eines Erzeugungsteils und eines Präsentationsteils der Zeichen. Lediglich letzterer hat mit den uns vertrauten Schriftzeichen Ähnlichkeit. Der Erzeugungsteil basiert auf der subsymbolischen Ebene unanschaulicher Zeichenzustände. Notieren Buchstaben den Klang der Sprache, so notieren Binärfolgen Schriftzeichen und geben diese programmgesteuert als visuelle Ereignisse auf den Bildschirm aus. Dabei wandelt sich der Vorgang des Notierens. Notationen werden zu programmgesteuerten Operationen, um maschinell ausführbar zu sein. Die Notationen bestehen in der Explizierung des Variablentyps, der Variablendeklaration und der Adressierung, also in den Angaben, um welche Art von Zeichenklasse es sich handelt (Variablentyp), welchen konkreten Wert die Variable besitzt (Deklaration) und wo sie gespeichert ist (Adresse). Zeichen in diesem Sinne verstanden sind nicht mehr materialfixierte Zeichenträger in Form visuell verdinglichter Entitäten als Basis von Symboloperationen, sondern sie sind Resultate von Operationen. Schrift wird zum Produkt digitaler Zeichenoperationen, d.h. der Buchstabe a wird als numerischer Wert 65 kodiert und als 1000001-Folge notiert, ein b wird als 66 kodiert und als 1000010-Folge notiert. *Elektronisch realisierte Zeichensysteme* lassen sich deshalb beschreiben als eine endliche Menge von Operationen diskreter Zustände und expliziter Regeln über den Gebrauch dieser Operationen. Da die Regeln selbst Operationen sind, stellen maschinell realisierte Zeichensysteme Operationen über Operationen dar.

1.4 Bytezahlen und ASCII-Kode

Zur Realisierung von Zahlen im Computer bieten sich zwei Möglichkeiten an: Zahlen lassen sich zum einen als Zeichen auf Basis der ASCII-Kodes wie Buchstaben oder Hilfszeichen auch darstellen, zum anderen als numerische Werte unanschaulicher Byte-Zahlen. Hier zeigt sich die Besonderheit der Entitäten, die wir mit Ziffern symbolisieren, denn Zahlen referieren auf einen geordneten Bereich von Entitäten. Und diese Ordnung läßt sich kalkülisieren und mechanisieren. Indem dieses Verhältnis umgekehrt wird und eine Zahl oder ein numerischer Wert nicht als Stellvertreter einer abstrakten Entität gilt, sondern als durch eine bestimmte Ordnungsstruktur erzeugbar, ist der Bereich der Zahlen, soweit er einer Ordnung unterliegt,[21] für die maschinelle Erzeugung und Verarbeitung prädestiniert. Zahlen lassen sich also nicht nur darstellen, sondern auch erzeugen. Grundlage dieser Ordnungsstruktur ist

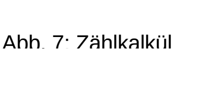

Abb. 7: Zählkalkül

[21] Dies betrifft beispielsweise nicht die komplexen Zahlen, die keiner Ordnungsstruktur kleiner, gleich, größer unterliegen.

die Generierung von Zahlen mit Hilfe eines Zählkalküls. Dazu eignen sich einfache additive Zeichensysteme wie endliche Folgen von Strichen und Regeln zur Erzeugung von Strichfolgen.[22] Es läßt sich zeigen, daß arithmetische Aussagen sich mit diesen Zählzeichen schreiben lassen: so die Gleichheit von Figuren (Konstruktionsäquivalenz), die Addition von Strichlisten *n* und *m* sowie deren Kommutativität usf. Voraussetzung ist die eindeutige Bestimmung eines ersten Zählzeichens durch eine einzige Anfangsregel, die Eindeutigkeit des Nachfolgers und damit die Verschiedenheit der unterschiedlich erzeugten Zählzeichen mit Hilfe des induktiven Prinzips.[23] Im Unterschied zu älteren additiven Zählzeichensystemen ist hier Gleichheit keine anschauliche Gestaltgleichheit von Figuren, sondern eine Folge der Herstellbarkeit nach gleicher Abfolge von Regelanwendungen. Auch wenn dies eine pragmatische Voraussetzung ist, so garantiert sie deren Wiederholbarkeit und Reproduzierbarkeit. Unabhängig von der Art der Zählzeichen müssen arithmetische Aussagen allgemeinverbindlich sein, d.h. die gleiche Abfolge von Regeln erzeugt zähläquivalente Strukturen. So ist *III + II = IIIII* äquivalent mit *3 + 2 = 5*, da *III* zähläquivalent mit *3* und *II* zähläquivalent mit *2* ist. Der Ausdruck *Zahl* gibt die vorgenommene Abstraktion wieder, die in Absehung von dem jeweiligen Zählzeichen bzw. Ziffer bezüglich deren spezifischer Eigenschaften geschieht. Dabei sind die Ziffern weder als Namen von Zahlen, noch die Zahlen als ontologische Gegebenheiten aufzufassen, sondern lediglich als Redeweisen.

Die Realisierung von Zahlen kann in unterschiedlichen Medien implementiert sein und Zählzeichen können beliebige, zähläquivalente Figuren annehmen. So läßt sich die Beschreibung *fünf* mit Hilfe von fünf Steinen oder fünf Äpfeln veranschaulichen, mit den Zahlzeichen 5, IIIII oder V darstellen oder als 00000101 Binärfolge notieren. Im Falle der rechnergestützten Realisierung wird auf zwei verschiedene Möglichkeiten zur Darstellung von Zahlen zurückgegriffen: auf *binary numbers* (byte -, integer -, floating-point numbers) und auf *ASCII text numbers*. Mit den bereits dargestellten ASCII-Kodes lassen sich die Zahlen 0 bis 9 sowie Interpunktionszeichen als Charaktere darstellen, die für uns als Zeichen lesbar sind.[24] Binäre Zahlen hingegen sind nur Computern zugänglich. *Byte-Zahlen* sind in einem byte gespeichert und decken das Zahlenspektrum von 0 bis 255 (unsigned byte) oder von –128 bis 127

[22] „... *mit der Anfangsregel „Þ ½" und der Fortsetzungsregel „n Þ n½", in der „n" als schematischer Buchstabe gemäß N herstellbare Figuren vertritt. Jede nach den Regeln von N herstellbare Figur wollen wir ein „Zählzeichen" oder eine „Ziffer" nennen, also abweichend von dem normalen Sprachgebrauch* ..." Thiel, 1995, S. 114

[23] Prinzip der vollständigen Induktion oder Schluß von n auf n + 1 zur Erzeugung natürlicher Zahlen. Ähnlich geht David Hilbert vor, allerdings ohne bereits auf das induktive Prinzip zu referieren, indem er lediglich einen axiomenfreien Auf- und Abbau der Zahlzeichen vorschlägt: „*Wir beginnen also mit folgenden Erklärungen der Zahlen. Ein Zeichen 1 ist eine Zahl. Ein Zeichen, das mit 1 beginnt und mit 1 endigt, so daß dazwischen auf 1 immer + und auf + immer 1 folgt, ist ebenfalls eine Zahl, z.B. die Zeichen 1 + 1, 1 + 1 + 1. Diese Zahlzeichen, die Zahlen sind und die Zahlen vollständig ausmachen, sind selbst Gegenstand unserer Betrachtung, haben aber sonst keinerlei Bedeutung. Außer diesen Zeichen wenden wir noch andere Zeichen an, die etwas bedeuten und zur Mitteilung dienen, z.B. das Zeichen 2 zur Abkürzung für das Zahlzeichen 1+ 1 ... ferner wenden wir die Zeichen =, > an, die zur Mitteilung von Behauptungen dienen*." Hilbert, 1965b, S. 163

[24] So läßt sich die Zahl 3.145 als 5-byte ASCII-text-string *51 46 49 52 53* darstellen. Der Vorteil von ASCII text numbers ist, daß sie für jede Plattform portabel und als Zeichen für uns lesbar sind. D.h. jede binäre Zahl, die lesbar sein soll, muß in eine ASCII-Zahl konvertiert werden.

(signed byte) ab.[25] Zur Notierung von höherstelligen Zahlen werden *integer numbers* verwendet, die aus zwei bytes (short integer), vier bytes (long integer) oder acht bytes (64-bit-integer) bestehen. Das Zahlenspektrum läßt sich für Integerzahlen erweitern, indem Zahlen als *fixed-point numbers* repräsentiert werden. So kann beispielsweise die Zahl π = 3,14159... mit 100.000 multipliziert als Integerzahl 314.159 gespeichert werden, wobei der Multiplikationsfaktor im Programm expliziert sein muß.[26] Für viele Berechnungen genügt dieses Zahlenspektrum jedoch meist nicht. Großrechner sind in der Regel mit floating-point Prozessoren ausgestattet, welche die schnelle Verarbeitung von *floating-point numbers* ermöglichen. Der IEEE-Standard 754 regelt die Arbeitsweise mit diesen Zahlendarstellungen (single-precision floating format). Für dieses 32-bit Format ist das erste bit zur Kennzeichnung einer negativen Zahl (0) bzw. positiven Zahl (1) reserviert, die nächsten acht bits zur Darstellung des Exponenten (1 – 254) und die restlichen 23 bits für die Fraktion (1,0 – 9,999).[27] Das Zahlenspektrum der floating-point numbers reicht von 1,175 x 10^{-38} bis 3,403 x 10^{38}. Aufgrund der 23 bit Kapazität zur Speicherung der Fraktion lassen sich Zahlen bis auf sieben Stellen genau in Dezimalnotation darstellen. Das bedeutet, daß jede weitere Stelle gerundet wird, so daß die Zahlen 3,14159265 und 3,14159264 als selbe Zahl gespeichert werden. Der Rundungsfehler kann mitunter sehr unangenehm werden.[28] Mit Hilfe weiterer floating-point Formate, die 64 bzw. 80 bit umfassen (double precision floating-point format bzw. extended format) lassen sich Zahlen auf 15 bzw. 18 Dezimalstellen genau wiedergeben.[29]

Computerrealisierte Zahlen sind im Grunde natürliche Zahlen. Erst mit der Kodierung spezifischer Strukturen lassen sich weitere Zahlensysteme darstellen.[30] Dabei ist zu beachten, daß Computer elektrische Zustände verarbeiten, während für uns das Rechnen nur in symbolischer Notation (Charak-

---

[25] Für unsigned bytes werden alle acht bit zur Speicherung der Zahl verwendet (00000000 – 11111111), für signed bytes wird das erste bit zur Kennzeichnung einer negativen Zahl (0) bzw. positiven Zahl (1) und die restlichen sieben bits zur Speicherung der Zahl verwendet (10000000 – 01111111). Ob eine Zahl als signed oder unsiged byte gespeichert ist, muß im Programm festgelegt werden, denn anhand der Speicherung läßt sich das nicht erkennen.

[26] „*... you must know how the numbers are stored, their multipliers, and the offsets. The data itself does not contain this information. In other words, integer and fixed-point datafiles are not self-describing*.“ Fortner, B.: The Data Handbook, 1995, S. 39

[27] Dazu wird beispielsweise die Zahl 1,1277089 x 10^{-90} in die Fraktion 1,1277089 und den Exponenten –90 zerlegt und wie folgt als floating-point Zahl (float) gespeichert:

s	Exponent	Fraktion
0	-90	11277089
0	-1011010	10010000001011000110001

[28] „*This proplem can be espacially acute for simulations that do calculations thousands or millions of times. For example, after a million calculations, your roundoff error could be larger than 0.1*.“ Fortner, 1995, S. 51

[29] Diese Zahlenspektren werden für die Simulation galaktischer oder atomarer Szenarien benötigt. Über die Genauigkeit hinaus ist die Geschwindigkeit entscheidend. MFLOPS geben an, zu wieviel Millionen floating-point Operationen pro Sekunde ein Rechner in der Lage ist. Und obwohl die Portabilität des Floating-point Formates aufgrund der Verschiedenheit der Großrechnerformate gering ist, liegt der Vorteil darin, daß floating-point Daten im Gegensatz zu Integerzahlen selbstbeschreibend sind, d.h. alle Informationen über die Zahl sind in der Zahl gespeichert.

[30] Beispielsweise signed bits zur Darstellung negativer (0) oder positiver (1) Zahlen.

tere) zugänglich ist. Die Reduktion numerischer Werte auf binäre Zustände eröffnet die Möglichkeit, diese nicht nur als Charaktere darzustellen, sondern auch als Farbwerte oder Klänge. Während es die Verschriftung erlaubt, die Darstellung von Zahlen und das Rechnen in einem Medium auszuführen, zerfällt dieser Prozeß mit der Verwendung von Computern in zwei Teile: zum einen in die Erzeugung von Zahlen und die Durchführung von Berechnungen auf Maschinenebene, zum anderen in die visuelle Präsentation der Ergebnisse auf dem Bildschirm oder dem Drucker als ASCII text number oder als Farbwerte. Die Einführung der subsymbolischen Ebene erlaubt die frei wählbare Präsentation der Zeichen als Ziffern, Farbwerte oder sogar Klang, ohne die eindeutige Notation als binary number zu verlieren. Damit wird die Fixierung auf eine spezifische Form des Zeichenträgers überwunden, d.h. die an sich für den Menschen unleserlichen, binärkodierten Zeichen können in unterschiedlichen Inskriptionen präsentiert werden und diese Inskriptionen lassen sich programmgesteuert ineinander überführen. Da sie aufgrund derselben arithmetischen Regeln gebildet sind, sind sie zähläquivalent, ohne daß dies aus ihrer symbolischen Form erschließbar wäre. Lediglich als Ziffern sind die numerischen Werte für uns leserlich, als Farbwerte sind sie ins Ikonische transformiert anschaulich geworden. Dies ist eine maßgebliche Konsequenz der computerbasierten Realisierung von Zahlen. Sowohl computerrealisierte Zeichen als auch Zahlen weisen bezüglich der frei wählbaren Präsentation neue semiotische Eigenschaften auf. Vor allem für die Zahlenausdrücke bedeutet dies, daß formal-operativ erzeugte Ordnungsstrukturen in einer neuen Art sichtbar und analysierbar werden.

1.5 Subsymbolische Ebene digitaler Zeichen

In einem ersten Resümee läßt sich feststellen, daß die Rede von Zeichen, Zeichensystemen, Zeichenoperationen und Zeichenmanipulationen mehrdeutig ist. Unser Verständnis von schriftbasierten Zeichen, also Texturen, ist an das visuell Wahrnehmbare und Lesbare gekoppelt. Es ist nicht ganz zutreffend, wenn wir von einem strengen Begriff der Formalisierung und Kalkülisierung ausgehen, der auf jegliche symbolische Beschreibung verzichtet. Dies wird spätestens mit der Transformation der Zeichen in ein maschinelles Medium wie den Computer deutlich. Um mit Texturen tatsächlich maschinell zu operieren, werden diese zu binären Zuständen, die für uns mangels jeglicher Anschauung nicht mehr zugänglich sind.[31] Es bedarf der programmgesteuerten Übersetzung und Interpretation der Zustände in für uns wahrnehmbare Zeichen, wobei jegliche symbolische Beschreibung oder Darstel-

[31] Binary number representations: „*The numbers are coded in a very efficient way that is not `human readable´, meaning that printing the file will produce garbage. Binary data is meant to be read only by computer programs*." Fortner, 1995, S. 16

lung expliziert und operationalisiert werden muß.[32] Wenn wir leichthin ein Zeichen *a* als Buchstaben oder gar als den Buchstaben a identifizieren, ein Zeichen *5* als Zahl oder gar als Summe von 3 + 2, bedarf es zur maschinellen Realisierung dieser Zeichen unterschiedliche Kodierungs- und Kennzeichnungsvorgänge: *a* (char), *5* (int).

Es zeigt sich, daß die maschinell realisierten Zeichen nicht dem typographisch orientierten Zeichenbegriff und dessen Prinzip der visuellen Verdinglichung entsprechen. Was aber ist ein maschinell (computerbasiertes) realisiertes Zeichen? In der erfolgten Beschreibung elektronisch realisierter Zeichensysteme wurde der Begriff *diskrete Zeichen* durch *Operationen diskreter Zustände* ersetzt. Diskrete Zeichen sind aufgrund ihrer geschlossenen Gestalt wohlunterschiedene, syntaktisch disjunkte und differenzierte, graphische Konfigurationen, die eindeutig als Buchstaben, Zahlzeichen, Operationszeichen oder Hilfszeichen wahrgenommen werden können, wenn sie wohlgeformt sind. Im Falle formal-operativer Zeichensysteme ist es unerheblich, die Zeichen als spezifische Buchstaben wahrzunehmen. Es ist jedoch von entscheidender Bedeutung, die Zeichen (Buchstaben, Zahlzeichen, Operationszeichen, Hilfszeichen) als Zeichen bestimmter Zeichenklassen mit eindeutig durch Regeln und Definitionen zugeordneten Funktionen zu interpretieren. Dies macht deutlich, daß auch in schriftbasierten Kalkülsystemen die Zeichen nicht beliebige Entitäten sind, sondern daß sie als syntaktisch disjunkte und differenzierte Inskriptionen intrasymbolisch wohl definiert und unterschieden sowie Zeichenklassen aufgrund ihrer Gestalt zugeordnet sein müssen.[33]

Bei genauer Betrachtung zeigt sich, daß nicht die uns vertrauten graphischen Zeichen für die computerbasierte Realisierung transformiert werden, sondern ihre syntaktischen Eigenschaften. Die syntaktische Disjunktheit und Differenziertheit der Zeichen wird zur eindeutig kodierten Diskretheit der elektrischen Zustände. Die in den unterschiedlichen graphischen Gestalten der Zeichen konventionell kodierten Informationen werden als numerische Werte notiert (ASCII-Kode). D.h. nicht die Schrift als Technologie zur Sichtbarmachung von Zeichen, sondern das Prinzip der Schrift, das ein syntaktisch disjunktes und differenziertes Symbolschema konstituiert, wird in ein computables Symbolsystem

---

[32] Übersetzungen und Interpretationen können jedoch fehlerhaft sein, d.h. wir können nicht immer sicher sein, ob die auf dem Bildschirm präsentierten graphischen Zeichen tatsächlich den Eingaben entsprechen. Nun mag man einwenden, daß schlecht reproduzierte Druckbuchstaben oder unleserliche Handschriften ebenfalls fehlerhaft sein können. Der Unterschied besteht jedoch darin, daß diese Form der Fehlerhaftigkeit leicht erkannt werden kann, da sie die Wohlgeformtheit der Zeichen verletzt. Eine falsche Interpretation des Kodes seitens des Programms mag ein falsches, aber wohlgeformtes Zeichen erzeugen, die Fehlerhaftigkeit besteht in einer falschen Relation zwischen subsymbolischer und intrasymbolischer Ebene. Dies ist vor allem für numerische Simulationen von Bedeutung, da hier das Kriterium der Wohlgeformtheit kein Unterschiedungskriterium mehr zwischen fehlerhafter und fehlerfreier Darstellung ist.

[33] Natürlich können Buchstaben oder Zahlen Operations- oder Hilfszeichen markieren, doch wenn die Interpretation einmal festgelegt ist, sind die Zeichen nicht mehr beliebig austauschbar. Unabhängig welche Zeichen wie interpretiert werden, die Interpretation gilt dann für die gesamte Zeichenklasse, wie die der Buchstaben oder Operationszeichen. D.h. die Fähigkeit zur Unterscheidung von Zeichenklassen aufgrund ihrer Gestalt wird vorausgesetzt.

überführt. Dieses Symbolsystem besteht aus einem diskreten Symbolschema und einem digitalen Schema. Die so konstituierten Zeichen sollen *digitale Zeichen* genannt werden. Digitale Zeichen führen eine zusätzliche Ebene ein, die als subsymbolische Ebene bezeichnet wird. Während die Einführung der intrasymbolischen Ebene durch die Formalisierung der Zeichenverwendung die Handhabung der Zeichen operationalisiert und diese Operationen mit entsprechenden Zeichen symbolisiert, operationalisiert die subsymbolische Ebene die Zeichen als Maschinenzustand selbst. Dies geschieht zum Zwecke der maschinellen Realisierung des Zeichenumgangs. Oder anders gesprochen: Die Zeichen sind nicht mehr nur Entitäten der Symboloperationen, sie werden im elektronischen Medium selbst operationalisiert. Die wahrnehmbare Präsentation ist lediglich eine Referenz an unsere Anschauung.

Der Zusammenhang zwischen Formalisierung und Mechanisierung ergibt sich demnach aus der Strukturierung der Zeichenverwendung. Dabei werden die Zeichen weiter zerlegt und besitzen eine digitale, unanschauliche als auch eine anschauliche Extension. Der Vorteil der weiteren Zerlegung der Zeichen ist ihre Operationalisierbarkeit auf der subsymbolischen Ebene. Erst als digitale Zeichen besitzen sie diejenigen Eigenschaften, um maschinell verarbeitet werden zu können. Zu diesen Eigenschaften gehören ihr diskretes Symbolschema und ihr digitales Schema sowie ihre maschinelle Erzeugbarkeit. Maschinell ausführbare Operationen ergeben sich aus der Konfiguration elektrischer Zustandsänderungen, die mit den Schaltfunktionen beschreibbar sind. Schaltfunktionen sind semiotische Maschinen zur Erzeugung definierter Zustände, die als Werte interpretiert und gegebenenfalls mit schriftbasierten Zeichen präsentiert werden können. Die dafür notwendigen Schaltungen sind in der Hardware implementiert. Die Trennung zwischen Erzeugung und Speicherung von Werten und deren Präsentation erlaubt für letztere die mediale Freiheit. D.h. Werte können als schriftbasierte Zeichen präsentiert werden (ASCII-Zeichen), als Farbwerte oder Töne. Die Mechanisierung atomisiert die Zeichen weiter[34] und führt einen neuen Umgang im Vergleich zum schriftbasierten Zeichengebrauch ein, welcher die Erzeugung[35] und Speicherung der Entitäten, die von deren Präsentation abgekoppelt ist, berücksichtigt.[36] Neben der Entkopplung von Semantik und Syntax durch die Formalisierung und die daraus resultierende Reglementierung und Schematisierung der Zeichenverwendung zum operati-

[34] Ist für den klassischen Zeichenbegriff das bedeutungsvolle Wort die kleinste Einheit, so werden diese `Zeichen` für formalisierte und kalkülisierte Zeichensysteme in ihre Teile (Grapheme) atomisiert. Die Aufmerksamkeit konzentriert sich auf das bloße Zeichen und seine syntaktischen Eigenschaften. Die Mechanisierung führt zu einer weiteren Normierung des Zeichenbegriffs.
[35] In typographisch realisierten Zeichensystemen gilt die Normierung der Zeichenerzeugung - der Schreib- oder Druckvorgang - mit der Abkoppelung von extrasymbolischen Bedeutungen als vernachlässigbar. Jede beliebige graphische Konfiguration kann, wenn sie hinreichend syntaktisch disjunkt und differenziert ist, verwendet werden und der Zeichenvorrat wird als gegeben für das Zeichensystem vorausgesetzt. Der Erzeugungsprozeß hat keinerlei Einfluß auf die Zeichen und das Zeichensystem. Eine Ausnahme bilden Zeichen für Werte, die anzeigen, woraus sie zusammengesetzt, also wie sie erzeugbar sind.
[36] Entgegen dem Argument, das computerrealisierte Zeichen bestünde nur in seiner symbolischen Präsentation und diese Präsentation referiere auf elektrische Zustände, soll angeführt werden, daß die eigentlichen Symbolmanipulationen der Maschinen nicht auf der Präsentationsebene stattfinden. Ein solcher Zeichenbegriff würde die Eigenschaften der Mechanisierung von Zeichensystemen nicht erfassen und nicht klären, was es heißt, den Umgang mit den Zeichen zu mechanisieren.

ven Gebrauch beliebiger graphischer Entitäten erfordert die Mechanisierung die Entkopplung der Zeichen von ihrer graphischen Form - und damit letzte Reste intrasymbolischer Bedeutungen - und die Operationalisierung der in der graphischen Form implementierten Informationen als geordnete Konfigurationen diskreter Zustände, die numerische Werte repräsentieren.

2. Form der Simulation

Mit dem Computer ist ein Instrument gegeben, das Zeichen gemäß Operationsvorschriften umformt und erzeugt. Aufgrund der dargestellten Besonderheit handelt es sich dabei um Zeichen, die dem Symbolsystem des Computers entsprechend als digitale Zeichen numerisch kodiert sind. Die naheliegendste Verwendungsweise des Computers ist daher der Gebrauch als Rechenmaschine und die Entwicklung mathematischer Methoden, die jene arithmetischen Vorteile des Computers nutzen. Die numerische Simulation ist eine solche Methode, welche die Arbeitsweise und Leistungsfähigkeit der Rechner in einer neuen Strategie zur Lösung komplexer Gleichungen umsetzt. Das Besondere der numerischen Simulation im Unterschied zu anderen Anwendungen dabei ist, daß die Simulation die Eigenheiten der digitalisierten Zeichen nutzt. Oder anders gesprochen: Die Möglichkeiten digitalisierter Zeichen zeigen sich nirgends deutlicher als bei der Methode der numerischen Simulation und ihrer Visualisierung.

2.1 Organisationsprinzip

Nachdem untersucht wurde, was Zeichensysteme sind, wie sich deren beschreibende Funktion von der berechnenden unterscheidet und wie sich die Transformation schriftbasierter Zeichen in computerbasierte vollzieht, gilt nun das Interesse der Frage, welche neuen Darstellungsmöglichkeiten sich mit digitalen Zeichen eröffnen. Die numerische Simulation wird dazu als prototypisches Anwendungsbeispiel analysiert. Im Unterschied zur Schrift basiert die Mechanisierung der Zeichen primär nicht auf den Eigenschaften visueller Entitäten, sondern auf den (unanschaulichen) operativen Eigenschaften der Zeichen. Die notwendige Interpretation dieser Zeichenoperationen als Schriftzeichen, Farbwerte oder Klänge entbindet computerbasierte Zeichen von einer fixierten, sinnlich wahrnehmbaren Präsentation. Dies ist möglich, weil die Erzeugung der Zeichen selbst einer Formalisierung unterworfen wird, welche die syntaktischen Eigenschaften der Schriftzeichen erhält (Disjunktheit und Differenziertheit), ihre gestaltbasierte Unterscheidbarkeit numerisch kodiert und die Präsentation von der Erzeugung, Verarbeitung und Speicherung der Zeichen abkoppelt. Die unanschauliche Verarbeitung elektrischer Zustände erreicht mit den aktuellen Rechnerleistungen eine enorme Geschwindigkeit: Es werden rund 100 Milliarden Operationen in der Sekunde durchgeführt, d.h. es können wesentlich mehr computerbasierte Zeichen in der selben Zeit generiert und verarbeitet werden als es Menschen möglich ist, schriftbasierte Zeichen zu erzeugen und zu erkennen. Vor allem für die numerische Simulation ist dies von Bedeutung.

Welche Art von Symbolsystem konstituiert die numerische Simulation? Dazu wird die Form, die Entfaltung der Form und die Präsentation der Form numerischer Simulationen untersucht. Mit *Form* ist zunächst nicht die äußere Form oder Gestalt gemeint, sondern die spezifische Weise des Umgangs mit digitalen Zeichen. Sie stellt ein bestimmtes Organisationsprinzip dar ähnlich jenen der Schrift, des Kalküls oder der Liste für typographische Zeichen. Ein Text beispielsweise basiert in diesem Sinne auf einer spezifischen Strukturierung (Wortbildung und –reihung) von Materialien (Buchstaben), und die Organisationsprinzipien der Schrift erlauben die Möglichkeit der Texterstellung. Sie bestehen nicht nur in der Anweisung, die Buchstaben von links nach rechts anzuordnen oder die Zeilen von oben nach unten fortzuschreiben, sie beinhalten auch die Implementierung des Materials in dessen Strukturierung auf einem Trägermaterial in der Weise, daß die Lesbarkeit gewährleistet ist (Zeichengestalt-Hintergrund-Differenzierung). Dies und mehr trägt dazu bei, mit der typographischen Form Informationen als Text zu realisieren, wobei hier unter Information lediglich die syntaktische Bereitstellung differenzierter Zeichenfolgen in einer normierten Form gemeint ist, d.h. die Informationen sind in der gestaltbasierten Differenzierung der Zeichen verschlüsselt. Die Bedeutung generiert sich dann durch die Interpretation der Zeichen des geschriebenen Textes.[37] Im Unterschied zur typographischen Form, die bereits fünfhundert Jahre alt ist und auf der noch älteren skriptographischen basiert, wird die Form der numerischen Simulation zur Informationsdarstellung aktuell entwickelt. Simulationen als spezifische Anwendungen mechanisierter, zeichenbasierter Operationen erschließen einen semiotischen Bereich, der eine neue Form der Syntax und der Semantik mit sich bringt, die von der Dynamik der Zeichenoperationen entfaltet und ikonisch dargestellt wird. Damit sind neue Einsichten möglich, wobei zu klären sein wird, was sich da zeigt.

[37] Für einen herkömmlichen Text mag dies unzutreffend klingen, da bereits die Erstellung des Textes das Wissen um die Bedeutung voraussetzt. Für die hier betrachteten formal-operativen Zeichensysteme ist die Bedeutung ohne Belang, d.h. Erzeugung und Interpretation sind getrennte Vorgänge.

Die Form der numerischen Simulation ist wesentlich komplexer als die typographische. Die Mechanisierung der Zeichenverarbeitung setzt ein maschinelles Medium voraus, das nicht nur Spuren hinterläßt, sondern den Zeichenumgang mitstrukturiert.[38] Beim Vorgang des Schreibens wird mit passiven Trägermaterialien (Papier, Tinte) die Schrift von einem Autor ausgeführt. Die Trägermaterialien schaffen die Voraussetzung für die Ausführung der Implementierung von Inskriptionen. Die Mechanisierung der Zeichenverarbeitung in einem fluiden Medium (Strom) wandelt die Passivität des Vorhandenseins des Trägermaterials in funktionalisierte Prozesse, die von der Hardware für die Implementierung der Software bereitgestellt werden. Dabei wird der gesamte Vorgang der Zeichenerzeugung umstrukturiert und die eigentliche Handlung der Implementierung wird an das maschinelle Medium delegiert. Dazu bedarf es einerseits der Algorithmisierung der Implementierungsvorgänge im Rahmen von Computerprogrammen, andererseits der Normierung der Inputeingabe in Form taktiler Aktivierungen von Zeichenprozessen.[39] Der Vorgang des *Schreibens* erhält eine zweifache Bedeutung. Zum einen, indem die herkömmliche Form als Implementierung von Inskriptionen mit Hilfe der Tastatur simuliert wird, so als schreibe man direkt auf den Bildschirm, zum anderen als Programmierung maschinell ausführbarer Instruktionen. Durch die Delegation der Implementierung von Inskriptionen an den Computer und die Reduktion des Schreibvorgangs als Input- oder Dateneingabe wird es möglich, die maschinelle Inkorporation von Texten wie von Objekten vorzunehmen, insofern letztere digitalisierbar sind.[40] Die Mechanisierung des Trägermaterials funktionalisiert das fluide Medium des Elektrischen entsprechend der logischen Struktur der Schaltungen (Gatter), welche die Hardware konstituiert, und ermöglicht die binäralgebraische Verarbeitung der elektrischen Zustände. Die Kombination der logischen Struktur der Hardware und der operativen Struktur der Software mit der Datenstruktur der Inputeingabe und -ausgabe bilden die Bedingungen des Organisationsprinzips des Computers als semiotische Maschine. Die Zeichenverwendung in Form numerischer Simulationen ergibt sich darüber hinaus aus der mathematischen Struktur der zu simulierenden Gleichungen.

---

[38] Im Unterschied zur Auffassung - Medien wären nur sinntransportierend und nicht sinnstiftend – kann die vorliegende Untersuchung als Versuch gewertet werden, den Einfluß der Medien darzulegen. „*Das Medium ist nicht einfach die Botschaft, vielmehr bewahrt sich an der Botschaft die Spur des Mediums*." Krämer, S.: Das Medium als Spur und Apparat, 1998a, S. 81. Es zeigt sich, daß die elektronischen Medien mehr als nur vage Spuren hinterlassen, sondern die Bedingungen der Zeichenoperationen in Form von Normierungen stellen. Allgemein läßt sich zeigen, daß die Medien die Form der Symbolsysteme mitkonstituieren, wie es anhand der digitalen Zeichen deutlich wird.

[39] Das klassische Eingabegerät ist die Tastatur, doch auch die akustische Inputeingabe via Mikrophon ist mittlerweile möglich. Dabei wird die Inputeingabe zur Dateneingabe und läßt sich automatisieren, d.h. die Dateneingabe muß nicht ausschließlich von einem menschlichen Autor vorgenommen werden, sondern kann aus der Tätigkeit von Maschinen wie Scanner, digitale Kameras, Meßgeräte etc. resultieren.

[40] Damit sind nicht nur Bilder von Objekten gemeint, sondern Objekte selbst, insofern sich deren Oberflächen mit Hilfe von 3D-Scannern digitalisiert *einschreiben* lassen, oder die durch Meßdaten charakterisiert werden.

2.2 Überschreiben als neue Technologie

Die logische Struktur der Hardware resultiert aus der Schaltalgebra, die es erlaubt, die Binärarithmetik mit den logischen Operationen UND, ODER, NICHT auszudrücken. Die physikalische Implementierung der Schaltalgebra (Gatter) entspricht der Modulation des Aussagenkalküls mit diskreten elektrischen Zuständen als Interpretationen von wahr und falsch bzw. 1 und 0 (Stromfluß/kein Stromfluß). Die Regeln der Schaltalgebra bilden eine Boolesche Algebra und es läßt sich mit den logischen Werten gemäß diesen Regeln rechnen. Aus diesen basalen Schaltungen werden in einer Art hierarchischen Baukastenprinzips zahlreiche weitere Schaltungen zur Speicherung, Multiplikation, Division und mehr aufgebaut, und insbesondere die Verschaltung zu Flipflops ermöglicht die Speicherung von Werten. Diese Strukturierung des Trägermediums erfordert die Formalisierung der Zeichen als programmgesteuerte Zeichenoperationen. Ein computerrealisiertes Zeichen generiert sich aus den numerisch gespeicherten Informationen über seinen Typ (char, int, float), seinen Ort (Adressierung) und seinen Wert im Kontext eines Programms.

Die Zeichen zu verarbeiten bedeutet, die Werte entsprechend einer Umformungsvorschrift direkt zu verändern und dabei den alten Zustand zu *überschreiben*. Diese Verarbeitungsweise unterscheidet sich von einem schriftbasierten Formalismus, der aus einem schriftlich notierten Umformungsschritt den nächsten ableitet, niederschreibt und so einen linearen Ablauf der Verarbeitungsschritte erzeugt und gleichzeitig speichert. Die Arbeitsweise der Computer hingegen erfordert einen anderen Umgang. Auf Basis eindeutiger Instruktionen werden die Zustände der Zeichen umgeformt und überschrieben, wobei die Instruktionen Verrechnungen auf binärarithmetischer Basis darstellen. Dies setzt voraus, daß Aufgabenstellungen in Form von Problemlösungen formuliert und daß die einzelnen Schritte zur Problemlösung in ihre elementarsten Teile zerlegt werden, so daß sie als Umformungsvorschrift regelbasiert abarbeitbar sind und schließlich als Lösung die neuen Zustände ausgeben.[41] Die Definition des Algorithmischen ist Ausdruck dieses operativen Vorgehens in einem maschinellen Medium.[42] Zumeist wird der Fokus auf de analytische Verfahrensweise der Algorithmen gerichtet, die im Sinne der Konzeption Alan Turings so elementar ist, daß die Begriffe Algorithmus, allgemeines Problemlösungsverfahren und Berechenbarkeit synonym werden. Von Bedeutung ist jedoch, daß Algorithmen operative Instrumente sind, welche die Lösung einer Aufgabe nicht nur darstellen, sondern im Rah-

[41] Dabei muß es sich nicht nur um Umformungen handeln, sondern es können auch Löschungen oder Hinzufügungen stattfinden. Das charakteristische der Arbeitsweise der Computer liegt jedoch in den Umformungen und Überschreibungen von Daten (Zeichenmanipulationen). Entsprechend der Konzeption Alan M. Turings wäre es die Vorschrift des Löschens und Beschriftens eines Feldes und damit das Überschreiben von Zeichen.

[42] "*An Algorithm is a precise and unambigous specification of a sequence of steps that can be carried out mechanically*." Aho, A.V./Ullmann, J.D.: Fundation of Computer Science, 1995, S. 5

men eines Programms zugleich ausführbar machen. Sie zeichnen sich durch ihre darstellende und exekutive Funktion aus, wobei die exekutive Funktion im Medium des Computers zur Entfaltung kommt und sich in der Notwendigkeit zeigt, daß ein Algorithmus abbrechend sein muß, will man ihn nicht zwangsweise stoppen oder den Rechner bis in alle Ewigkeit arbeiten lassen. D.h. die Endlichkeit des Textes des Algorithmus garantiert nicht seine Ausführbarkeit in endlicher Zeit. Dabei gilt das pragmatische Kriterium der Effizienz als Maß der Ausführbarkeit bezüglich endlicher Ressourcen. So ist eine Funktion dann berechenbar, wenn ein abbrechender Algorithmus existiert, der bei vorgegebenen Argumenten den Funktionswert liefert.[43]

Der Kode eines Computerprogramms besteht aus Kommandozeilen, die in linearer Abfolge einen vielschichtigen und verschachtelten Operationsablauf in einen endlichen Text fassen. Die Ausführung der Instruktionen hingegen folgt in der Regel nicht dem linearen Ablauf des Textes von Kommandozeile zu Kommandozeile, sondern ist durch Subroutinen, Schleifen und Sprünge zwischen Programmteilen strukturiert. Neben dieser logischen und algorithmischen Struktur basieren die numerischen Simulationen auf mathematischen Strukturen, die sich aus partiellen Differentialgleichungen herleiten. Mit diesen läßt sich die Dynamik zeitbasierter Systeme darstellen, indem eine Beziehung zwischen einer Funktion und ihren Ableitungen geschaffen wird. Die zu bearbeitenden Informationen sind in den Gleichungen symbolisiert, und die Aufgabenstellung besteht in der Berechnung der Approximation der Lösungen dieser Gleichungen für ausgewählte Eingabewerte. Um Berechnungen auf einem Computer auszuführen, bedarf es der Anpassung der zu berechnenden Gleichung an das Organisationsprinzip der Rechner. Alle symbolischen Kodierungen müssen in maschinell ausführbare, endliche Operationen übersetzt werden, d.h. die Gleichungen sind zu algorithmisieren und zu diskretisieren. Schließlich wird mit einem Programm der gesamte Ablauf der Simulation festgelegt. Die Triade Modell – Algorithmus – Programm bildet die Grundlage der numerischen Simulation.[44]

[43] Vrgl. Hermes, 1978, S. 1ff
[44] Neunzert, 1995, S. 50ff. *„Der bekannte russische Mathematiker A.A. Samarskii ... spricht dabei vom „Computerexperiment", da es dieselbe Aufgabe hat wie ein normales Experiment, nämlich die Richtigkeit des Modells zu prüfen. Dieses Computerexperiment mit seinen drei Stufen (der „Triade", wie Samarskii das nennt) Modell – Algorithmus – Programm ist nach Samarskii nicht nur eine neue Wissenschaftsrichtung – „es ist eine neue wissenschaftliche Methode, die sowohl dem Denkstil eines modernen Wissenschaftlers als auch den Kreis der Probleme bestimmt, welche sich der Forscher zu stellen vermag." Es ist „eine neue Art die Natur zu befragen" und es ist darüber hinaus auch eine neue Methode des technischen Designs*." Neunzert, H.: Von Modellen und wie man sie nutzt, 1990, S. 13

2.3 Zeichenmaterial und Umsetzung

Das Material der numerischen Simulation sind Zahlen. Die Dateneingabe wie die Ausgabe besteht in numerischen Werten, die als Folge von 0 und 1 bzw. elektrischen Zuständen auf der Maschinenebene repräsentiert sind. Die Übersetzung der 0-1-Folgen in Pixel-Darstellungen zur Präsentation auf dem Koordinatensystem des Monitors kann in Gestalt von ASCII-Zeichen (Ziffern) oder Farbwerten erfolgen. Numerische Werte unterscheiden sich von ASCII-Zeichen oder Farbwerten durch ihre regelbasierte Erzeugungsweise, die es erlaubt, Werte gemäß einem Mechanismus operativ zu generieren.[45] Als digitale Zeichen sind sie diskret und differenziert sowie eindeutig. Allerdings begrenzt sich die Eindeutigkeit aufgrund der Endlichkeit der Zahlendarstellung im Computer auf 15 oder mehr Dezimalstellen je nach Rechnertyp. Ein numerischer Wert kann als Input vorgegeben sein oder durch Berechnung erzeugt werden. Im Gegensatz dazu sind ASCII-Zeichen wie auch Farbwerte jeweils im Konkreten zu definierende visuelle Entitäten, für die es nur eine Vorschrift, aber keinen Mechanismus der Erzeugung gibt. Die Vorschrift besteht in der programmgesteuerten Zuordnung zwischen einem numerischen Wert und einem ASCII-Zeichen oder Farbwert. ASCII-Zeichen sind diskret und syntaktisch differenziert. Farbwerte sind zwar als Werte diskret sowie syntaktisch differenziert, in ihrer Präsentation jedoch syntaktisch dicht. Computer als Maschinen zur Verarbeitung von Zeichenoperationen auf binärarithmetischer Basis sind daher bestens geeignet, mit numerischen und binärlogischen Werten dynamisch zu operieren und diese mit Programmen in beliebigen Zeichenzuordnungen zu präsentieren, jedoch nicht umgekehrt. Die Simulationen diskretisierter Gleichungen operieren mit numerischen Werten und präsentieren diese auf Basis ASCII-kodierter oder farbwertbasierter Zuordnungen. Die sichtbaren Ergebnisse sind dann endlos lange Zahlenkolonnen oder ikonische Visualisierungen.

Die Simulation erlaubt die semiotische Realisierung dessen, was in der Neuzeit als grundlegender Wandel in der Mathematik stattgefunden hat und seither symbolisch in den formalen Strukturen kodiert ist: "*Die wichtigste herausragende Bedeutung der neuen Mathematik war dagegen die umfassende Einbeziehung der Bewegung. Das führte zu einer Umbildung der Objekte aller grundlegenden mathematischen Objekte.*"[46] Der Computer stellt das maschinelle Medium, die Simulation, die Form zur Realisierung dynamischer Prozesse und deren Sichtbarmachung dar, insofern sich die Dynamik im Werteverlauf einer simulierten Struktur zeigt. Die formalen Strukturen zur Kodierung zeitbasierter

---

[45] Die Möglichkeit der Darstellung im Dualsystem läßt sich allgemein für jede natürliche Zahl zeigen und als Operation formalisieren."*Allgemein läßt sich zeigen, daß jede natürliche Zahl a als Summe geeigneter Faktoren von Potenzen einer natürlichen Zahl b (Basiszahl) in der Form* $a = a_n \cdot b^n + a_{n-1} \cdot b^{n-1} + ... + a_1 \cdot b^1 + a_0 \cdot b^0$ *eindeutig darstellbar ist, falls* $b > 1$ *und daher die Faktoren* $a_0, ..., a_n$ *nur die Werte* $0, 1, ..., b-1$ *annehmen.*" Mainzer, 1995, S. 39

[46] Kedrovskij, O.: Wechselbeziehungen von Philosophie und Mathematik im geschichtlichen Entwicklungsprozeß, 1984, S. 130

Entwicklungen sind Differentialgleichungen. So läßt sich beispielsweise das begrenzte Wachstumsverhalten einer Population mit der linearen Differentialgleichung $dM/dt = r(k/r - M)$ darstellen. Die Lösung der Gleichung ist eine Funktion und lautet $M(t) = k/r\ (1 - e^{-rt})$.[47] In diesem Fall ist die Lösung analytisch anzugeben und der Funktionsverlauf läßt sich exakt berechnen. Die Zustandsänderung der Population M ist die abhängige, die Zeit t ist die unabhängige Variable. Die Zustandsgröße M ist als gesuchte Unbekannte abhängig vom Systemparameter r, der die Wachstumsgeschwindigkeit charakterisiert. Die Begrenzung des Wachstums ergibt sich aus dem Systemparameter k, der beispielsweise die Sättigungsgrenze der Population in bezug auf eine begrenzte Nahrungsquelle angibt. Das System läßt sich komplexer gestalten, indem man eine Konkurrenzpopulation bezüglich der Nahrungsquelle einführt. Die Entwicklung des Gesamtsystems zeigt sich dann in der Wechselwirkung zwischen $M_1(t)$ und $M_2(t)$ anhand neuer Systemparameter. Die neue Differentialgleichung ist nichtlinear und es ist keine analytische Lösung bekannt, d.h. es gibt keine entsprechende Deduktion von der Gleichung zur Lösungsfunktion im Rahmen des formal-operativen Zeichensystems. Aussagen über die Lösung und damit die Dynamik des Systems, also die zeitabhängige Veränderung der Zustandsvariablen, können nur durch numerische Simulationen anhand ausgewählter Werte der Parameter, Anfangs- und Randbedingungen gewonnen werden.[48]

Für das beschriebene Beispiel ist die räumliche Verteilung uninteressant. Andere Anwendungen hingegen haben die zeitliche Entwicklung eines Systems im Raum zum Thema, wie beispielsweise die Strömungsdynamik. Um eine zeitabhängige räumliche Strömung zu simulieren, bedarf es der Lösung eines Systems partieller Differentialgleichungen (Erhaltungsgleichungen)[49] für Masse-, Impuls- (mit den drei Geschwindigkeitskomponenten) und Energiebilanzen für infinitesimal kleine Volumenelemente. Zur Bestimmung von fünf Unbekannten steht ein System aus fünf partiellen Differentialgleichungen zur Verfügung (Kontinuitätsgleichung, Energiegleichung und drei Impulsgleichungen).[50] Da keine analytische Lösung bekannt ist, müssen die Lösungen für eine raum- und zeitdiskrete Version der Gleichungen approximiert werden. Dazu bedarf es der Erzeugung eines Raumgitters, für dessen Berech-

[47] Vrgl. Abbildung 3 auf Seite 36. Die Lösungen linearer Differentialgleichungen sind in der Regel nichtlinear. Für lineare, gewöhnliche Differentialgleichungen lassen sich analytische Lösungsfunktionen angeben, doch bereits für lineare, partielle Differentialgleichen ist dies oft nicht mehr der Fall. Für nichtlineare, partielle Differentialgleichungen finden sich nur selten analytische Lösungen. Dann bleibt nur die Möglichkeit Lösungen der diskretisierten Gleichungen mit Hilfe der Computer zu berechnen.
[48] Die Terminologie wird in der Literatur unterschiedlich gehandhabt. So besteht für Hartmut Bossel ein System aus vielen Komponenten, deren Beziehungen zueinander vielfältig sind. Er unterscheidet Parameter (konstante Größen), Umwelteinwirkungen (vom System unabhängige Faktoren, die jedoch auf das System wirken), Zustandsgrößen, deren momentanen Werte den Zustand des Systems vollständig beschreiben, Anfangswerte der Zustandsgrößen, Veränderungsraten der Zustandsgrößen und Zwischengrößen (Größen die sich durch Umwelteinwirkungen oder Zustandsgrößen verändern). Vrgl. Bossel, 1989, S. 22ff. In der vorliegenden Arbeit wird zwischen *Parametern, Anfangs- und Randbedingungen* sowie *Zustandsgrößen* unterschieden, wobei letztere die gesuchten Unbekannten sind.
[49] Die Erhaltungsgleichungen basieren auf den Erhaltungssätzen der Physik. Je nach aufzulösenden Effekten erhält man die Euler- oder die Navier-Stokes-Gleichungen, wobei letztere im Unterschied zu den Euler-Gleichungen Reibungs- und Wärmeleitungseffekte berücksichtigen.
[50] Vrgl. Krause, E.: Einige grundsätzliche Aspekte numerischer Strömungssimulationen, 1996, S. 13ff

nungspunkte die Gleichungen gelöst werden,[51] sowie der Diskretisierung der Gleichungen selbst, indem die Differentiale durch endlich viele Differenzenquotienten substituiert werden.[52] Die Simulation strukturiert sich aus den Operationsvorschriften der diskretisierten Erhaltungsgleichungen und den Eingaben ausgewählter Werte der Parameter, der Anfangsbedingungen der Unbekannten (Zustandsgrößen), die sich auf experimentelle Beobachtungen oder theoretische Annahmen stützen sollten, und der Randbedingungen, welche das Verhalten der Zustandsgrößen am Rand des Berechnungsgitters definieren.[53] Die Algorithmisierung zerlegt die Gleichungen in ein System von $p \times u$ Gleichungen für p Gitterpunkt und u Unbekannte.[54] Um also die fünf Unbekannten der Erhaltungsgleichungen für 1000 Punkte zum Zeitpunkt t_1 zu berechnen, müssen 5000 Gleichungen generiert und gelöst werden.[55] Die Arbeit des Computers besteht nun darin, aus den numerischen Werten der Parameter, Anfangs- und Randbedingungen durch Einsetzungen und Umformungen die 5000 Lösungen für t_1 zu erzeugen, die sich der exakten Lösung annähern. Theoretisch wird angenommen, daß eine genügend große Verfeinerung des Raum-Zeit-Gitters zu immer besseren Näherungen an die exakte Lösung führt, daß sich also der Verlauf des Graphen der approximierten Lösungen zunehmend dem Verlauf des Graphen der exakten Lösung annähert. Da jedoch für die meisten nichtlinearen und einige lineare Gleichungen die exakten Lösungen unbekannt sind,[56] ist die numerische Simulation der einzige Weg, um zu Lösungsdarstellungen zu kommen, wenn auch nur zu einer approximierten. Die Lösungsalgorithmen ersetzen die nicht bekannten analytischen Lösungen und stellen insgesamt eine neue Methode zur numerischen Behandlung von Differentialgleichungen dar.[57] Während jedoch im Falle linearer Gleichungen jeder Schritt der algorithmischen Darstellung der numerischen Approximation inklusive der Fehlerabschätzung bezüglich der exakten Lösung begründbar ist, ist man für nichtlineare Gleichungen auf

---

[51] „*So lassen sich zum Beispiel die Anzahl der in der Reihenentwicklung berücksichtigten Terme und der Abstand der Gitterpunkte im Prinzip beliebig variieren. Da aus Genauigkeitsgründen die Abstände zwischen Gitterpunkten möglichst klein gewählt werden müssen, ist die Anzahl der zu lösenden Differenzengleichungen und damit auch die Anzahl der zu bestimmenden Unbekannten stets groß. ... Auch bei der Verwendung der größten und schnellsten heute zur Verfügung stehenden Rechenmaschinen sind zehn, ja sogar mehrere hundert Stunden Rechenzeit zur Berechnung komplexer Strömungsfelder nicht ungewöhnlich*.“ Krause, 1996, S. 14/15. Zur Erstellung des Programms einer solchen Simulation bedarf es mehrerer Mannjahre Arbeit.

[52] Als Alternativen zur Differenzenmethode gibt es die finite Elemente- und finite Volumenmethode als auch andere Methoden.

[53] Für einen Simulationslauf sind für t_0 die Parameterwerte, die Werte der Anfangsbedingungen und in der Regel der Randbedingungen im Programm vorgegeben und ändern sich während des Simulationslaufs nicht. Die Randbedingungen ergeben sich aus der Charakterisierung der künstlichen Berandung des zu berechnenden Strömungsgebiets, beispielsweise durch feste Wände und die dadurch bedingten Annahmen zur Flüssigkeitshaftung, Wärmefluß- und Temperaturverteilungen.

[54] „*Wenn der Abstand zwischen zwei Gitterpunkten auf dem Äquator* [für eine globale Klimasimulation] *beispielsweise 120 Kilometer beträgt, kann man sinnvolle einigermaßen realistische Berechnungen anstellen. Für jede Wetterprognose müssen dabei ungefähr 250 Millionen Unbekannte berechnet werden. Erst seit etwa 20 Jahren sind Superrechner so leistungsfähig, daß die Wetterberechnungen weniger Zeit erfordern, als die Wetterentwicklung selbst dauert*.“ Trottenberg, 1998, S. 7

[55] Gleichungslöser können aus mehreren hundert bis tausend Kommandozeilen bestehen, die für jeden Zeitschritt durchlaufen werden. Ein guter Gleichungslöser optimiert während der Berechnung aufgrund von Fehlerabschätzungen die räumliche und zeitliche Diskretisierung. Der gesamte Simulationskode kann weit über zehntausend Zeilen umfassen.

[56] Theoretisch wird davon ausgegangen, daß es zu jeder Differentialgleichung eine exakte, analytische Lösung gibt, die für alle Zeitpunkte, Anfangsbedingungen und Parameterwerte gilt. Während für lineare, gewöhnliche Gleichungen die analytischen Lösungsfunktionen bekannt sind, trifft das auf lineare, partielle Gleichungen nur bedingt zu. Es kann auch der Fall sein, daß aufgrund komplexer Randbedingungen die Lösung formal nicht angebbar ist.

[57] „*Sein* [John von Neumanns] *Anstoß war die Stagnation der analytischen mathematischen Methoden zur Lösung partieller Differentialgleichungen, vornehmlich in der Strömungsdynamik, und er wollte mit seinem Konzept des sequentiellen Digitalrechners, dessen Flexibilität seitdem den breiten Durchbruch des Computers in Wissenschaft, Wirtschaft und Gesellschaft bestimmt hat, den „digitalen Windkanal“ schaffen, um im Computer die Barriere der Stagnation zu durchbrechen*.“ Hoßfeld, 1996, S. 2

Heuristiken angewiesen, die sich aus der Übertragung von Erfahrungen mit der Simulation linearer Differentialgleichungen für nichtlineare ergeben. Es ist also für nichtlineare Probleme wesentlich schwieriger Lösungsverfahren zu finden sowie die Approximationsresultate zu bewerten, denn die begrenzte Genauigkeit der Zahlendarstellung im Computer führt dazu, daß aufgrund der Rundungen etwa bei Differenzenbildungen große Fehler entstehen können und die Lösungen numerisch instabil werden.[58] Da sich zwar für die meisten linearen, aber nur einige nichtlineare Problemstellungen die Richtigkeit der finiten Approximation nachweisen läßt,[59] ist die numerische Stabilität der Differenzenapproximation nur daran zu beurteilen, daß die Abbruch-, Rundungs- und Verfahrensfehler bei der Auflösung der Differenzengleichungen nicht beliebig anwachsen.[60] Zudem können die numerischen Instabilitäten durch das Verhalten der Gleichungen gegenüber spezifischen Parameterwerten, Anfangs- oder Randbedingungen bedingt sein, denn nur im Falle eines wohldefinierten Problems, für das die Existenz und die Eindeutigkeit der Lösung wie auch die stetige Abhängigkeit der Lösung von den Parametern, Anfangs- und Randbedingungen nachgewiesen ist, sind die auftretenden Instabilitäten Folgen falsch gewählter Diskretisierungen. Für ein nicht wohldefiniertes Problem können die auftretenden Instabilitäten strukturimmanent sein, und es läßt sich nur anhand von Studien zum Einfluß der Parameter, Anfangs- oder Randbedingungen ein Bild über das Lösungsverhalten gewinnen. Dabei können sogenannte Bifurkationen im Lösungsverhalten auftreten und zu determiniert chaotischen Zuständen führen, so daß sich Prognosen des Lösungsverhaltens sowie Aussagen über die Richtigkeit der Lösung schwierig, wenn nicht gar unmöglich gestalten. Es läßt sich also nicht vorhersagen, ob eine Lösung sich bei gering veränderten Parameterwerten oder Anfangsbedingungen stabil oder chaotisch verhalten wird und ob es sich um eine gute Approximation handelt. Kleine Ursachen können in nichtlinearen Systemen große Wirkungen haben. Da die Parameterwerte und Anfangsbedingungen jedoch bereits in ihrer empirischen Bestimmung aufgrund der Fehlertoleranz der Meßinstrumente ungenau sind, ist der Prognosewert simulierter Szenarien für naturwissenschaftliche Anwendungen nur von bedingter Aussagekraft.[61]

---

[58] „*Für gröbere Diskretisierung handelt man sich also einen Diskretisierungsfehler ein, für kleinere Diskretisierungen erhält man sehr plötzlich drastische Fehler wegen der Beschränktheit der numerischen Zahlendarstellung. Natürlich verbessert sich das Ergebnis bei höherer Genauigkeit der internen Zahlendarstellung, das grundsätzliche Problem aber bleibt: Bildet man numerisch eine Differenz etwa gleich großer Zahlen, so kann der Fehler leicht gegen 100% gehen!*" [Quelle 5: Müller-Krumbhaar, H.: Einleitung, S. 1.7]

[59] Der Unterschied zwischen einer linearen und einer nichtlinearen Struktur besteht in der analysierbaren Fehlerabschätzung bezüglich der Approximation für lineare Probleme, da die Struktur der Lösung für lineare Gleichungen bekannt ist.

[60] „*Der Laxsche Äquivalenzsatz sagt aus, daß der Nachweis der numerischen Stabilität die notwendige und hinreichende Bedingung für die Konvergenz der Lösung darstellt, wenn die Differenzenapproximation wieder in die zu approximierende Differentialgleichung übergeht, wenn die Abstände der Gitterpunkte gegen Null streben*." Krause, 1996, S. 15. Allerdings besteht hier dann das Problem, daß dies zu Fehlern führt, wie oben erwähnt, und pragmatisch nur begrenzt durchführbar ist.

[61] Diese Probleme schlagen sich in der Hypothetizität der dadurch erzielten wissenschaftlichen Aussagen nieder. „*Ein Gesichtspunkt ist die Komplexität der Zusammenhänge, die sich darin äußert, daß jede Wenn-Dann-Aussage durch die Überlagerung anderer Kausalverhältnisse, durch »intervenierende Variable«, außer Kraft gesetzt werden kann*." Bechmann, G. et al.: Sozialwissenschaftliche Konzepte einer interdisziplinären Klimawirkungsforschung, 1995, S. 77/78. Diese wissenschaftliche Situation ist typisch für Bereiche wie die Wettervorhersage oder die Klimaforschung, die sich mit komplexen Systemen beschäftigen. Variierende Anfangs- oder Randbedingungen können zu unterschiedlich Ergebnissen führen, wie die Diskussion um die Erder-

2.4 Semiotische Interpretation

Wie sieht die semiotische Interpretation dieser Vorgänge aus? Ein formal-operativer Zeichenumgang hantiert mit verschrifteten Zeichen. Operationen werden mit diesen Zeichen ausgeführt und die Erzeugung neuer Zeichen bzw. Zeichenfolgen besteht einerseits in der Umgestaltung der Folgen, andererseits in der Ersetzung definierter Folgen durch Symbole, die als Kürzel von Operationen zur Erzeugung dieser Folgen verstanden werden können. Das Integralzeichen oder der Differentialquotient mögen hier als Beispiele dienen. Gemäß den Regeln erlaubt der formal-operative Zeichenumgang die deduktive Umformung der formalen Strukturen. Diese Umformung kann im Falle algebraischer Gleichungssysteme zu Lösungsfunktionen für die gesuchten Unbekannten führen. Diese beschreiben einen Zusammenhang, der für jede beliebige Einsetzung numerischer Werte durch Befolgung der Funktionsvorschriften den entsprechend Wert der Unbekannten angibt. Der Unterschied zwischen der formal-operativen und der numerischen Methode besteht in der Reduktion der letzteren auf einen Typ von Zeichen: auf Zahlen, und damit auf eine Form: die numerischen Werte als Darstellung der Lösung. Kennt man nur das numerische Resultat, so lassen sich keine Rückschlüsse auf dessen Erzeugung samt Bedingungen gewinnen.[62]

Der Computer operiert mit digitalen, also numerisch kodierten Zeichen gemäß den instruierten Operationen, und die Simulation ist eine numerische Methode im Unterschied zur analytischen Methode der Lösungsfindung durch deduktive Umformungen algebraischer Gleichungen. Dabei werden durch numerische Umformungen der gegebenen Werte eines Gleichungssystems die Lösungen für die Unbekannten des Gleichungssystems berechnet. Dieses Vorgehen kann als *numerische Fallunterscheidung* der formalen Struktur für spezifische Parameterwerte, Anfangs- und Randbedingungen gelten. Dazu muß diese dem Organisationsprinzip der Computer angepaßt werden: Alle Variablen sind zu parameterisieren, und alle durch Operationszeichen artikulierten Beziehungen zwischen den Variablen sind in Rechenoperationen zu transformieren. Die formale Struktur wird in eine operative Struktur

---

wärmung zeigt. Dabei schlagen nicht nur die mathematischen Probleme der numerischen Simulation nicht-linearer Gleichungen zu Buche, sondern die Unkenntnis über Zusammenhänge und sensitive Parameter sowie die mangelnde Datengrundlage (Datenproblem).

[62] Bereits René Descartes weist auf die Vereinheitlichung durch die numerische Methode hin und kritisiert die *Rechner*: „... *wir dagegen* [können] *an dieser Stelle sogar von den Zahlen abstrahieren, ebenso wie kurz zuvor von den geometrischen Figuren und von jedem beliebigen Gegenstand. Wir tun das einerseits, um zum Überdruß langes und überflüssiges Rechnen zu vermeiden, andererseits vor allem, damit die Teile des Gegenstandes, die zur Natur der Schwierigkeiten gehören, immer getrennt bleiben und nicht durch unnütze Zahlen verhüllt werden. Wenn z.B. die Basis des rechtwinkligen Dreiecks gesucht wird, dessen Seiten 9 und 12 gegeben sind, wird der Rechner sagen, sie sei gleich ...* [Wurzel aus 225] *oder 15; wir aber werden 9 und 12 a und b setzen und die Basis als ...* [Wurzel aus $a^2 + b^2$] *finden. So bleiben die beiden Teile a^2 und b^2 getrennt, die in der Zahl miteinander verschmolzen sind*." Descartes, R.: Regeln zur Ausrichtung der Erkenntniskraft, 1972, S. 75. „*Dies alles unterscheiden wir, die wir eine evidente und deutliche Erkenntnis suchen, nicht aber die Rechner, die zufrieden sind, wenn ihnen das gesuchte Ergebnis unterläuft, selbst wenn sie nicht sehen, wie es von den Daten abhängt, obgleich allein darin die Wissenschaft eigentlich besteht*." Descartes, 1972, S. 77

überführt, die zum einen den Berechnungsvorgang darstellt, zum anderen diesen in Form instruierender Operationen auf der binärlogischen Ebene der Rechner ausführt. Der Lösungsalgorithmus kann als Simulationsmaschine bezüglich der zu simulierenden Gleichungen verstanden werden. Indem die Inputdaten der Parameter, Anfangs- und Randbedingungen eingegeben werden, erhält man nach endlich vielen maschineninternen Zustandsänderungen den Output der Lösungen. Zwischen den Anweisungen des Programms und der Präsentation der numerischen Lösungen steht der unanschauliche Prozeß der iterativen Berechnungen. Für die Frage nach den richtigen – also stabilen und sich der exakten Lösung annähernden – Approximationsresultaten lassen sich Plausibilitätsargumente anführen, die durch analytische Verfahren, Heuristiken und die experimentelle Arbeitsweise im Rahmen der Simulation anhand der Beurteilung der numerischen Resultate gestützt werden.

Der Betrachtung der visualisierten Ergebnisse kommt dabei eine große Bedeutung zu. Der Reiz der numerischen Simulation, die Begrenzung der formalen Methode zu erweitern, hat seinen Preis. Formal lösbare Probleme sind meist auch lineare. Nur wenige nichtlineare Gleichungen lassen sich auf diese Weise bearbeiten. Die numerische Simulation wagt sich somit in das Gebiet nichtlinearer Problemstellungen vor, in welchem mit Approximationen und Heuristiken gearbeitet wird. Von daher dürfen keine exakten Resultate erwartet werden. Es kann also nicht die Rede davon sein, daß Simulationen theoriengestützte Deduktionen sind. Zwar basieren die simulierten Gleichungen ausschließlich auf formalisierten Theorien, doch die Sicherheit deduktiver Umformungen bieten sie nicht.[63] Hinzu kommen pragmatisch bedingte Schwierigkeiten, die aus der Unanschaulichkeit der Rechenprozesse, deren Geschwindigkeit und dem Umfang der Kodierung resultieren. Dies zeigt sich zum einen in der Begrenzung der Handhabung komplexer Simulationen, da die Kontrolle der Programmierfehler mit zunehmender Komplexität schwieriger wird, und zum anderen darin, daß die Unterscheidung zwischen Effekten, die durch die formale Struktur kodiert sind, und Effekten, die sich durch Fehler oder unadäquate Diskretisierungen, Berandungen etc. ergeben, nicht einfach ist.[64]

[63] Da Simulationen auf Theorien basieren, katalysieren sie keine Theoriendynamik und sind deshalb Bestandteil der normalen Wissenschaft im Sinne von Thomas Kuhns Konzeption wissenschaftlichen Fortschritts. Vrgl. Kuhn, 1993. Doch die Neuheit der visualisierten Darstellung sowie die Erweiterung der Lösungsmöglichkeiten, indem auf Heuristiken zurückgegriffen wird, erweiteren den Spielraum der Verwendung wissenschaftlicher Theorien. Dies kann zu neuen Erkenntnissen führen - allerdings im Rahmen bekannter Theorien -, die sich experimentell verfizieren oder falsifizieren lassen. Genutzt wird dieser Erkenntnisweg beispielseise im molecule engineering, indem neue Moleküle im Computer gestaltet und anschließend im Labor synthetisiert werden.

[64] „*Eine präzise Simulation stößt an natürliche Grenzen, wo kleine Änderungen in den Daten große instabile Effekte bewirken und eine chaotische Dynamik dominiert. In vielen Fällen, zum Beispiel in der Klimaforschung, ist es jedoch weitgehend ungeklärt, ob die Unsicherheiten durch zu ungenaue Numerik (viel zu große Gitterstrukturen ...), durch Modellierungsfehler oder tatsächlich durch das potentiell chaotische Verhalten der Realität selbst (Klimakatastrophe) bedingt ist.*" Trottenberg, 1998, S.10. Determiniert chaotische Zustände treten jedoch nur in nicht wohldefinierten Problemstellungen auf.

3. Entfaltung der Form - Zeitlichkeit

3.1 Raum-Zeit-Raster

Simulationen sind maschinell ausführbare Erzeugungsmechanismen numerischer Werte zur semiotischen Modellierung dynamischer Prozesse, insofern sich die Dynamik im errechneten Werteverlauf der gesuchten Unbekannten artikuliert. Die Dynamik wird dabei über die Symbolisierung durch Zeichen hinaus direkt auf semiotischer Basis realisiert. Dazu ist ein fluides Medium von Nöten, das die Implementierung eines dynamisch organisierten Symbolschemas erlaubt. Der Dynamisierung der Simulation liegen verschiedene Zeitformen zugrunde: Im Gesamten zeigt die Simulation einen Prozeß in einem definierten Raum-Zeit-Raster, das sich aus dem Berechnungsraum und den Berechnungsschritten ergibt. Im Detail vollziehen sich die Berechnungen als rekursive Operationen, die an geeigneter Stelle abgebrochen werden und eine approximierte numerische Lösung ausgeben. Der Einblick in die Rekursion, die im Raum-Zeit-Gitter dargestellt wird, veranschaulicht anhand der Visualisierung die Dynamik der Datenstrukturen bzw. des Lösungsverhaltens der berechneten Gleichungen. Differentialgleichungen stellen kontinuierliche Prozesse in Raum und Zeit dar.[65] Der Differentialquotient, der sich mit beliebiger Genauigkeit an jeden Punkt des Verlaufs der exakten Lösungsfunktion annähert, symbolisiert die infinitesimale Operation der Differentiation. Dabei wird von den pragmatischen Bedingungen des Berechnens für endliche numerische Werte abstrahiert.

Für den formal-operativen Umgang mit Strukturen in der Mathematik, die von infinitesimalen Größen handeln, haben sich seit der Neuzeit Rechenregeln entwickelt. Die Zeit selbst wird dabei als Größe interpretiert, die sich mit beliebiger Genauigkeit angeben läßt und in ihrer Dynamik als Zeitstrahl mit dem Zahlenstrahl der reellen Zahlen identifiziert wird.[66] Kleinste Zeiteinheiten einer atomistischen Zeitauffassung werden durch infinitesimale Approximationsprozesse substituiert, wie sie für die reellen Zahlen und Differentiale charakteristisch sind.[67] Indem Zeitpunkte zu reellen Zahlen werden und reelle Zahlen durch infinitesimale Approximationsprozesse definiert sind, wird die Integration der Zeit als reellwertige Variable in mathematische Strukturen möglich. Die Gleichsetzung infinitesimaler Approximationsprozesse mit der Idee des Kontinuums ersetzt einerseits eine atomistische Auffassung und verweist andererseits das Kontinuum in die symbolische Sphäre des schriftbasierten Zeichenumgangs, das Annahmen über die Kontinuität natürlicher Prozesse evoziert. Tatsächlich sind Berech-

[65] Raum und Zeit sind die unabhängigen Variablen der Gleichungen.
[66] Die mathematische Darstellung der Zeit auf einer Zeitkoordinate geht auf Nikolaus Oresme zurück. Vrgl. Mainzer, K.: Zeit, 1995a, S. 32ff
[67] Die reele Zahl a läßt sich durch die rationale Zahlenfolge a_* als $a(a_*)$ bzw. als $\lim_{n \to \infty} a_n$ darstellen.

nungen, Messungen und Wahrnehmungen jedoch immer dskret.[68] Die Inkommensurabilität zwischen den formal-operativen Darstellungsmöglichkeiten und den maschinell umgesetzten Anwendungsmöglichkeiten zeigt sich nirgends deutlicher als in der numerischen Simulation partieller Differentialgleichungen. Numerische Simulationen vollziehen sich in einem endlichen Berechnungsraum, dessen Dimensionen sich aus der Anzahl der Variablen ergeben, von welchen die zu berechnenden Unbekannten abhängig sind.

Die Darstellung des Berechnungsraums als dreidimensionale Raum-Gitter-Abfolge in der Zeit transformiert den n-dimensionalen Berechnungsraum als Wirkungen der Unbekannten auf die Raumpunkte in den dreidimensionalen Anschauungsraum, und die Dynamik zeigt sich in der zeitlichen Veränderung der Wirkungen.[69] Der Berechnungsraum der Simulation charakterisiert sich aufgrund der Endlichkeit der Ressourcen an Berechnungszeiten und Speicherkapazitäten durch eine atomare Zeit- und Raumauffassung. Die prinzipiell für jeden Raum- und Zeitpunkt geltenden symbolischen Gleichungen können nur für ein mehr oder weniger feines Raster von Berechnungspunkten gelöst werden.[70] Dabei ersetzt der Vorgang der Diskretisierung die Differentiale durch Differenzenquotienten, die zwischen den Werten der Unbekannten für die Berechnungspunkte *i, j* und *i, j-1* einen Zusammenhang für den Zeitpunkt t_1 darstellen: u_x » $(u_{i,j} - u_{i,j-1})$ / ¥*x*. Die Erzeugung des Berechnungsgitters besteht in der Berechnung der Werte für ¥*x*, die dann in die Gleichungen der Differenzenquotienten eingesetzt werden.

Die Gestalt des Berechnungsgitters bzw. des Raum-Gitters ist durch die Gitterstrukturierung sowie die Randbedingungen bestimmt. Während erstere die grundlegende Struktur zur Erzeugung des Gitters definiert,[71] enthalten die Randbedingungen Parameter bezüglich der Begrenzung und Charakterisierung des Verhaltens der Zustandsgrößen am Gitterrand als auch bezüglich der Abgrenzung von Binnenformen wie beispielsweise zweier Flüssigkeiten, deren Einfluß aufeinander unter spezifischen Bedingungen getestet wird, oder für Objekte in einem Strömungsmedium. Die für die Simulation interessanten Prozesse spielen sich dabei an jenen Berechnungsumgebungen ab, an welchen die beiden

[68] Auch unsere Wahrnehmung ist aufgrund der Geschwindigkeitsbegrenzung der Nerven diskret.

[69] Das auf Seite 66 besprochene begrenzte Wachstum handelt beispielsweise nicht von räumlichen Prozessen. Der zweidimensionale Darstellungsraum der Graphik repräsentiert weder ein Raum-Gitter, noch den gesamten Berechnungsraum, sondern den Lösungsraum für *M(t)* in Abhängigkeit von der Zeit. „*Es gibt eine weite Klasse von Systemen, bei denen es gerade auf die räumliche Verteilung der Zustandsgrößen ankommt, bei denen also zwischen benachbarten Punkten Gradienten der Zustandsgrößen bestehen, die durch weitere Differentialquotienten, diesesmal für die Raumkoordinaten dargestellt werden müssen. Müssen also in eine solche Systemdarstellung Differentialquotienten in bezug auf mehr als eine Koordinate ... aufgenommen werden, so spricht man von partiellen Differentialgleichungen*.“ Bossel, 1989, S. 270

[70] „*Dazu werden die Änderungsgeschwindigkeiten (Ableitungen) der einzelnen Variablen durch algebraische Ausdrücke ersetzt, die auf den endlichen Größen Dx und Dt beruhen. Die ursprünglich an jedem Punkt in Raum und Zeit definierten stetigen Gleichungen werden so zu algebraischen Gleichungen, die nur an einer endlichen Zahl von Gitterpunkten exakt sind, die dafür aber leicht gelöst werden können. Benutzt man bei der Lösung ein ausreichend feines Gitter, so nähert sich die numerische Lösung der der ursprünglich exakten Gleichung*.“ Kaufmann/Smarr, 1994, S. 71

[71] Das Verfahren der Triangulation beispielsweise ist zur Gittergenerierung geeignet.

Medien aufeinander treffen und sich gegenseitig beeinflussen. Da sich an diesen Stellen die eigentlichen Veränderungen ereignen, bietet es sich an, das gleichförmige Berechnungsgitter lokal zu verfeinern. Dies kann bereits bei der Erzeugung des Gitters geschehen oder während der Berechnungsschritte durch adaptive Gitteranpassung. Dort also, wo sich das Lösungsverhalten erwartungsgemäß stärker ändert, empfiehlt sich eine höher auflösende Raumdiskretisierung.

Theoretisch wird davon ausgegangen, daß bei einer immer feiner werdenden Diskretisierung die simulierte Lösung der exakten Lösung beliebig nahe kommt, also konvergent ist. Dies muß im Konkreten nicht der Fall sein, auch wenn die Konvergenz theoretisch bewiesen wurde. Denn aufgrund der endlichen Darstellbarkeit numerischer Werte im Computer durch Rundungen kann die Differenzenbildung annähernd gleich großer Werte zu großen Fehlern führen. D.h. die Approximation nähert sich bis zu einem bestimmten, sich verringernden Wert von ¥*x* der exakten Lösung an. Wird der Wert unterschritten, können sich Instabilitäten einstellen.[72] Die Umwandlung des Zeit-Kontinuums der Gleichungen in eine diskrete Zeitabfolge definiert den Rahmen, innerhalb dessen sich die Simulation abspielt. Dabei bilden Simulationen „... *die reale Zeit auf eine Simulationszeit oder Modellzeit ab. Die Simulationszeit wird durch eine Variable dargestellt, die wie die reale Zeit wachsende Werte annimmt. Diese Variable wird auch Simulationsuhr genannt. ... Während die reale Zeit stetig wächst, kann die auf einem Computer dargestellte Simulationsuhr nur sprunghaft wachsen. Diese Sprünge können von gleicher Länge sein und zu äquidistanten Zeitpunkten führen, oder sie können sich an den Zeitpunkten orientieren, wo sprunghafte Wert- oder Zustandsänderungen stattfinden, die man als Ereignisse (Events) bezeichnet*."[73] Simulationen partieller Differentialgleichungen sind Zeitfolgensimulationen, für welche die Veränderung der Lösung eine stetige Funktion der Simulationszeit ist.[74] Dabei müssen die Zeitpunkte nicht über den gesamten Simulationsverlauf äquidistant sein und können für ausgewählte Zeitfenster verfeinert werden. Dies empfiehlt sich, wenn die approximierten numerischen Werte Insta-

[72] Vrgl. Anm. 210, Seite 68. Dazu kommt erschwerend hinzu: „*Eine eindeutige Auswahl bei der Festlegung der Gittergenerierung, die Diskretisierung wie auch bei der Auswahl des Lösungsverfahrens ist selten möglich, da die Lösungen oft durch die Eigenarten der Anfangs- und Randbedingungen stark beeinflußt werden*." Krause, 1996, S. 24

[73] [Quelle 6: Lorenz, P.: 1.2 Simulation, 1999, S. 6]. Die Simulationsuhr kann ganzzahlige oder reellwertige Werte anzeigen. „*Reellwertige Variablen haben den Vorteil eines größeren Wertebereichs. Wenn reellwertige Variablen große Werte annehmen, können kleine Inkremente verloren gehen. Außerdem kann die Feststellung oder Prüfung der Gleichzeitigkeit von Ereignissen problematisch werden*." [Quelle 6: Lorenz, 1999, S. 6]. Die Endlichkeit der Zahlendarstellung im Rechner ist auch hier das Problem. Üblicherweise besitzen reellwertige Variablen 15 gültige Dezimalziffern (1E15). Ein Zeitzuwachs von 1 (1E16) kann nicht vollzogen werden und führt zur Programmwarnung: *Zahlenbereichsüberschreitung, kein Zeitzuwachs*. Bei gannzahligen Variablen sind die Zeiteinheiten so klein zu wählen, daß keine Abbildungsfehler entstehen. Der größte Wert der ganzzahligen Zeitvariable beträgt in der Regel $2^{31} – 1$.

[74] Es handelt sich dann um eine sogenannte Taktsimulation bzw. Zeitfolgesimulation. Vollziehen sich die Sprünge zwischen den Zeitpunkten von Ereignissen so wird die Simulationsuhr für jeden Sprung auf diese Zeitpunkte eingestellt (Ereignissimulation). „*Während der Abarbeitung dieses Simulationsprogrammes läuft die Simulationsuhr oder Simulationszeit, und es werden Zustandsänderungen in einer zeitlichen Reihenfolge nachgebildet, die dem Zeit- und Prozeßablauf im Original entspricht. Die Simulationsuhr läuft in der Regel mit wechselhaftem Tempo. Sie kann sich in gleich oder ungleich langen, diskretisierten Zeitschritten bewegen und hat dabei keine festliegende Beziehung zur realen Zeit. Sie verkörpert die Modellzeit und ist Abbild der realen Zeit*." [Quelle 6: Lorenz, 1999, S. 1]

bilitäten aufweisen.[75] Prinzipiell ist die Zeitdiskretisierung von den Zeitskalen der Gleichungen abhängig.[76] Bezogen auf die Realzeit kann die Simulationszeit eine Zeitraffung oder Zeitlupe darstellen. Sie ist abhängig von der Leistungsfähigkeit des Computers und von der Effektivität des Programms. Das Ziel ist es, Proportionalität zu schaffen und die Simulationszeit mit der Realzeit zu synchronisieren. Anschaulich wird die Simulationszeit durch die Animation der Visualisierungsbilder einzelner Zeitschritte zu einer Folge von Bildern. In der Bilderfolge entfaltet sich die Dynamik der Datenstrukturen.

Für die Zeitdiskretisierung bieten sich zwei Möglichkeiten an, ein explizites und ein implizites Verfahren. Während für das explizite Verfahren der Wert für u_{n+1} von Zeitschritt zu Zeitschritt – ausgehend vom Anfangswert u_n - direkt erzeugt wird ($u_{n+1} = g(u_n)$), indem die Funktion g mit Hilfe eines Algorithmus lediglich einmal ausgewertet, also berechnet werden muß, wird die Lösung für u_{n+1} in der impliziten Methode durch Rekursionen gewonnen ($f(u_{n+1}) = g(u_n)$).[77] Dazu muß f invertiert werden, d.h. nach u_{n+1} aufgelöst und rekursiv berechnet werden. Der Vorteil des impliziten Verfahrens besteht in der Berücksichtigung der Wechselwirkung aller alten Lösungen für das neue Zeitniveau und führt zu stabileren Lösungen. Demhingegen werden mit dem expliziten Verfahren nur die in der partiellen Ableitung auftretenden Lösungen berücksichtigt.[78] Für das explizite Verfahren, das direkt von Zeitschritt zu Zeitschritt rechnet, bedarf es deshalb einer sehr kleinen Zeitschrittweite, denn zu grobe zeitliche Diskretisierungen können zu drastischen Fehlern und Instabilitäten führen.[79]

3.2 Rekursion

Der Begriff der Rekursion oder Iteration handelt von der Wiederholung einer definierten Operation, im Falle der Simulation von der wiederholten Berechnung der Zustandsgrößen auf Basis der jeweils zuvor erzeugten Werte.[80] Die Simulation besteht zum einen aus iterativen Schritten zur Berechnung der

[75] Ähnlich der räumlichen Diskretisierung durch die adaptive Gitteranpassung wird die zeitliche Diskretisierung für die Berechnung dann verfeinert, wenn die Approximation der Lösung mit der Fehlerschätzung nicht harmoniert. Gute Algorithmen nehmen diese Verfeinerung aufgrund einer optimierten Fehlerabschätzung selbständig vor.
[76] „*Bei Systemen gekoppelter Differentialgleichungen mit stark verschiedenen inneren Zeitskalen führt dies jedoch dazu, daß die Wahl des Diskretisierungsparameters immer durch die kürzeste Zeitskala nach oben beschränkt ist, selbst wenn auf den kurzen Zeitskalen physikalisch nichts interessantes passiert*." [Quelle 5: Müller-Krumbhaar, S. 1.9]
[77] Die Variable u_n bzw. u_{n+1} kann für ein oder mehrere Unbekannte stehen.
[78] In partiellen Ableitungen erster Ordnung tritt nur die Beziehung zwischen zwei benachbarten Berechnungspunkten auf, für partielle Ableitungen höherer Ordnung dementsprechend die Beziehung zwischen mehreren Berechnungspunkten.
[79] Wird beispielsweise die Zeitschrittweite *Dt* für das durch die Gittergenerierung bestimmte Raumintervall zu groß gewählt, um die Berechnungen zu beschleunigen oder da die Rechenkapazitäten nicht ausreichen, können die Abweichungen der numerischen Lösungen von den genauen Lösungen dramatisch werden und zu schwingen beginnen. Dieses Problem kann vermieden werden, indem *Dt* ausreichend klein gewählt wird oder Ableitungen höherer Ordnung verwendet werden, die für größere *Dt* nicht zu Instabilitäten führen.
[80] Der Begriff der Iteration wird zur besseren Unterscheidbarkeit für die Abläufe auf der Ebene des Zeitrahmens der Simulation verwendet, der Begriff der Rekursion für die Abläufe zwischen den Zeitschritten.

Übergänge von Zeitschritt zu Zeitschritt, ausgehend von ausgewählten Anfangsbedingungen (Iteration). Zum anderen arbeitet das implizite Verfahren mit rekursiven Berechnungen zwischen den einzelnen Zeitschritten (Rekursion). Das hat den Vorteil, daß das Ausbreitungsverhalten aller Lösungen für den neuen Zeitschritt berücksichtigt wird. Dabei gestaltet sich der Rekursionsmechanismus als eine Schleife im Programm, die ausgehend von einem Startwert – dem Anfangswert der gesuchten Unbekannten – den approximierten Wert als neuen Startwert wieder einsetzt, solange bis ein Lösungswert der Zustandsgröße erzielt ist, der entsprechend den Fehlerabschätzungen als solcher für ausreichend genau für den darauffolgenden Zeitschritt eingeschätzt wird. Nach mehreren hundert oder tausend Wiederholungsschritten wird die Berechnung an geeigneter Stelle abgebrochen, und der Wert dient als Startwert der Unbekannten für die Berechnung des nächsten Zeitschritts. Auf diese Weise entsteht ein Muster dynamischer Operationen, die ineinandergreifen. Die Methode der Rekursion erlaubt es, im Prinzip unendlich viele Werte gemäß den Operationsvorschriften aus der zugrundeliegenden formalen Struktur zu erzeugen. Pragmatisch ist dies jedoch nicht durchführbar, weshalb es interner Abbruchkriterien bedarf.

3.3 *Dynamik der Datenstrukturen*

Die Rekursion und die Iteration bilden das syntaktische Muster der Simulation, das, angewandt auf eine formale Struktur, eine Kaskade numerischer Werte entfaltet. Die tatsächliche Berechnungszeit eines Simulationslaufs ergibt sich jedoch auf der Ebene der Berechnungen und ist ohne Bezug zur Simulationszeit. Da die rekursiven Berechnungen zwischen einem Zeitschritt lösungsabhängig sind, können sie in ihrer Zeitdauer erheblich variieren und der Berechnungszeitraum kann die Simulationszeit übersteigen. Folgen hat dies beispielsweise für die Wetterprognose, wenn die Berechnung mehr Zeit erfordert als die Wetterentwicklung des Vorhersagezeitraums.[81] Numerisch simulierte Differentialgleichungen erzeugen durch die Entfaltung der Resultate eine inhärente Dynamik, die sich aus der lösungsabhängigen Strukturierung der Veränderungen der numerischen Werte entlang der Zeitschritte ergibt (Werteverlauf). Die Visualisierung dieser Dynamik gibt Aufschluß über die mit den Gleichungen dargestellten und durch die Berechnungen konkretisierten Beziehungen zwischen den Variablen eines Prozesses. Der Informationsgehalt, der in der Struktur der Gleichungen kodiert ist und nun numerisch entfaltet wird, zeigt sich in der Dynamik der Datenstrukturen und variiert mit deren Änderungsrate respektive mit ihrer Gestalt. Auf diese Weise generieren Simulationen eine eigene Semantik, die als Grundlage heuristischer Annahmen über die Simulation, das Lösungsverhalten der diskretisierten

[81] Vrgl. Anm. 174, Seite 56.

Gleichungen für spezifische Parameterwerte, Anfangs- und Randbedingungen sowie den simulierten Prozeß dient. Die numerischen Resultate sind zwar durch die Gleichungen determiniert, aber nicht immer vorhersagbar, da aufgrund der Komplexität die einzige Form der Artikulation in ihrer numerischen Simulation besteht. Oder anders gesprochen: Der Zusammenhang zwischen den Gleichungen und den numerischen Lösungen ist mangels nicht bekannter analytischer Lösungen oder Lösungsstrukturen nur bedingt analysierbar und ergibt sich aus der Betrachtung der Dynamik der Datenstrukturen, also aus *quasi-empirischen* Interpretationen der Computerexperimente. Sie sind die sichtbaren Darstellungen der *numerischen Fallunterscheidungen* der formalen Gleichungen und erzeugen keine generellen Aussagen über die Gleichungen, sondern lediglich singuläre, bedingte Hinweise.[82]

Aus systemtheoretischer Sicht stellt eine Simulation die strukturellen Beziehungen zwischen den Variablen (Parametern, Zustandsgrößen, Randbedingungen) eines Systems für ausgewählte Anfangsbedingungen dar, wobei die Strukturverknüpfungen des Systems durch Beobachtung, Abstraktion und Theorie gewonnen wurden. Diese Beziehungen zeigen sich in den Wirkungen, welche die Parameter und Zustandsgrößen aufeinander haben. Die Wirkungen lassen sich funktionalisieren, indem bestimmt wird, „*ob zwei auf ein Element treffende Wirkungen ... addiert, multipliziert oder anderwertig miteinander verrechnet werden müssen und ob z.B. eine komplizierte funktionale Verknüpfung analytisch oder durch eine Tabellenfunktion vorgeschrieben werden kann*."[83] Die Funktionalisierung mit Hilfe formaler Strukturen (Gleichung) bedarf der Quantifizierung, um auf einem Computer ausführbar zu sein. Das bedeutet, daß alle formal artikulierten Variablen quantifiziert, d.h. für ausgewählte numerische Werte bestimmt werden müssen.[84] Dieses Vorgehen legt die Annahme zugrunde, daß sich im Rahmen der Funktionalisierung und Quantifizierung die konstitutiven strukturellen Merkmale eines Systems inklusive der in den strukturellen Beziehungen enthaltenen Informationen erhalten. Die Simulation der funktionalisierten Strukturen entfaltet nun numerisch die in den formalen Strukturen kodierten Informationen bezüglich der systeminternen Wirkungen der Parameter und Zustandsgrößen aufeinander.[85] Daraus ergibt

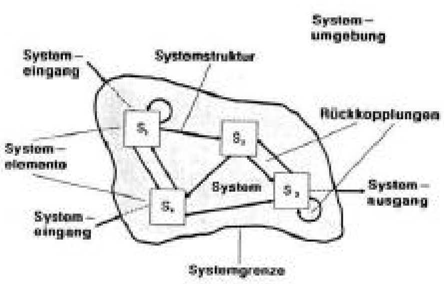

Abb. 8: Systemkonzept

[82] „*Steht hingegen ein analytisches Lösungsverfahren nicht zur Verfügung, dann lassen sich durch Simulation nur partielle bzw. singuläre Ergebnisse (konkret: Zahlenreihen) erzielen, die – nach geeigneter Auswertung – durch Induktionsschluß zu Funktionalzusammenhängen verallgemeinert werden*." Mückl, W. : Simulation als methodisches Problem, 1981, S. 192. Diese Ergebnisse werden dann zur Erklärung der Wirklichkeit, zu Prognosezwecken oder für Entscheidungshilfen verwandt.
[83] Bossel, 1989, S. 31
[84] „*Das bedeutet, daß alle externen oder Systemparameter, die Anfangswerte der Zustandsgrößen und zusätzlich die Quantifizierung aller funktionalen Verknüpfungen zwischen Systemelementen bestimmt werden müssen*." Bossel, 1989, S.31
[85] Der Abstraktionsweg vollziet sich wie folgt: Systembeobachtung, strukturelles Konzept, Formalisierung, Funktionalisierung, Algorithmisierung, Quantifizierung, Simulation, Visualisierung, Interpretation.

sich das Erkenntnisinteresse bezüglich der Simulationsresultate in Form von Sensitivitäts-, Wirkungs- oder Präferenzanalysen. Bei der Sensitivitätsanalyse geht es um die Nutzung der Simulation für Erkenntnisse über die Sensitivität des Systemverhaltens bezüglich des Einflusses bestimmter Parameter auf das Lösungsverhalten. Die Wirkungsanalyse konzentriert sich auf die Sensitivität bezüglich ausgewählter Ziel-Variablen, um einen definierten Zielzustand des Systems optimal zu realisieren. Die Präferenzenanalyse geht von alternativen Präferenzen gegenüber der Untersuchung eines Systems aus, das für verschiedene Parameter sensitiv und für unterschiedliche Ziel-Variablen bestimmt ist.[86]

Zeitlich betrachtet entfaltet sich mit der Dynamik der Datenstrukturen das inhärente Verhalten der simulierten Prozesse.[87] Sind die Lösungen für einen Raumpunkt von Zeitschritt zu Zeitschritt gleichbleibend, so geht deren Dynamik gegen Null und die Variablen haben keine Wirkung aufeinander in der Zeit. Verändern sich die Lösungen jedoch von Zeitschritt zu Zeitschritt, weisen also ein dynamisches Verhalten auf, bedeutet dies, daß sich die inhärenten Wirkungen in der Zeit zeigen. Die funktionalisierten Wirkungen beeinflussen die Veränderungsrate der Zustandsgrößen und können von mäßig bis turbulent beschaffen sein.[88] Die numerische Simulation setzt sich aus zahlreichen Berechnungsebenen zusammen, die sich unterschiedlich darstellen lassen, wie beispielsweise als Lösungsentwicklung einer bestimmten Zustandsgröße oder in Form von Parameterstudien.[89] Hier gewinnt die ikonische Visualisierung ihre Bedeutung, denn die Dynamik der Datenstrukturen, als Veränderungen der Lösungswerte im Raum-Gitter über mehrere Zeitschritte hinweg, zeigt sich im Gesamten nur in der Bildabfolge.[90] Die Entfaltung der Form der Simulation bringt also den Aspekt der Dynamik ins Spiel, der bislang dem klassischen Artikulationsinstrument der Schrift für die Organisation seiner Symbole verschlossen war. Die Dynamisierung der formalen Strukturen ist das Resultat der Formalisierung und Mechanisierung der Schrift. Dabei vereint die Entfaltung der Form der Simulation die unterschiedlichen Zeitkonzepte der verschiedenen Ebenen, wie die lineare Entwicklung der Simulationszeit (Zeitpfeil), das lösungsabhängige Zeitmuster der Rekursionen und die systeminterne Dynamik der Daten-

[86] Vrgl. Mückl, 1981, S. 198ff

[87] Das Konzept eines äußeren Zeitrahmens und der inhärenten Zeit der Prozesse leitet sich aus dem Zeitkonzept Isaac Newtons her, das er für seine Mechanik und das mathematische Instrument der Differentialgleichung entwickelte. Es basiert auf der Idee einer absoluten Zeit und eines absoluten Raums. Zeit wird dadurch zu einer meßbaren und berechenbaren Größe und zur reellen Koordinate in den Bewegungsgleichungen der Mechanik. Die absolute Zeit ist nach Newton ein feststehender äußerer Rahmen zur Beschreibung von Prozessen, dessen metrische Struktur im Unterschied zu den nicht unbedingt gleichförmigen Bewegungen der verschiedenen Prozesse und ihrer Zeitskalen geordnet ist. Bezüglich der Newtonschen Zeitauffassung vrgl. Mainzer, 1995a, S. 32ff

[88] Chaotische Zustände liefern trotz Determiniertheit keinerlei Informationen über die Wirkungszusammenhänge. Determiniert chaotische Zustände treten jedoch nur in nicht wohldefinierten Problemstellungen auf, für die keine stetige Abhängigkeit der Lösungen von den Parametern, Anfangs- oder Randbedingungen nachgewiesen werden konnte.

[89] Mit Parameterstudien wird der Einfluß bestimmter Parameter auf das Lösungsverhalten der Zustandsgrößen untersucht. Diese Studien sind die Grundlage der sog. Chaosforschung und haben mit der Darstellung der gesuchten Lösung eines Systems nichts zu tun.

[90] Natürlich läßt sich der Werteverlauf für einzelne Zahlenreihen auch anhand der Zahlendarstellung einsehen. Auf der Ebene der Rekursionen wird die Beurteilung der Lösungsentwicklung von den Algorithmen vorgenommen.

strukturen bzw. des Lösungsverhaltens. All diese Zeitkonzepte passen sich der computerbasierten Organisationsweise an, indem sie auf diskreten Zeiteinheiten basieren und sich von der Kontinuumsidee der schriftbasierten Symbolik unterscheiden. Während die zeitliche Diskretisierung der Simulationszeit und das Zeitmuster der Rekursionen algorithmisch bestimmt sind, ist die Dynamik der Datenstrukturen das eigentliche Resultat des Simulationsvorgangs, das nur in der ikonischen Darstellung intuitiv erfaßbar ist. So wie sich die Bedeutung von Texten aus der semantischen Kodierung der Zeichen erschließt, so erschließt sich die Bedeutung der Simulation aus der anschaulichen Darstellung der Dynamik des Lösungsverhaltens und konstituiert die Semantik numerischer Simulationen. Realisiert die typographische Form Informationen als Texte, indem differenzierte und disjunkte Zeichen und Zeichenfolgen in einer normierten Form bereitgestellt werden, deren Bedeutung sich aus der Interpretation der Zeichenfolgen ergibt, so gilt dies auch für die Simulationen, wobei sich die Informationen aus den dynamischen Zeichenoperationen ergeben und die Interpretation auf die visualisierte Dynamik der Datenstrukturen Bezug nimmt. Während Texte jedoch Beschreibungen von Systemabläufen im Medium der Schrift realisieren, stellen Simulationen funktionale Nachbildungen dieser Prozesse bzw. konstitutiver struktureller Aspekte anhand der regelbasierten Entfaltung von Werteverläufen dar, wobei die Werte zugleich als maschineninterne Zustände und visualisierte Präsentationen existieren.

4. Präsentation der Form - Bildlichkeit

4.1 Strukturierungen der Datenbasis

Die Visualisierung transformiert die numerischen Werte in farbige Pixeldarstellungen und fügt auf diese Weise zahlreiche individuelle Werte zu intuitiv erfassbaren Strukturen zusammen. Sich verändernde Farbwerte kreieren dabei den Eindruck der Dynamik der Datenstrukturen als Wechsel diskreter Ereignisse, die nur als singuläre Ereignisse, symbolisiert mit Zahlen, lesbar wären. Die Wahrnehmung der Dynamik würde dann jedoch in der Kolonne von Zahlen verschlüsselt sein, und ihre Entwicklung wäre nur schwer oder gar nicht einsichtig. Der Informationsgewinn gegenüber der Darstellung mit Ziffern besteht in der visualisierten Entfaltung der relationalen Strukturen zwischen den numerischen Werten. Wie bereits skizziert, besitzen digitale Zeichen eine eindeutige Kennzeichnung im Rahmen eines Programms (Adresse, Variablentyp, Wert). Die Simulation versieht die errechneten Resultate jedoch mit weiteren Informationen, die sich aus dem Raum-Zeit-Raster ergeben. Es ist also festgelegt, für welchen Berechnungspunkt und Zeitschritt ein spezifischer Wert erzeugt wurde. Auf diesem Wege werden die numerischen Werte zu raumzeitlich lokalisierten Daten.[91] Zur Aufbereitung der Daten einer Simulation für die Visualisierung bedarf es eindeutiger Angaben über die Art des Zahlenmaterials (binary -, floating-point -, double-precision floating-point numbers, etc.), über die Dimensionalität des Datensatzes und die Form der Speicherung (Matrix oder Liste) sowie über die Verteilungsstruktur der Daten (grid, nodes, cells).

Alle relevanten Angaben bedürfen einer strukturierten Form der Darstellung in Form einer Matrix oder Liste.[92] Dabei referiert die Datenstruktur bereits auf die Bedingungen der visuellen Darstellbarkeit, wie sie vor allem durch die begrenzte Dimensionalität des Anschauungsraums vorgegeben sind. Die Transformation des n-dimensionalen Berechnungsraums in eine matrix- oder listenförmige Darstellungsweise bildet einen Zwischenschritt zwischen den unanschaulichen Berechnungsstrukturen und den Visualisierungsstrukturen. Für diesen Übergang ist es notwendig, fehlende Daten auszugleichen. Dazu gibt es verschiedene Verfahren, die mehr oder weniger anspruchsvoll und zeitaufwendig sind. Dabei kann es jedoch leicht zu Verfälschungen der Werte und der gesamten Visualisierung kommen.[93] Schließlich gilt es, Überlegungen zur geeignetsten Form der Visualisierung der Datensätze anzustellen.[94] Dazu stehen neben den drei Dimensionen des Koordinatenraums auch Farben, Farbschattierungen, graphische Elemente und Formen sowie im Falle der Bildanimation die Bewegung als Gestaltungselemente zur Verfügung.

---

[91] Ein Wert t ist ein Datum an der Stelle x,y,z für die Variable Temperatur. Für computerrealisierte Daten kommen noch die Adressierung und der Datentyp als Angaben hinzu.

[92] Die Datenwerte allein sind nicht selbsterklärend und bedürfen entsprechenden Lokalisationen auf Basis unabhängiger Variablen sowie zusätzlicher Beschreibungen, die für alle Daten gelten. Übliche Lokalisationsangaben sind die Zeitkoordinate und die Raumkoordinaten sowie Angaben für welche Variable die Werte stehen. Je höher die Dimensionalität des Datensatzes ist, desto schwieriger wird es, eine strukturierte Form für die Datendarstellung zu erzeugen. Dabei unterscheiden sich Listen von Matrizen dadurch, daß in den Listen numerische Werte als auch ASCII-Text gespeichert sein können und daß die Dimensionalität des Datensatzes nicht eingeschränkt ist. Dagegen bereitet die Darstellung einer 3D-Matrix für eine abhängige Variable Schwierigkeiten und kann nur als eine Abfolge von 2D-Matrix-Layern konzipiert werden. Listen sind deshalb die üblichere Strukturierung, um Daten einer Simulation für die Visualisierung zu speichern. Ein weiterer Aspekt ist die Verteilungsstruktur der Datenwerte in einer Matrix (structured grid) respektive in einer Liste (unstructured grid). Die Dimensionalität von Matrizen legt eine Darstellungsweise in Form eines durch die Koordinatenachsen strukturierten 2D- oder 3D-Gitters nahe, in welchem sich die Datenwerte der abhängigen Variable gleichmäßig oder ungleichmäßig verteilen. Für polygonale Daten, deren Berechnungsgitter komplexer strukturiert ist - wie dies für numerische Simulationen von partiellen Differentialgleichungen der Fall ist – eignen sich Matrizen und die sich daraus ergebenden Verteilungsstrukturen nicht zur Speicherung der Datenwerte. Hier werden unstrukturierte Gitter verwendet, die nur in Form von Listen angeschrieben werden können. Die Verteilungsstruktur der Datenwerte ergibt sich aus der relativen Position zu den benachbarten Werten, wobei der Datenwert in einer Zelle gespeichert ist.[92] Je nachdem aus wievielen Nachbarwerten die Verteilungsstruktur resultiert, ergibt sich die polygonale Form der Zellen. Polygonale Datensets benötigen zwei Listen: eine Liste für die Polygone und eine für die Kontenpunkte P1 ... PN und deren Koordinatenangaben.

[93] Fehlende Werte (Leerstellen) werden durch speziell dafür gekennzeichnete Werte repräsentiert (NaN, 99999, -99 etc.). Die Methode der linearen Interpolation beispielsweise interpoliert zwischen den beiden nähesten Werten einer Leerstelle und setzt die so gewonnen Werte an dieser Stelle ein. Dabei werden jedoch nur die Werte einer Reihe oder einer Spalte berücksichtigt, nicht die gesamte Umgebung der Leerstelle. Verfahren die in kleiner oder größeren Radien die gesamte Umgebung berücksichtigen und eventuell die Nähe oder Weite der beeinflussenden Werte gewichten, liefern differenziertere Ergebnisse. Da bei diesen Verfahren die bekannten Werte angepaßt werden, kann es zu einer verfälschenden Glättung des Werteniveaus kommen. Die aktuell fortschrittlichste und aufwendigste Technik basiert auf dem Kriging-Algorithmus. Vrgl. Fortner, 1995, S. 161ff

[94] Der gesamte Visualisierungsprozeß läßt sich in drei Abschnitte gliedern: Filtering, Mapping, Rendering. Beim Filtering werden die Rohdaten der Simulation durch entsprechende Filterprozesse aufbereitet, das Mapping bildet die Datenwerte auf eine bestimmte Geometrie und ausgwählte Farbwertskalen ab und das Rendering ist die eigentliche Bilderstellung. „*Scientific visualization is an amalgam of tools and techniques that seeks to promote new dimensions of insight into problem-solving using current technology*." Earnschaw, R.A./Wieseman, N.: An Introductory Guide of Scientific Visualization, 1992, S. 5

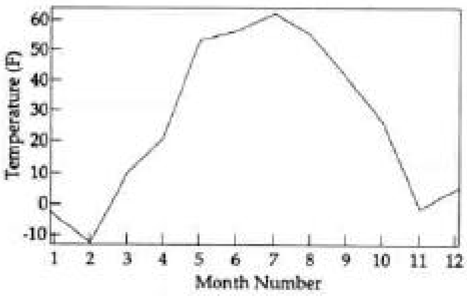

Abb. 9: Liniengraphik einer 1D-Liste

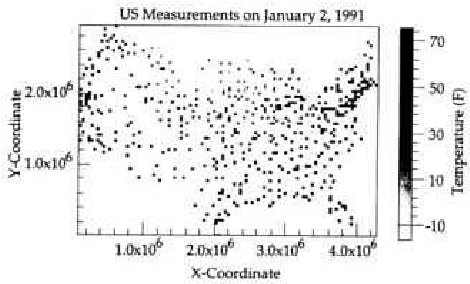

Abb. 10: Streugraphik einer 2D-Liste mit Grauwerteskala

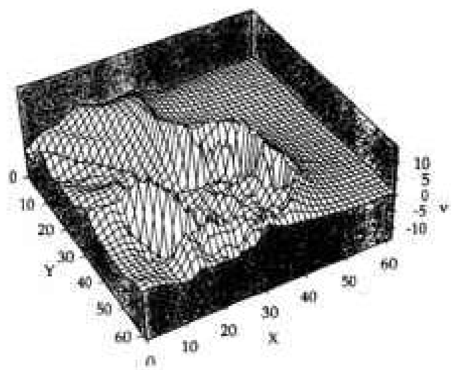

Abb. 11: 3D Oberflächendarstellung eines 2D Datensatzes

Abb. 12: Konturendarstellung einer 2D-Matrix

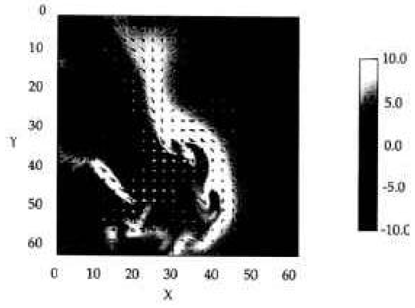

Abb. 13: Vektordarstellung eines 2D-Geschwindigkeitsfeldes

Die durchschnittliche Temperatur eines Monats läßt sich mit diskreten Punkten in funktionaler Abhängigkeit von den Zeiträumen darstellen. Die graphische Verbindung der Punkte simuliert eine Kurve (Abb. 9). Die räumliche Verteilung der Temperaturwerte zu einem einzigen Zeitpunkt bedarf bereits der Farbnuancierung (Grauwerte) als weiterere Dimension zur Darstellung der Temperaturwerte (Abb. 10). Das Niveau verschiedener Datenwerte kann aber auch als dritte Dimension in eine 3D-Darstellung integriert werden (Abb. 11) oder als Verbindungskontur zwischen gleichen Werten (Abb. 12). Um die Windgeschwindigkeit sowohl in ihrer Stärke als auch in ihrer Richtung darzustellen, eignen sich Vektordarstellungen. Dabei hängt der Vektor – graphisch symbolisiert als Pfeil - von den 2D-Datenwerten für *x* und *y* ab. Die Stärke ergibt sich aus der Berechnung der *Wurzel von* $(x^2 + y^2)$ und die Richtung aus $\tan^{-1} (y/x)$ (Abb. 13). Es sind vielfältige graphische Elemente zur Veranschaulichung von Entwicklungen in den Daten möglich, so etwa Spurbänder, welche die Luftströmung in einer Wolke verdeutlichen. Die bislang gezeigten Beispiele bewegten sich im zweidimensionalen Bereich, mit oder ohne zusätzliche Informationen über Farbskalen oder graphischen Elementen. Die in Abbildung 11 auftretende dritte Dimension ergibt sich aus der Verwendung der Niveaus der Datenwerte als Auftrag in der z-Koordinate.

Tatsächliche dreidimensionale Darstellungen sind in ihrer Visualisierung sehr aufwendig und basieren auf einer Fülle von Datenwerten, die in der Regel in einer 3D-Matrix mit einem uniformen Gitter gespeichert sind.[95] Für die 3D-Visualisierung gibt es verschiedene Verfahren, die entweder einzelne Bilder als Ebenen animiert zusammenfügen (slicing und dicing), die Oberfläche mit Konturlinien in 3D erzeugen (isosurfaces) oder das gesamte 3D-Objekt darstellen (volumetric visualization). Für die volumetrische Darstellung werden die Datenwerte in Intensitätswerte transformiert, die aufeinander angeordnet addiert werden. Ähnlich einer Wolke zeigt sich große Intensität in Opakheit und niedrige Intensität in Transparenz. „*Volumetric visualization mimics nature by creating a `cloud´ of your data*."[96] Schattierungen und Lichtsetzungen unterstützen die piktorale Wirkung. Dabei wird, ausgehend vom Auge des Betrachters vor dem Bildschirm, der Weg der Lichtstrahlen berechnet, die entsprechende Brechungen und Reflexionen auf der Oberfläche der visualisierten Objekte erfahren. Berechnet man die dreidimensionale Darstellung stereometrisch und betrachtet die Objekte mit entsprechendem Equipment (3D-Brillen), so erhält man einen realistisch anmutenden Eindruck eines Objekts, beispielsweise eines Moleküls, das man beliebig in jede Richtung drehen, von jeder Distanz aus betrachten und eventuell gar darin eintauchen kann. Undurchdringlichkeit ist - falls nicht als collison detection programmiert - keine Eigenschaft elektronisch realisierter Objekte.

[95] „*A small volumetric dataset of dimension 100 x 100 x 100 contains one million data values! The disk file size would be two megabytes in size for short integer values, four megabytes for floating-point, and approximately 12 megabytes for ASCII text*." Fortner, 1995, S. 131/132

[96] Fortner, 1995, S. 140. Eine weitere 3D-Technik sind Vektorfelder und Strömungslinien im Dreidimensionalen. „*Streamlines are similar to throwing smoke bombs in a windstrom. A windstrom defines a 3D vector field of wind velocity ... The smoke bomb traces the path in that 3D vector field* ..." Fortner, 1995, S. 141

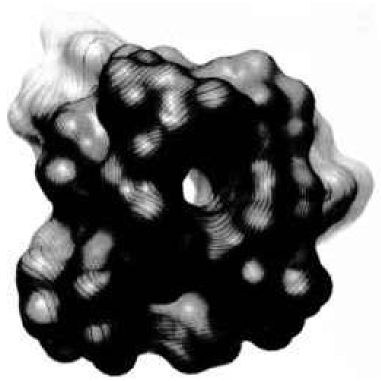

Abb. 14: Isosurface-Darstellung

4.2 Farbdifferenzierung und Gestalt

Die Farbdifferenzierung spielt in der Visualisierung von Datenwerten eine bedeutende Rolle, da sie eine weitere Darstellungsdimension eröffnet. Dabei können Grauwerteskalen oder Farbskalen verwendet werden. Jeder Farbwert wird in drei Komponenten zerlegt und zwar in einen numerischen Wert zwischen 0 und 255 für rot, grün und blau. Dunkelrot ist beispielsweise als (128, 0, 0), schwarz als (0, 0, 0) und weiß als (255, 255, 255) numerisch kodiert. Die Grauwerteskala (Pseudofarbskala) besitzt 255 verschiedene Abstufungen und weist dem Rot-, Grün- und Blau-Index jeweils denselben Wert zu. Indem die quantitative Spanne zwischen den Datenwerten in 255 Stufen eingeteilt wird, erhält man eine Ordnung, die sich auf die Grauwerteskala übertragen läßt. Aufgrund der farblichen Eindimensionalität besitzen die Grauwerte eine deutliche Aussagekraft gegenüber der Quantität der numerischen Werte. So ist entscheidbar, ob ein Grauwert heller oder dunkler ist und ob er entsprechend einen niedrigeren oder höheren numerischen Wert repräsentiert. Für das Farbspektrum ist diese einfache Zuordnungsweise aussagelos, denn es gibt keine aufsteigende Ordnung zwischen den verschiedenen Farben in der Wahrnehmung, selbst wenn eine numerische Ordnung der Farbwerte von (0, 0, 0) bis (255, 255, 255) denkbar ist. Die Zuordnung geschieht üblicherweise im Rahmen einer expliziten Farbskala (z.B. Regenbogenskala), um quantitative Aussagen mit Farben zu verknüpfen. So lassen sich beispielsweise niedrige Werte blau und hohe Werte rot darstellen. Dazwischen ergibt sich ein Farbverteilungsspektrum unterschiedlicher Farben gemäß der Farbskala, die der Visualisierung als Beschreibungskomponente beigestellt werden muß, soll eine Interpretation zu adäquaten Aussagen über die repräsentierten numerischen Werte gelangen.

Die Visualisierung der Datenwerte kann zeichenbasiert, graphisch oder ikonisch erfolgen. Der Unterschied zwischen einer graphischen und einer ikonischen Darstellung ist einsichtig, aber schwer beschreibbar.[97] Graphiken organisieren Farbe in voneinander abgegrenzten Formen mit einfachen Elementen wie Punkten, Linien oder Flächen. Die Form ist der dominante Faktor, nicht die Farbe, weshalb die meisten Graphiken mit wenigen oder nur einer Farbe auskommen. Zudem sind die Formen in einheitlichen Farben gehalten. Sie basieren nicht auf Farbschattierungen und wirken deshalb abstrakt.

[97] Nelson Goodman billigt Diagrammen ein pikturales Aussehen zu, verweist aber darauf, daß Diagramme meist digital oder gemischt, also digital und analog sind. Für ihn ist die Bezeichnung graphisch mit analog verbunden: „... *dann ist das Diagramm rein analog beziehungsweise graphisch*.“ Goodman, 1995, S. 163

Dagegen weisen ikonische Visualisierungen sowohl unterschiedliche Farben als auch Farbschattierungen über die gesamte Fläche der Darstellung auf. Formen entstehen durch Abgrenzungen zwischen unterschiedlichen Farben und durch Farbschattierungen. Aufgrund ihrer Farbdichte und der sich dadurch ergebenden Geschlossenheit der Fläche erhält diese Form der Visualisierung ikonische Eigenschaften. Als Unterscheidungskriterium zwischen graphischen und ikonischen Darstellungsformen soll hier die Identität von Farbe und Form gelten, d.h. insofern eine Farbe ohne Farbschattierungen sich klar von anderen Farben abhebt und somit identisch mit der Form ist, die sie anzeigt, soll von einer Graphik die Rede sein. Farbformen dieser Art sind deutlich voneinander abgegrenzt und erscheinen als einfarbige Elemente oder Flächen. Ikonische Darstellungsformen hingegen erhalten aufgrund der Farbschattierungen eine weitere Dimension zur Darstellung von Informationen, und es können Muster zu Tage treten, die in den einfachen Farbformen der Graphiken nicht ersichtlich sind. Durch die Farbschattierungen treten die Formen weniger deutlich voneinander abgegrenzt auf und es kommt ein plastisches Element zur Geltung, das sich auch in Grauwerteverteilungen zeigt. Beide sind jedoch pikturale Darstellungsformen.

Dies wirft die Frage auf, woher die Formen ihre Gestalt beziehen: aus den Datenwerten oder durch gestaltgebende Programmierungen? Für den Verlauf einer Kurve ergibt sich die Gestalt der Kurve aus dem Werteverlauf einer Funktion, d.h. die Gestalt wird durch die Datenwerte generiert. Insofern es sich um eine stetige Funktion handelt, ist die Verbindung der berechneten, diskreten Werte mit einer durchgängigen Linie legitim. Dabei erfolgt die Legitimierung aufgrund mathematischer Eigenschaften der zugrundeliegenden Funktion (Nachweis der Stetigkeit). Handelt es sich jedoch um diskrete Durchschnittswerte wie in Abbildung 9, so stellt die Verbindung der Werte mit einer Linie eine gestaltgebende Geste dar und suggeriert einen Kurvenverlauf auf der Basis gemittelter Werte. Unabhängig davon bietet die Farbe der Linie keine zusätzliche Information und ist mit der Form identisch bzw. sie konstituiert die Form. Die in Abbildung 10 erscheinende Form des nordamerikanischen Kontinents ergibt sich aus der räumlichen Verteilung der Meßinstrumente und nicht aus den Datenwerten. Die Grauschattierungen präsentieren die relevanten Informationen der Temperaturunterschiede. Da die jeweiligen Datenwerte in abgeschlossenen Formen (Punkten) dargestellt sind, die jeweils einer Grauschattierung entsprechen, sind diese mit der Form identisch, und es handelt sich um eine graphische Visualisierung. Die Temperaturunterschiede können auch ohne Farbschattierungen als Auftrag auf der z-Achse zum Ausdruck kommen (Abb. 11). Die Form der Konturen in Abbildung 12 ergeben sich aus einer spezifischen Strukturierung der räumlich verteilten Datenwerte gemäß ihren Werten mit Hilfe einer schwarzen Linie. Die komplexe Linienführung und die Häufung von Linien machen es an manchen Stellen unmöglich, die Linien voneinander zu unterscheiden. Dadurch entstehen scheinbar dunklere Gebiete. Die Dichte der grauschattierten Stellen in den Abbildungen 10 bis 12 lassen diese Abbildungen im Unterschied zur Kurvendarstellung pikturaler erscheinen und zeigen den Übergang von einer graphischen zu einer ikonischen Darstellung. Ein Grund dafür ist die Plastizität, die sich aus der Grauwerteschattierung, der dreidimensionalen Darstellung und der Verstärkung der Linien zu dunkleren Gebieten ergibt. Deutlich zeigt sich dieser Effekt der Plastizität in Abbildung 14 für die Isosurface-Darstellung eines Moleküls. Die Gestalt von Objekten und ihrer Umgebung resultiert aus der Strukturierung der Datenwerte. Insofern es sich um eine Simulation im Raum handelt, ist das Berechnungsgitter mit der Struktur der Datenlokalisation identisch (2D: x-, y-Koordinatenachse, 3D: x-, y-, z-Koordinatenachse). Um eine Wechselwirkung zwischen einem Objekt und seiner Umgebung zu erzeugen, werden verschiedene Verfahren verwendet. Für die Crashsimulation beispielsweise müssen die Verformungskräfte, die auf das Objekt wirken, für jeden Zeitschritt berechnet werden (Abb. 16).[98] Anders verhält es sich in Abbildung 17.[99]

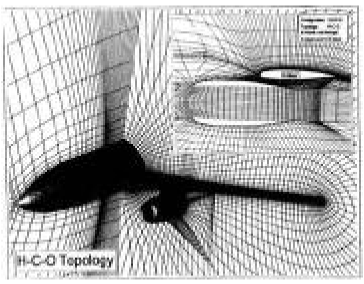

Abb. 15: Strukturiertes Gitter für die Umgebung eines Flugzeuges

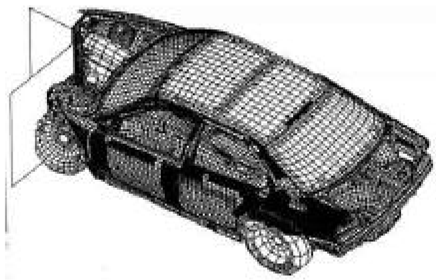

Abb. 16: Finite Elemente-Struktur eines Autos für eine Crashsimulation

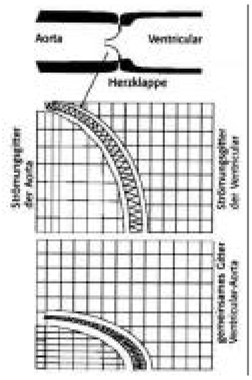

Abb.17: Kopplung zwischen Gitter und Finiten Elementen

[98] „*Ein Ziel bei der Crashsimulation ist es, die Verformung des Autos zu bestimmen. Hierzu wird bei der Anwendung der Finite Elemente-Methode das Auto in viele Elemente aufgeteilt ... Die erstmalige Entwicklung der Gitterstruktur von bis zu 200 000 Elementen dauert in der Regel mehrere Wochen. Die verschiedenen Lösungsschritte, wie die Berechnung der Kräfte oder Spannungen, werden dann für jedes einzelne Element durchgeführt. Bei einem Crashvorgang, der etwa 80 Millisekunden dauert, wird zum Beispiel die Verformung des Autos in über 80 000 Zeitschritten aus dem jeweils vorherigen Zeitschritt explizit berechnet*." Galbas, H.G. et al.: Schnelle parallele Kontaktverfahren zur Crashsimulation, 1997, S. 44

[99] Im Falle der Simulation des dynamischen Verhaltens einer Herzklappe in der Blutströmung wird eine gekoppelte Berechnung der Blutströmung und der sich darin bewegenden Herzklappe vorgenommen. Dabei werden die Herzklappe als auch die Umgebung durch unterschiedliche Gitter dargestellt. Die Wechselwirkung wird durch Nachbarschaftsbeziehungen zwischen den Gitterelementen modelliert (Abb. 17). „*In unserem Beispiel der Herzklappe würde die Blutströmung durch ein strukturiertes oder unstrukturiertes (regelmäßiges) Hexaeder-Gitter dargestellt, die Herzklappe jedoch durch Finite-Elemente in Dreiecksform nachempfunden ... Für die Untersuchung der Interaktion zwischen Struktur und Strömung muß die Strukturverformung der Herzklappe direkt aus dem aktuellen Strömungsdruck abgeleitet werden: die Druckinformation in dem Gitter muß in einen Kraftkoeffizienten in der Dreiecksstruktur umgesetzt sein ... Dazu müssen die Nachbarschaftsbeziehungen der Strukturen festgestellt werden: welcher Gitterpunkt ist mit welchem Dreieckspunkt direkt benachbart und muß mit diesem Daten austauschen*." Brakkee, E. et al.: Herzklappen, Drehmomentwandler, Fährschiffe und andere Probleme, 1997, S. 31

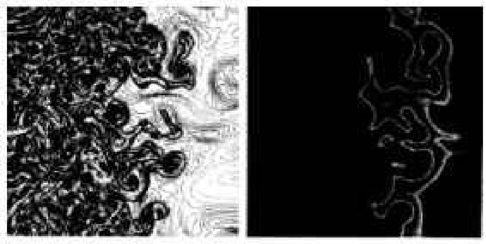

Abb. 18: Strömung in einer Methanflamme

Doch es gibt Gestalten, die sich ausschließlich aus den Datenstrukturen ergeben, wie beispielsweise die Wirbelbildungen in der Strömungsdynamik. Sie basieren auf den farblich visualisierten Datenwerten in einem gleichförmig strukturierten Gitter. So zeigt Abbildung 18 die numerische Simulation der Strömung in einer turbulenten Methanflamme (links: Strömungsfeld, rechts: relativer Massenanteil des Sauerstoff-Wasser-Radikals). Diese Formen sind keine vorgegebenen Effekte, sondern resultieren aus dem dynamischen Verhalten der Datenwerte. Visualisierungen dieser Art sind abstrakte Bilder auf Basis reiner Farben, welche die Formen konstituieren, wie die farbigen Abbildungen auf Seite 86 demonstrieren. Die entfalteten und visuell präsentierten Datenstrukturen verweisen auf Zusammenhänge, wie sie mit den Gleichungen beschrieben und durch die Berechnungen aktualisiert werden. Die Visualisierungen erzeugen je nach Lösungsverhalten mehr oder weniger stark strukturierte Bilder. Aussagen über das Verhalten resultieren aus der Identifizierung gestaltbildender Elemente, die sich durch visuell wahrnehmbare Zusammenhänge, Abgrenzungen und Gliederungen ergeben.

Die Gestalt bietet dabei einen wichtigen Anhaltspunkt zur Interpretation des strukturellen Lösungsverhaltens sowie für den Vergleich mit dem physikalischen System, das als extrasymbolische Interpretation der Simulation zugrunde liegt. Gestalt entsteht, wenn sich eine Gestalt-Hintergrund-Differenzierung ergibt. Dies ist dann der Fall, wenn eine geschlossene Struktur die Teilelemente in ein übergeordnetes Ganzes integriert und sich vom Hintergrund abhebt, obwohl die Gesamtheit der Teilelemente nicht unbedingt mit dieser Gestalt identisch sein muß.[100] Die Gestalterkennung ist von daher ein interpretativer Wahrnehmungsprozeß. „*Wir nennen Gestalt die Form eines Gebildes, wenn diese nicht der Starrheit des Materials zu verdanken ist und auf einer Festlegung jedes einzelnen Punktes für sich, sondern auf einem Gleichgewicht von Kräften (Spannungen usw.) beruht*."[101] Diese Kräfte ergeben sich aus den Gestaltqualitäten visueller Elemente wie beispielsweise ihrer Gleichartigkeit, ihrer Nähe zueinander, ihrer Geschlossenheit und Symmetrie. Vor allem die symmetrisch angeordneten Elemente führen zu prägnanten Gestalten. Die Gestalten ergeben sich aus berechneten Größen, deren Quantität mit Hilfe der Farbgebung in eine qualitative Ausdrucksweise transformiert wurden. So nehmen die Gestalten – wie in der Strömungsdynamik - die Form von laminaren Strömungen, Wellen

[100] Beispielsweise im sog. Bourdon-Effekt: Drei Punkte in einer spezifischen Anordnung ergeben die Gestalt eines Dreiecks und vier Punkte eines Quadrats. Die Gestalt entsteht durch imaginäre Brückenlinien zwischen den Punkten.

[101] Metzger, W.: Gestalt-Psychologie, 1986, S. 130. „*In jedem Fall aber handelt es sich um ganze, das heißt um überpunktuelle Gebilde oder Sachverhalte, die räumlich, zeitlich oder raumzeitlich ausgedehnt sind, mit Eigenschaften, die sich nicht aus artgleichen Eigenschaften der punktuellen Elemente herleiten lassen. Diese Eigenschaften nannte v. EHRENFELS „Gestaltqualitäten"* ... " Metzger, 1986, S. 125

und Wirbeln an bzw. werden entsprechend als laminare Strömungen, Wellen und Wirbel interpretiert. Zudem spielt die Wahl der Farbskala eine bedeutende Rolle, denn gewisse Farben sind mit bestimmten Interpretationen verbunden: so werden rote Bereiche eher als kritisch angesehen (z.B. hohe Druckwerte) und hellere als weniger intensiv als dunklere. Auch die Farbfolge im Farbspektrum ist nicht ohne Einfluß. Eine Regenbogenskala erstreckt sich zwischen violett und rot, doch die Wahrnehmung ordnet beide Farben graduell zueinander und gibt die Unterschiede zwischen den Werten unadäquat wieder. Auch die Dynamik der Datenstrukturen ergibt eine Gestalt in der Zeit, insofern die Übergänge von Bild zu Bild nicht zu sprunghaft sind.[102]

4.3 Animierte Bildobjekte

Die Sichtbarmachung von Prozessen ist der eigentliche Vorteil der Visualisierung numerischer Simulationen, da sie diese intuitiv in der Zeit erfaßbar macht, d.h. die Veränderungen werden als Zeitphänomen interpretiert und nicht als ein symbolisch kodierter Wechsel numerischer Werte. Die Zeitschritte der Simulation geben die Taktfrequenz des Bildwechsels vor, und die Veränderung zwischen den einzelnen Bildern generiert sich durch die Dynamik der Datenstrukturen. Mit der ikonischen Visualisierung nutzt die Simulation die mediale Freiheit der Präsentation digitaler Zeichen und verläßt die diskrete Darstellungsweise. Die Transformation der Lösungswerte in Farbwerte macht die Strukturen, die sich in der Veränderung der numerischen Werte entfalten, als Gestalt in der Zeit sichtbar und ermöglicht so Aussagen über das Lösungsverhalten der Gleichung unter spezifischen Bedingungen. Da mit diesen Gleichungen naturwissenschaftlich interessante Systemprozesse mathematisch modelliert werden, erlauben die Visualisierungen darüber hinaus einen anschaulichen Vergleich mit den beobachteten Systemen bzw. visualisierten Prozesse, die nicht beobachtet werden können. Dabei transferieren die Visualisierungen Anschauungskonzepte unserer mesoskopischen Erfahrungsebene auf an sich unsichtbare Bereiche. Beispielsweise, indem funktionalisierte und quantifizierte Wirkungen zwischen Elektronen als Isosurfaces dargestellt werden und so den Eindruck eines Atoms als solides Objekt vermitteln (Abb.14). Mit Hilfe innovativer Visualisierungstechniken, wie Stereo-3D-Darstellungen und taktile Simulationen werden diese Bilder in der Wahrnehmung zu raumfüllenden, beweglichen und tastbaren Objekten. Die Interferenz von Wahrnehmungsraum und Realraum wird dabei aufgehoben. Dieser Bruch kennzeichnet die Virtualität simulierter Objekte.

[102] Dieses Phänomen wurde in der Gestalttheorie unter dem Begriff Scheinbewegung untersucht. Vrgl. Köhler, W.: Die Aufgabe der Gestaltpsychologie, 1971. Das Gehirn nimmt eine Folge von ca. 24 Bildern pro Sekunde als kontinuierliche Bewegung wahr, insofern die Bilder einen nicht zu sprunghaften Ablauf zeigen. Eine weiteres Gestaltphänomen in der Zeit wäre eine Melodie.

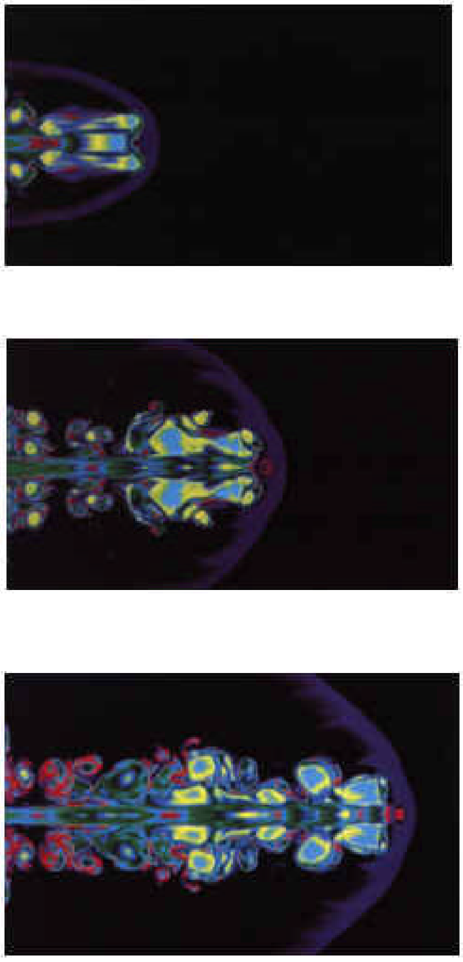

Abb. 19: Simulation von Stoßwellen für Jets, die mit Überschallgeschwindigkeit fliegen, auf Basis der Euler-Gleichungen

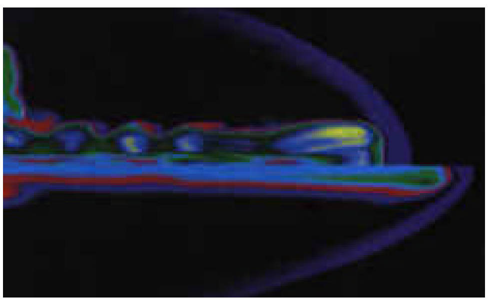

Abb. 20: Vergleich zweier Simulationsgänge mit feinem Berechnungsgitter (obere Bildhälfte) und gröberem Gitter (untere Hälfte)

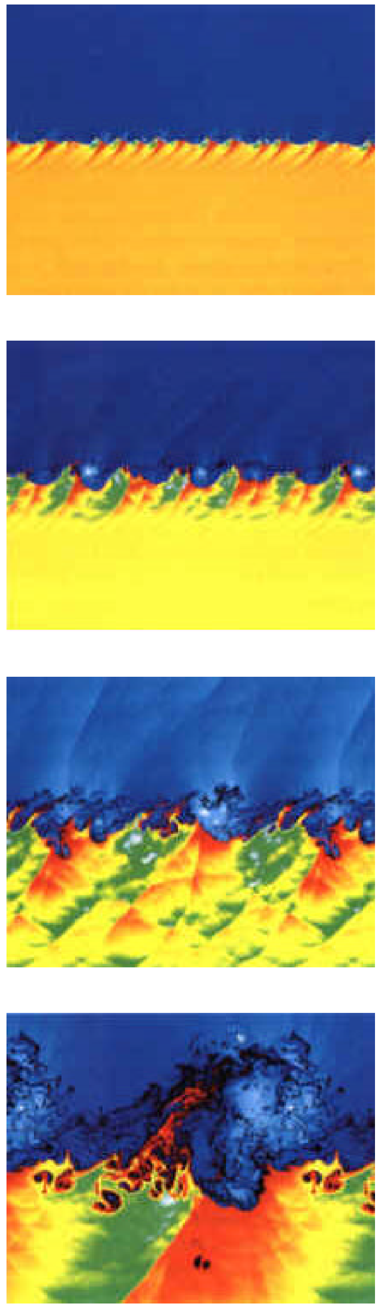

Abb. 21: Simulation der Kelvin-Helmholtz-Instabilitäten für zwei Flüssigkeiten, die sich mit Schallgeschwindigkeit aneinander vorbei bewegen, auf Basis der Euler-Gleichungen (hohe Dichte = grün, niedrige Dichte = blau)

5. Von der Berechenbarkeit zur Simulierbarkeit

Um einen Zusammenhang berechenbar zu machen, muß dieser zum einen in einer formalen Struktur artikulierbar, zum anderen quantifizierbar sein, d.h. auf zähläquivalenten Zeichen basieren, welche numerische Werte notieren. Numerische Werte sind durch eine Ordnung charakterisiert, die anhand eines Kalküls rekursiv erzeugbar ist (vollständige Induktion von **N**) oder auf entsprechend anderen Konstruktionsverfahren aufbaut.[103] Die Berechnung einer Lösung kann einerseits allgemein in der deduktiven Umformulierung von Folgen von Variablen- und Operationszeichen bestehen und als formale Struktur angeschrieben sein, oder sie kann sich konkret in der Anwendung der mit der formalen Struktur symbolisierten Operationen auf numerische Werte vollziehen, welche für die Variablen eingesetzt werden und einem zuvor definierten Ordnungsmuster entstammen. Die numerische Lösung ist dann der aus den Operationen resultierende numerische Wert der formalen Struktur. Dieses Vorgehen läßt sich funktional als eine Zuordnung zwischen den numerischen Werten des Definitionsbereichs und dem Wertebereich gemäß der in der formalen Struktur symbolisierten Operation darstellen: $f: D \rightarrow W$. Die Funktionsvorschrift drückt den entsprechenden Zusammenhang zwischen den Zeichenbereichen aus und faßt diesen im Werteverlauf - visualisiert als Graph - zusammen, der sich aus den jeweiligen Zeichenpaaren ergibt. Funktionen sind semiotische Maschinen zur Erzeugung von Werten. Die Entwicklung des Werteverlaufs respektive des Graphen in einem Koordinatensystem, bezogen auf die voranschreitende Erzeugung der numerischen Werte des Definitionsbereichs und entsprechend der Funktionsvorschrift des Wertebereichs, läßt sich mit der Ableitung der Funktion darstellen. Differentialgleichungen artikulieren eine Beziehung zwischen Funktionen und ihren Ableitungen. Sie sind mathematische Instrumente zur Darstellung der Veränderung veränderlicher Größen und finden deshalb in naturwissenschaftlichen Bereichen Anwendung, die quantifizierbare Prozesse zum Gegenstand ihrer Forschung haben. Differentialgleichungen sind die mathematische Basis deterministischer Simulationen (*dy/dt = Gleichung*), und ihre Lösung ist eine stetige Funktion *y(t)*. Diese exakte Lösungsfunktion ergibt sich aus den Umformungen der Differentialgleichung gemäß festgelegter Regeln.[104] Doch diese formal-analytische Handhabung ist nur auf einen kleinen Bereich von Differentialgleichungen beschränkt.[105] Für alle anderen Differentialgleichungen sind bislang keine Lösungsfunktionen bekannt. Die numerische Simulation bietet nun eine neue Methode, diskretisierte Versionen der Differentialgleichungen zu erstellen und approximativ zu lösen, d.h. für endlich viele Berech-

[103] Das können neben Zahlen auch binärlogische Werte sein, die sich mit 0/1 darstellen lassen. Lediglich die komplexen Zahlen besitzen keine Ordnungsstruktur im Sinne von größer, gleich, kleiner.
[104] Die Lösungsfunktion gilt für alle numerischen Werte, welche für die Variablen eingesetzt werden.
[105] Lösungsfunktionen sind für gewöhnliche, viele lineare und einige partielle nichtlineare Differentialgleichungen bekannt.

nungspunkte, so daß die Ergebnisse der exakten Lösung beliebig nahe kommen.[106] Bei dieser Transformation formaler Strukturen in algorithmisierte Zeichenoperationen tritt der prinzipielle Unterschied zwischen dem Symbolisierungsgehalt schriftbasierter Zeichen und computerbasierten Zeichenoperationen zu Tage. Denn während schriftbasierte Zeichen auch infinitesimale Operationen symbolisieren und sich für den Umgang mit diesen Zeichen Regeln aufstellen lassen, es also einen Formalismus zur Handhabung der Symbole infinitesimaler Operationen auf dem Papier gibt,[107] müssen die Symbolisierungen – sollen sie in das maschinelle Medium des Computers übertragen werden – in konkrete Operationen umgesetzt werden. D.h. sie müssen effektiv auf endlichen und eindeutigen, zur Not gerundeten numerischen Werten, die Maschinenzustände notieren, operabel sein. Die Operationen werden direkt auf den numerischen Werten ausgeführt und nicht gemäß einem Formalismus umgeformt. Dies ist ein wichtiger Unterschied zwischen dem formal-operativen und dem maschinellen Zeichenumgang. Kennzeichnend dafür ist, daß im Verlauf formaler Umformungen die Variablenzeichen sich zwar in ihren konfigurierenden Kombinationen ändern, an sich aber als Entitäten erhalten bleiben. D.h. die Variablen werden von einem strukturellen Zusammenhang regelbasiert in einen anderen transformiert.

Berechnen in diesem Sinne meint die deduktive Umformung von Gleichungen in allgemeine Lösungen, welche eine formale Struktur für die gesuchte Unbekannte erzeugt. *Berechnen* im numerischen Sinne geht auf dem Papier von der Einsetzung numerischer Werte in die formale Struktur der Lösung aus, um so numerische Werte der gesuchten Unbekannten zu erzeugen. Diese Verarbeitungsweise ist der numerischen Simulation verschlossen, da in der Regel die Umformungsschritte zur algebraischen Lösung einer komplexeren Differentialgleichung nicht bekannt sind. Deshalb wird der Umformungsprozeß an den Computer delegiert und zwar derart, daß ein Algorithmus Verarbeitungsanweisungen zur Manipulation numerischer Werte auf Basis der Gleichungen – nicht der Lösungen, die ja unbekannt sind - vorgibt. Der Computer löst durch Umformungen eingesetzter numerischer Werte für die Parameter, Anfangs- und Randbedingungen ein Gewebe aus Gleichungen, die in Abhängigkeit voneinander jeweils für eine Unbekannte zu einem Berechnungspunkt auf einem Zeitschrittniveau berechnet werden und eine Kaskade numerischer Werte erzeugen. Die Werte stellen die Näherungs-

[106] Wie auf Seite 72/73 gezeigt, ist die beliebige Annäherung, wenn auch theoretisch gefordert, praktisch nicht vollziehbar.
[107] Mittlerweile hat sich auch der Bereich der Computeralgebra etabliert, in welchem Algorithmen zur Umformung von Gleichungen entwickelt werden. Für die formale Bearbeitung partieller Differentialgleichungen sind die Ergebnisse bislang jedoch mager und lediglich für gewöhnliche Differentialgleichungen sind solche allgemeinen Lösungsalgorithmen bekannt (Kovacic-Singer-Ulmer-Algorithmus) „*Ein klassisches Problem der angewandten Mathematik von großer Bedeutung liegt in der Berechnung der allgemeinen Lösung einer Differentialgleichung. Mit dem Aufkommen der Computeralgebra erfuhr diese Aufgabe noch eine Verschärfung: Die Konstruktion der Lösung soll algorithmisch erfolgen. Diese Forderung erweist sich jedoch als zu stark, so daß man sich mit geringeren Zielen zufrieden geben muß*.“ Seiler, W.: Formale Theorie partieller Differentialgleichungen, 1991, S. 318. „*Obwohl zur Zeit also nur ein echter Lösungsalgorithmus für Differentialgleichungen existiert, kennt die Computeralgebra zahlreiche Verfahren zu ihrer Behandlung. Die wenigsten davon sind algorithmisch, viele hängen von Heuristiken ab. Die angestrebten Ziele können sehr unterschiedlich sein. Zahlreiche Ansätze streben eine Vereinfachung des Ausgangsproblems an: nichtlineare Gleichungen sollen in lineare, partielle in gewöhnliche überführt werden; gekoppelte Gleichungen sollen entkoppelt werden, die Ordnung der Gleichungen soll reduziert werden*.“ Seiler 1991, S. 321

lösung der als existent angenommenen exakten Lösung innerhalb eines definierten Raum-Zeit-Gitters dar und ihr Verhalten im Raum sowie in der Zeit erlaubt qualitative Aussagen über die erzielten Ergebnisse. Die Anweisungen für die numerischen Umformungen basieren zum Teil jedoch auf heuristischen Annahmen, d.h. der Umformungsprozeß ist nicht komplett analytisch herleitbar. Er gründet auf Plausibilitäten, die sich aus den Erfahrungen mit analytisch lösbaren Problemen ergeben. Der Übergang von der Berechenbarkeit zur Simulierbarkeit vollzieht sich also mit der Transformation einer deduktiven in eine heuristische Vorgehensweise auf Basis von Plausibilitäten bezüglich der Umformungsschritte und quasi-empirischen Auswertung der Resultate. Zur Beurteilung der Resultate dient die Präsentation des Lösungsverhaltens in Form von Zahlzeichen oder Farbwerten. Das bedeutet, es muß so lange von Fall zu Fall getestet werden, ob der Lösungsalgorithmus gut oder schlecht ist, bis für ausreichend viele Fälle die Lösungen adäquat erscheinen.

Diese quasi-empirische Methode der numerischen Simulation partieller Differentialgleichungen bedarf einer Masse von Datenwerten und wird durch die steigenden Leistungskapazitäten der Computer ermöglicht. Die Geschwindigkeit der Rechenoperationen auf Maschinenebene – aktuell rund 100 Milliarden Operationen pro Sekunde – wird durch die Zerlegung der Zeichen in binärkodierte Zustände möglich und stellt einen quantitativen Vorteil gegenüber dem schriftbasierten Zeichenumgang dar. Dieser quantitative Vorteil wandelt sich jedoch in einen qualitativen, insofern zum einen eine neue Methode zur mathematischen Behandlung komplexer Gleichungssysteme möglich wird, die sich einem deduktiven Zeichenumgang in einem formal-operativen Zeichensystem entziehen, und indem zum anderen die Visualisierung die Möglichkeit eröffnet, erstmals ikonisch basierte Einsichten in semiotisch modellierte Prozesse zu erhalten.[108] Damit wird nicht nur die Beschreibung oder Berechnung von Prozessen möglich, sondern deren semiotische Modellierung auf Basis numerischer Werte, die in Form von Maschinenzuständen direkt manipulierbar werden.[109] Der Weg von der Berechenbarkeit zur Simulierbarkeit setzt die Mechanisierung der Zeichenverwendung voraus. Mit der Mechanisierung ist die Zerlegung schriftbasierter Zeichen in digitale Zeichen - diskrete, binärkodierte Zustände - gemeint, die sich unanschaulich im maschinellen Medium des Computers gemäß programmgesteuerter Instruktionen vollziehen und die in ihrer sinnlich wahrnehmbaren Konstitution frei präsentierbar sind, also nicht mehr nur als Schriftzeichen ausgegeben werden. Die Überwindung der symbolischen Nutzung von Zeichen, einerseits durch die tatsächliche Ausführung der Operationen[110] – also nicht nur ihre

[108] Dabei muß es sich nicht nur um numerische Simulationen partieller Differentialgleichungen handeln, sondern es können auch quantenmechanische oder statistische Simulationen sein.
[109] Die zeitliche Dimension ergibt sich aus dem Vergleich mit dem mathematisch modellierten, physikalischen Prozeß. In der Mathematik selbst spielt die Zeit keine intuitive Rolle, sie ist lediglich eine Variable.
[110] Zwar gilt das auch für das Berechnen auf dem Papier, doch nur im eingeschränkten Maße, denn der Umfang der Berechnungen als auch deren Komplexität hat sich durch die Mechanisierung der Zeichen und die Algorithmisierung des Berechnens

formal-symbolische Umformung -, andererseits durch die ikonische Darstellungsweise der Resultate, eröffnet der numerischen Simulation eine neue Umgangsweise mit den Zeichen. Die Zeichen selbst sind das Material zur semiotischen Modellierung von Prozessen auf Basis formaler Operationen, und sie sind das Material für die Konstruktion semiotischer (Daten-)Objekte. Dabei muß die Generierung der Datenwerte so umfangreich sein, daß die in den Daten durch die Operationen definierten Strukturen entfaltet werden und daß für die Visualisierung ein syntaktisch dichtes Symbolschema ausgebildet werden kann, damit der ikonische Effekt wie die Dynamik der Datenstrukturen zu Tage tritt. Die numerische Simulation bietet aus dreierlei Gründen neue Einsichten: durch die semiotische Ausführung dynamischer Prozesse, die bislang nur schriftlich fixiert und von daher statisch symbolisiert werden konnten; durch die Erweiterung des Berechnungsraums, so daß komplexere Strukturen mathematisch behandelbar werden; und durch die ikonische Darstellung der Resultate, welche die Dynamik intuitiv zugänglich präsentiert. Der semiotisch interessante Aspekt besteht dabei in der Überwindung eines symbolischen Zeichenumgangs durch die Transformation zeichenbasierter Informationen auf maschinelle Operationen. Für das Simulationsgeschehen selbst handelt es sich um einen nicht-repräsentationalen Zeichenumgang, der freilich als Referenz an unsere Anschauung und unsere Zwecke strukturiert, visualisiert und interpretiert werden muß. Die Entkopplung von extra- und intrasymbolischen Bedeutungen geht dabei über das Maß der Formalisierung hinaus, da mit der Mechanisierung ein anderer Zeichenumgang gefordert ist, der zugleich in einer darstellenden (instruierenden) und einer exekutiven Funktion der regelbasierten Anweisungen besteht. Dies bedeutet, daß der intrasymbolische Gehalt der Zeichen auf der Ebene der Zeichenmanipulation nicht symbolisch angezeigt wird, sondern direkt als Operation ausgeführt werden muß.

Um die Mechanisierung tatsächlich maschinell umzusetzen, bedarf es einer weiteren Zerlegung der Zeichen durch die Einführung einer subsymbolischen Ebene. Das bedeutet, daß alle intrasymbolisch relevanten Aspekte wie die Funktion der Zeichen im Zeichensystem, ihre Unterscheidbarkeit anhand der visuellen Gestalt oder ihre Zuordnung zu einer definierten Klasse von Zeichen müssen transformiert werden. Zum einen in maschinell ausführbare Instruktionen (programmgesteuerte Interpretationen), zum anderen in das Symbolsystem der digitalen Zeichen (diskretes Symbolschema, digitales Schema), die Werte darstellen und insofern operativ erzeugbar sind. Die Reduktion verschiedener Zeichengestalten auf digitale Zeichen normiert die zu verarbeitenden Entitäten auf der subsymbolischen Ebene und macht diese ineinander überführbar. Die digitalen Zeichen selbst sind deshalb im Unterschied zu den graphischen Entitäten der Schrift formbar. Die Fortführung der Schrift ins Dynami-

erheblich erweitert. Das gilt auch für die graphische Darstellungsweise, die erst durch den enormen Umfang der Datenwerte zu einer ikonischen wird. Der quantitative Effekt schlägt mit der numerischen Simulation in qualitative Effekte um.

sche, als Konsequenz der semiotischen Revolution der Mechanisierung des Zeichenumgangs, eröffnet die direkte semiotische Modellierung von Prozessen – die zuvor nur durch formal-operative Zeichensysteme symbolisierbar waren - im (virtuellen) Raum und in der Zeit, indem die numerischen Operationen Objekte und Abläufe simulieren bzw. semiotisch erzeugen.