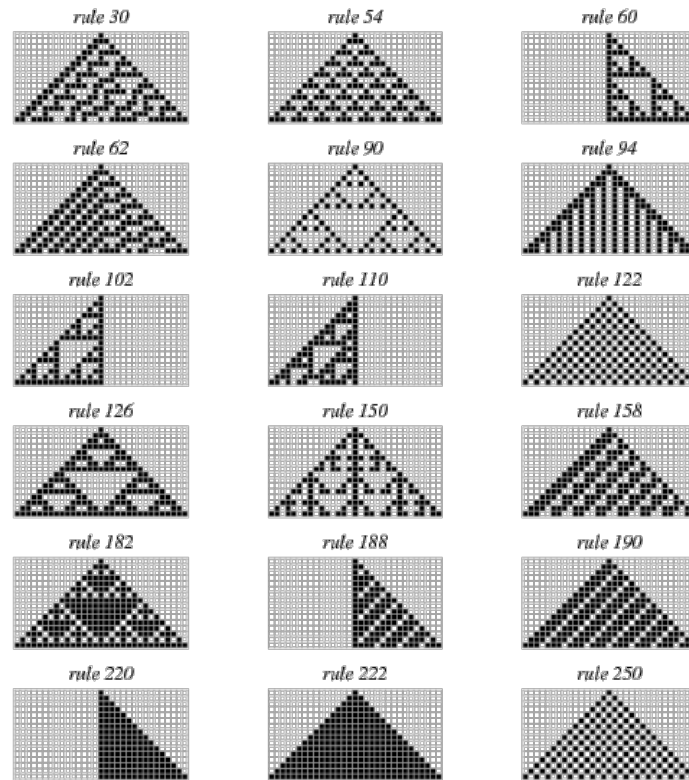


Kultur-, Sozial- und Bildungswissenschaftliche Fakultät  
Modul III: Zeitbasierte Medien und zeitkritische Medienprozesse  
Seminar: Fraktale



## Entwicklung fraktaler Strukturen mit Hilfe eindimensionaler zellulärer Automaten

-

### Am Beispiel der *Regel 90* nach Stephen Wolfram

Vorgelegt bei:

**Dr. Stefan Höltgen**

Abgabedatum:

**05. 02. 2018**

Vorgelegt von:

**Franziska Guddat**

Einschreibnummer: 576225

E-Mai: f.guddat@googlemail.com

## Inhaltsverzeichnis

<b>1. Einleitung.....</b>	<b>2</b>
<b>2. Fraktale.....</b>	<b>3</b>
2.1 Selbstähnlichkeit und Skaleninvarianz.....	4
2.2 Die fraktale Dimension.....	5
<b>3. Sierpinski-Dreieck.....</b>	<b>7</b>
<b>4. Zelluläre Automaten.....</b>	<b>9</b>
4.1 Vorbemerkungen und grundlegende Eigenschaften.....	9
4.2 Wolframs eindimensionales Universum.....	11
<b>5. Darstellung des Sierpinski-Dreiecks nach Wolfram.....</b>	<b>13</b>
5.1 Rule 90.....	13
5.2 Programmierung.....	15
<b>6. Pascal- und Sierpinski-Dreieck.....</b>	<b>19</b>
<b>7. Fazit.....</b>	<b>20</b>
<b>8. Literaturverzeichnis .....</b>	<b>22</b>
<b>9. Abbildungsverzeichnis.....</b>	<b>24</b>

# 1. Einleitung

*„Wolken sind keine Kugeln, Berge keine Kegel, Küstenlinien keine Kreise. Die Rinde ist nicht glatt - und auch der Blitz bahnt sich seinen Weg nicht gerade. [...] Die Existenz solcher Formen fordert uns zum Studium dessen heraus, was Euklid als formlos beiseitelässt, führt uns zur Morphologie des Amorphen.“<sup>1</sup>*

Mandelbrot hat bewiesen, dass Geometrie weit mehr leisten kann als Dreiecke, Quader und Kugeln zu beschreiben. In seinem 1975 publizierten Werk *Die fraktale Geometrie der Natur* erklärt er, dass viele in der Natur vorkommende Formen mathematisch als Fraktale bezeichnet werden können. Ein Wort, das er erfunden hatte, um Formen zu definieren, die gezackt und gebrochen aussehen.

Die Etablierung des Computers und die dadurch zur Verfügung stehenden visuellen Darstellungen ermöglichten den Siegeszug der fraktalen Geometrie und ihren raschen Einzug in zahlreiche Forschungsfelder. Es wurden verschiedenste Beziehungen zwischen dieser neuen Mathematik und zahlreichen anderen Wissenschaftsbereichen erkennbar, wie zum Beispiel der Biologie, der Geologie, der Städteentwicklung, der Wirtschaft, der Technologie und der Kunst.

Was aber ist ein Fraktal? Um diese Frage zu beantworten werde ich zunächst die grundlegenden Eigenschaften nennen und diese schließlich am Sierpinski-Dreieck, einem mathematischen Fraktal, aufzeigen. Fraktale Strukturen können auch durch zelluläre Automaten, die der Modellierung räumlich diskreter dynamischer Systeme dienen, erzeugt werden. So entsteht in der Zeitentwicklung des eindimensionalen Regel 90-Automaten ein Sierpinski-Dreieck. Zur Veranschaulichung implementiere ich dieses Modell unter Verwendung der Programmiersprache Java. Warum entsteht hier eine fraktale Struktur und inwiefern lässt sich ein medienwissenschaftlicher Bezug herstellen? Ich beziehe mich besonders auf die Arbeiten und Forschungserkenntnisse des britischen Physikers und Mathematikers Stephen Wolfram zu elementaren zellulären Automaten. Die Ergebnisse werden in einem abschließenden Fazit festgehalten.

---

<sup>1</sup> Mandelbrot, Benoît B.: *Die fraktale Geometrie der Natur*. Basel 1987, S. 13.

## 2. Fraktale

*„Die klassische Geometrie lieferte erste Näherungen an die Struktur physikalischer Objekte. [...]. Die fraktale Geometrie ist deren Erweiterung: Physikalische Strukturen – von Farnen, Wolken bis hin zur Galaxie – können mit ihrer Hilfe exakt modelliert werden.“<sup>2</sup>*

Der Terminus Fraktale wurde im Jahre 1975 vom polnisch-französischen Mathematiker Benoit B. Mandelbrot etabliert.<sup>3</sup> Bei Fraktalen handelt es sich um Objekte, die komplexerer und irregulärer Natur sind als die gängigen Objekte der klassischen, euklidischen Schulgeometrie wie beispielsweise die Linie, das Dreieck oder das Quadrat. So ist es nicht verwunderlich, dass Mandelbrot den Begriff von dem lateinischen Adjektiv *fractus* herleitete. Dies bedeutet so viel wie „in Stücke zerbrochen“ und „irregulär“.<sup>4</sup> Mandelbrot gilt zwar als Vater der Fraktal Forschung<sup>5</sup>, die ersten fraktalen Objekte<sup>6</sup> tauchten jedoch schon um 1900, „beim Versuch, den mathematischen Gehalt und die Grenzen von grundlegenden Begriffen (z.B. ‘Stetigkeit‘ oder ‘Krümmung’) vollständig zu erforschen“<sup>7</sup>, auf. In mathematischen Kreisen betrachtete man diese Objekte zunächst lediglich als von der Norm abweichende Kuriositäten und mathematische Monster, deren Bedeutung man nicht näher untersuchte. Mandelbrot demonstrierte schließlich, dass die frühen mathematischen Fraktale eher die Regel und nicht die Ausnahme darstellen und sich nicht durch die euklidische Geometrie beschreiben lassen. Die euklidischen glatten, einfachen Körper stoßen bei komplexeren Strukturen an ihre Grenzen. Mandelbrot erkannte, dass die sogenannten Monster „viele Eigenschaften mit natürlichen Formen gemeinsam haben“.<sup>8</sup> Er fasste die Ergebnisse schließlich in seinem 1982 erschienenen Buch *The Fractal Geometry of Nature* zusammen. Da wir es mit einem geometrischen Konzept zu tun haben, für das es bisher keine konkrete Definition und auch keine

---

<sup>2</sup> Barnsley, Michael F.: Fraktale. Theorie und Praxis der Deterministischen Geometrie. Heidelberg 1995, S. 2.

<sup>3</sup> Vgl. Jürgens, Hartmut; Peitgen, Heinz-Otto; Saupe, Dietmar: Bausteine des Chaos. Fraktale. Stuttgart 1992, S.81.

<sup>4</sup> Vgl. Mandelbrot.: Die fraktale Geometrie der Natur, S. 16f.

<sup>5</sup> Vgl. Jürgens; Peitgen; Saupe: Bausteine des Chaos, Vorwort.

<sup>6</sup> Mathematiker wie Georg Cantor (Cantor-Menge 1870), Giuseppe Peano (Peano-Kurve 1890), Helge von Koch (Koch-Kurve 1904), Waclaw Sierpinski (Sierpinski-Dreieck 1915) und Gaston Maurice Julia (Julia-Menge 1918) entdeckten schon früh Objekte mit Strukturen, die sich nicht mit der klassischen euklidischen Schulgeometrie erklären ließen. Diese Formen dienten zunächst eher als Beweis der Abweichung vom Normalen.

<sup>7</sup> Jürgens; Peitgen; Saupe: Bausteine des Chaos, S.81.

<sup>8</sup> Ebd.



eindeutige, und allgemein anerkannte Theorie gibt, erscheint es am sinnvollsten ein Fraktal nach seinen typischen Eigenschaften zu beschreiben.

Ein Fraktal ist:

- 1) das Resultat eines unendlich oft wiederholten rekursiven Erzeugungsprozesses.<sup>9</sup>
- 2) ein Objekt dem als Dimension eine nichtganzzahlige Zahl zugeordnet wird.<sup>10</sup>
- 3) eine Kurve die über keine charakteristische Länge verfügt, d.h. unendlich lang ist.<sup>11</sup>
- 4) ein Objekt, das auf jeder Größenskala aus mehreren gleichgroßen Teilen, die exakte Kopien des Ganzen sind, besteht (Selbstähnlichkeit)<sup>12</sup>

An dieser Stelle ist noch anzumerken, dass die aufgezählten Attribute jedoch oft nicht auf ein und dasselbe fraktale Objekt zutreffen. In den nächsten Teilkapiteln werden die Begrifflichkeiten Selbstähnlichkeit und Dimension näher erläutert.

## 2.1 Selbstähnlichkeit und Skaleninvarianz

Eines der entscheidendsten Charakteristika eines Fraktals ist die Selbstähnlichkeit, die wie bei mathematischen Fraktalen strikt bzw. exakt oder aber wie bei in der Natur vorkommenden Objekten rein statistisch sein kann.

Erhält man bei unendlicher Vergrößerung eines Objekts immer wieder die ursprüngliche Grundstruktur, spricht man von einer strikten Selbstähnlichkeit. Das Fraktal als Ganzes sieht also exakt so aus wie ein Teil davon, das genauso aussieht wie das nächst kleinere Teil. Die Ähnlichkeit des Musters setzt sich kontinuierlich fort. Das Ganze besteht also aus vielen Kopien seiner selbst.<sup>13</sup> Diese Form der Selbstähnlichkeit bezeichnet man aufgrund der Unabhängigkeit gegenüber Maßstabsveränderungen auch als Skaleninvarianz.<sup>14</sup> Dies lässt sich damit erklären, dass mathematische Fraktale, durch

---

<sup>9</sup> Vgl. Falconer, Kenneth: Fractal Geometry: Mathematical foundations and applications. Chichester 2003, Introduction.

<sup>10</sup> Vgl. Bräuer, Kurt: Chaos, Attraktoren und Fraktale. Mathematische und physikalische Grundlagen nichtlinearer Phänomene mit Anwendungen in Physik, Biologie und Medizin. Berlin 2001, S. 47.

<sup>11</sup> Vgl. Jürgens; Peitgen; Saupe: Bausteine des Chaos, S. 241.

<sup>12</sup> Vgl. ebd., S.163.

<sup>13</sup> Vgl. Bräuer: Chaos, Attraktoren und Fraktale, S. 24.

<sup>14</sup> Vgl. ebd.,

einen unendlich wiederholten deterministischen Algorithmus erzeugt werden, der zu einer exakten Selbstähnlichkeit bis ins unendlich Kleine führt. Ein Fraktal ist also ein Endprodukt, dass aus der unendlichen Iteration eines wohldefinierten geometrischen Prozesses entsteht. Dieser geometrische Prozess, der im Allgemeinen von extrem einfacher Natur ist, bestimmt die endgültige Struktur, die aufgrund der unendlichen Wiederholung ein außerordentlich komplexes Aussehen besitzt.<sup>15</sup> Anhand des Sierpinski-Dreiecks, einem sogenannten IFS-Fraktal (Iterated Function System), soll der Begriff der exakten Selbstähnlichkeit in Kapitel 3 näher veranschaulicht werden.

Natürliche vorkommende fraktale Objekte, wie z.B. der Romanesco, weisen hingegen eine statistische Selbstähnlichkeit auf, die nur über ein paar Größenordnungen bestehen kann. „Unterhalb bestimmter Größenverhältnisse zerfällt Materie in einer Ansammlung von Molekülen, Atomen und noch etwas weiter in Elementarteilchen. Auf dieser Stufe wird es natürlich unsinnig, Miniaturen von verkleinerten Kopien eines vollständigen Objekts zu erwarten“<sup>16</sup> Die Teilkopien bzw. Miniaturausgaben sehen dem original zwar sehr ähnlich, es sind aber immer kleine Abweichungen zu erkennen.<sup>17</sup>

## 2.2 Die fraktale Dimension

*„Eine Linie [ist eine] breitenlose Länge. [...]Eine Fläche ist, was nur Länge und Breite hat [...]Ein Körper ist, was Länge, Breite und Tiefe hat.“<sup>18</sup>*

Das Wort Dimension stammt vom lateinischen Begriff *dimetiri* ab und bedeutet „nach allen Seiten abmessen“. Sprechen wir über die Dimension eines Objekts, beziehen wir uns auf dessen Ausdehnung im euklidischen Raum  $E$ .<sup>19</sup> Somit ist eine Linie eindimensional, eine Fläche zweidimensional und ein Körper dreidimensional. Die Dimension eines Objektes ist immer ganzzahlig. Fraktale hingegen sind nur Teilmengen

<sup>15</sup> Vgl. Haftendorn, Dörte: Mathematik sehen und verstehen. Schlüssel zur Welt. Heidelberg 2010, S. 95.

<sup>16</sup> Jürgens; Peitgen; Saupe: Bausteine des Chaos, S. 172

<sup>17</sup> Vgl. Ebd.

<sup>18</sup> Schönbeck, Jürgen: Euklid. Um 300 v. Chr., Wiesbaden 2003, S. 184.

<sup>19</sup> Vgl. Redaktion Schule und Lernen: Schülerduden. Mathematik 2 - Ein Lexikon zur Schulmathematik für das 11. bis 13. Schuljahr. Mannheim 2000, S. 86.

des euklidischen Raumes  $E$ . „Ein Fraktal ist eine hinreichend komplizierte Punktmenge in einem geometrisch einfach Raum.“<sup>20</sup>

Die fraktale Dimension ist ein für das Fraktal kennzeichnendes Maß, welches nicht mehr nur ganzzahlig sein muss, sondern auch gebrochene Werte zulässt. Berechnen lässt sich diese unter anderem durch die Hausdorff-Besicovitch-Dimension<sup>21</sup>, mit Hilfe derer Mandelbrot die *mathematischen Monster* schließlich als eine neue Kategorie von Objekten gegenüber den klassischen, euklidischen Gebilden abgrenzen konnte. Mandelbrots Definition des Begriffs Fraktale basiert auf der Diskrepanz zwischen  $d_{hb}$  und  $d_{top}$ <sup>22</sup> einer fraktalen Struktur. Die Dimension  $D_{top}$  ist immer eine ganzzahlige Zahl, während  $D_{hb}$  auch eine nichtganzzahlige Zahl sein kann. Es handelt sich bei einem Objekt  $F$  um ein Fraktal, wenn gilt:

$$d_{hb}(F) > d_{top}(F)$$

„Ein Fraktal ist nach Definition eine Menge, deren Hausdorff-Biscovitch-Dimension echt die topologische Dimension übersteigt.“<sup>23</sup> Befindet sich ein fraktales Gebilde im euklidischen Raum  $\mathbb{R}^E$  ist  $d_{hb}$  größer als 0 und kleiner als die Dimension von  $E$ .<sup>24</sup> Ein Fraktal kann aber auch eine ganzzahlige fraktale Dimension besitzen wie beispielsweise raumfüllende Kurven (Peano-Kurve, Hilbert-Kurve).<sup>25</sup>

Die Selbstähnlichkeitsdimension  $d_s$  ist ein Sonderfall der Hausdorff-Besicovitch-Dimension. Sie kann nur auf exakt selbstähnliche Objekte angewendet werden und berechnet sich bedeutend schneller als die  $d_{hb}$ .<sup>26</sup> Wenn eine exakt selbstähnliche Figur  $F \subset \mathbb{R}^n$  aus  $k$  Kopien ihrer selbst zusammengesetzt ist, die alle um den Faktor  $0 < s < 1$  skaliert sind, ergibt sich die Selbstähnlichkeitsdimension von  $F$  folgendermaßen:

---

<sup>20</sup>Zeitler; Pagon: Fraktale Geometrie - Eine Einführung. Wiesbaden 2000, S. 166.

<sup>21</sup>Felix Hausdorff stellte die Hausdorff-Dimension 1919 vor. Sie bietet die Möglichkeit beliebigen metrischen Räumen eine Dimension zuzuordnen. Für euklidische Objekte wie Linien oder Flächen stimmt  $D_{top}$  mit  $D_{hb}$  überein. Abram Samoilovitch Besicovitch ergänzte die Überlegungen Hausdorffs für den Fall, dass man die Dimension von Nichtstandardobjekten berechnen will.

<sup>22</sup> $d_{top}$  = Topologische Dimension.

<sup>23</sup>Mandelbrot: Die fraktale Geometrie der Natur, S. 27.

<sup>24</sup>Vgl. ebd.

<sup>25</sup>Es gilt  $d_{hb} = 2$ , aber  $d_{top} = 1$ . Es handelt sich um ein Fraktal, da  $2 > 1$ .

<sup>26</sup>Vgl. Zeitler; Pagon: Fraktale Geometrie - Eine Einführung, S. 165.

$$k = \frac{1}{s^{d_s}} = \left(\frac{1}{s}\right)^{d_s}$$

$$\log k = \log \left( \left(\frac{1}{s}\right)^{d_s} \right) = d_s \log \left( \frac{1}{s} \right)$$

$$d_s = \frac{\log k}{\log \left( \frac{1}{s} \right)}$$

Hat man es also mit einer exakt selbstähnliche Figur zu tun, die aus  $k$  verkleinerte Kopien ihrer selbst besteht und ist der Verkleinerungsfaktor von der Ausgangsfigur zur verkleinerten Kopie  $s$ , dann kann man die Dimension der Figur über die obenstehende Formel berechnen.<sup>27</sup>

### 3. Sierpinski-Dreieck

Das Sierpinski-Dreieck ist ein klassisches mathematisches Fraktal und gilt als Musterbeispiel für exakte Selbstähnlichkeit. Es wurde im Jahre 1915 von dem populären, polnischen Mathematiker und Hochschullehrer Waclaw Sierpiński beschrieben, um zu verdeutlichen, dass Begriffe wie Linie und Fläche durchaus nicht trivial sind.<sup>28</sup> Um das Sierpiński-Dreieck zu konstruieren, geht man folgendermaßen vor:

Ein gleichseitiges Dreieck wird in vier kongruente Dreiecke geteilt. Dabei wird das mittlere Dreieck entfernt, wodurch die Fläche um  $\frac{1}{4}$  reduziert wird. Es bleiben also drei schwarze kongruente Dreiecke übrig, welche zusammen  $\frac{3}{4}$  des ursprünglichen Flächeninhalts entsprechen. Mit diesen Teildreiecken wird analog verfahren. Diese Prozedur kann nun beliebig oft durchgeführt werden. Mit jedem weiteren Iterationsschritt verdreifacht sich die Zahl der Dreiecke.<sup>29</sup>

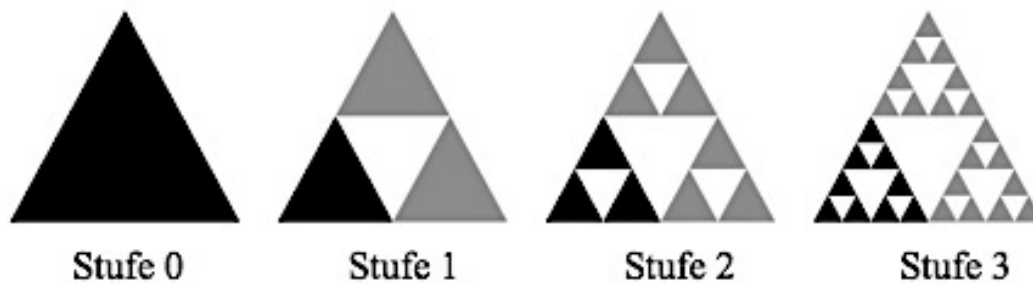
---

<sup>27</sup> Vgl. Jürgens; Peitgen; Saupe: Bausteine des Chaos, S.245-250.

<sup>28</sup> Vgl. Hoffmann, Dirk W.: Grenzen der Mathematik. Eine Reise durch die Kerngebiete der mathematischen Logik. Heidelberg 2011. S. 327.

<sup>29</sup> Vgl. Westermann, Thomas: Mathematische Begriffe visualisiert mit Maple für Lehrer und Dozenten. Heidelberg 2001, S. 107.

Abbildung 1: Die ersten drei Iterationsschritte des Sierpinski-Dreiecks<sup>30</sup>



Während der Flächeninhalt des Sierpinski-Dreiecks für  $n \rightarrow \infty$  gegen Null konvergiert, strebt der Umfang bzw. die Randlinie für  $n \rightarrow \infty$  gegen unendlich. Das Sierpinski-Dreieck hat somit einen unendlich großen Umfang, der eine Fläche der Größe 0 umschließt. Da es sich bei der betrachteten Figur um ein mathematisches Fraktal handelt, welches exakt selbstähnlich ist und ein hohes Maß an Skaleninvarianz aufweist, ist es in diesem Fall sinnvoll die Selbstähnlichkeitsdimension zu berechnen. Für das Sierpinski-Dreieck erhält man  $k=3$  Kopien und den Skalierungsfaktor  $s = \frac{1}{2}$ . Somit ergibt sich folglich:

$$d_s = \frac{\log 3}{\log \frac{1}{0.5}} = \frac{\log 3}{\log 2} \approx 1,585$$

Die Dimension des Sierpinski-Dreiecks ist größer als eine Strecke, aber kleiner als eine Fläche. Eine Zuordnung zu  $D=1$  (Linie) bzw.  $D=2$  (Ebene) kann nicht erfolgen, da jeweils das mittlere Dreieck entfernt wird.

Am Beispiel des Sierpinski-Dreiecks soll im folgenden Kapitel gezeigt werden, dass der Output zellulärer Automaten tatsächlich fraktal sein kann.

<sup>30</sup> Uni Bremen: Ausgesuchte Fraktale und ihre Selbstähnlichkeitsdimension. Verfügbar unter: [http://www.math.uni-bremen.de/didaktik/ma/ralbers/Veranstaltungen/MaDenken1313/Material/Dim\\_Skript\\_K2b.pdf](http://www.math.uni-bremen.de/didaktik/ma/ralbers/Veranstaltungen/MaDenken1313/Material/Dim_Skript_K2b.pdf), abgefragt am 20.01.2018.

## 4. Zelluläre Automaten

### 4.1 Vorbemerkungen und grundlegende Eigenschaften

Zelluläre Automaten werden zur argentenbasierten Modellierung und Simulation verschiedener Sachverhalte genutzt. „Die Entstehung des Lebens, Modelle des Verkehrswesens, populationsdynamische Entwicklungen oder die Ausbreitung von Epidemien stellen dabei nur eine kleine Auswahl der unzähligen Szenarien dar, die sich mit zellulären Automaten simulieren lassen.“<sup>31</sup> Die Entwicklung zellulärer Systeme geht auf den Mathematiker John von Neumann zurück. Sein Ziel war es, das mathematische Modell eines Automaten zu konstruieren, der in der Lage ist sich selbst zu reproduzieren und damit grundlegende Züge des Lebens trägt.<sup>32</sup> Unterstützung erhielt er von seinem polnisch-amerikanischen Fachkollegen Stanislaw Ulam, der für Neumann eine geeignete, formale Sprache entwickelte, durch die sich die Wechselwirkung einer Vielzahl von Komponenten nach bestimmten Regeln darstellen ließ. Ulam schlug die Felder eines einfachen, schachbrettartigen Gitters als Lebensraum dieser Komponenten vor, die Informationen aus ihrer direkten Nachbarschaft in die eigene Lebensentwicklung mit einbeziehen. Von Neumann nannte die einzelnen Felder des Gitters Zellen, in Analogie zu den Grundbausteinen des Lebens. In der Automatentheorie können grundsätzlich alle Zellen als einzelne Automaten bezeichnet werden, die alle nach denselben Regeln funktionieren. Die Besonderheit eines solchen Teilautomaten besteht darin, dass seine Zustandsentwicklung nicht nur von seinem eigenen Zustand, sondern auch von den Zuständen der benachbarten Teilautomaten abhängt.<sup>33</sup> Mit dem Spiel Life des Mathematikers John Horton Conway erlangten zelluläre Automaten schließlich große Popularität an Universitäten und Forschungseinrichtungen.<sup>34</sup> Der Gründer der Softwarepakets Mathematica Stephen Wolfram untersuchte neue Anwendungen zellulärer Automaten vor allem im Bereich der Physik. Er beschränkte sich dabei in seinen Forschungsarbeiten auf eindimensionale zelluläre Automaten.<sup>35</sup>

---

<sup>31</sup> Scholz, Daniel: Pixelspiele. Modellieren und Simulieren mit zellulären Automaten. Heidelberg 2014, Vorwort.

<sup>32</sup> Ebd., S.1.

<sup>33</sup> Vgl. Gerhardt, Martin; Schuster, Heike: Das digitale Universum. Zelluläre Automaten als Modelle der Natur. Wiesbaden 1995, S. 17 f.

<sup>34</sup> Vgl. Ebd., S. 15.

<sup>35</sup> Vgl. Gerhardt; Schuster: Das digitale Universum, S. 63.

Die grundlegenden Eigenschaften eines zellulären Automaten können wie folgt zusammengefasst werden:

- „Seine Entwicklung findet in Raum und Zeit statt.
- Sein Raum ist eine diskrete Menge zahlreicher Zellen.
- Jeder dieser Zellen hat nur eine endliche Anzahl möglicher Zustände.
- Die Zustände der Zellen verändern sich in diskreten Zeitschritten.
- Alle Zellen sind identisch und verhalten sich nach den gleichen Entwicklungsregeln.
- Die Entwicklung einer Zelle hängt nur ab von ihrem Zustand und dem sie ihr lokal umgebenden Nachbarszellen.“<sup>36</sup>

Der Fokus der folgenden Teilkapitel liegt auf Wolframs Forschungen zu eindimensionalen zellulären Automaten und ihren Eigenschaften. Sie zählen zu den einfachsten Automaten, führen aber dennoch zu interessanten und komplexen Mustern und funktionieren nach vollkommen deterministischen Regeln.

---

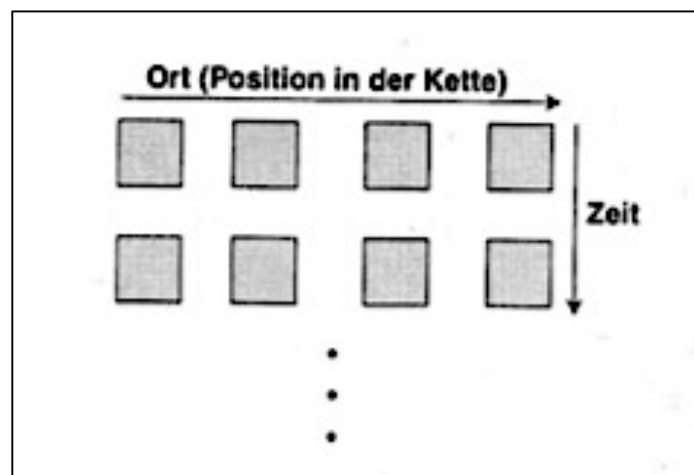
<sup>36</sup> Ebd., S. 18 f.

## 4.2 Wolframs eindimensionales Universum

*“In the existing sciences whenever a phenomenon is encountered that seems complex it is taken almost for granted that the phenomenon must be the result of some underlying mechanism that is itself complex. But my discovery that simple programs can produce great complexity makes it clear that this is not in fact correct.”<sup>37</sup>*

Der elementare Automat, auch eindimensionales Universum genannt, wurde im Jahre 1983 von Stephen Wolfram vorgestellt und gilt als der einfachste nicht-triviale zelluläre Automat.<sup>38</sup> Er setzt sich aus nur einer Raum- und Zeitdimension zusammen und ist auf einen booleschen Zustandsraum beschränkt (Zustände 0 und 1).

Abbildung 2: Zeitentwicklung eindimensionaler zellulärer Automaten<sup>39</sup>



Im eindimensionalen Fall sind nur die zwei unmittelbaren Nachbarn eines Elements von Bedeutung. Der Wert einer Zelle  $z$  zum Zeitpunkt  $t+1$  hängt von den Werten der Zellen  $z-1$  (Nachbarzelle 1),  $z$  (Zentralzelle) und  $z+1$  (Nachbarzelle 2) zum Zeitpunkt  $t$  ab. Es sind also zwei Zustände auf drei Zellen zu verteilen.

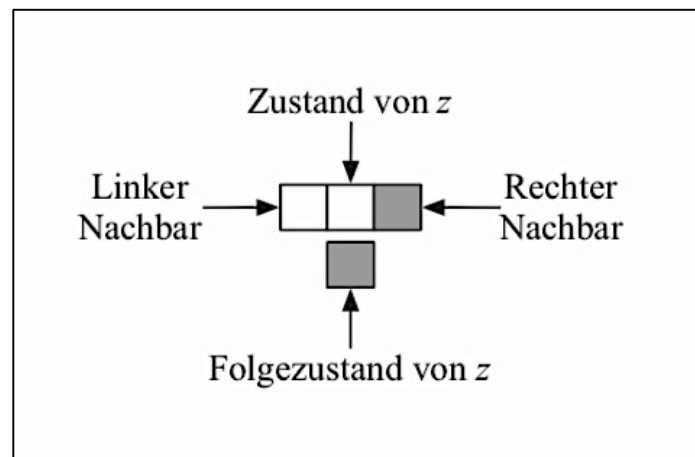
<sup>37</sup> Wolfram, Stephen: A new kind of science. 2002, S. 4.

<sup>38</sup> Vgl. Martin, Olivier; Odlyzk, Andrew M.; Wolfram, Stephen: Algebraic properties of cellular automata. In: Communications in Mathematical Physics H. 93/20. Jg. (1984), S. 219-258, S. 219.  
<http://www.stephenwolfram.com/publications/academic/algebraic-properties-cellular-automata.pdf>, S. 219

<sup>39</sup> Hütt, Marc-Thorsten: Datenanalyse in der Biologie. Heidelberg 2001, S. 116.



Abbildung 3: Zeitschritt-Verfahren für eindimensionale zelluläre Automaten<sup>40</sup>



Somit ergeben sich rein kombinatorisch  $2^3=8$  Konstellationsoptionen.<sup>41</sup> Durch diesen Gedankengang wird das wesentliche Vorgehen beim Umgang mit zellulären Automaten noch einmal verdeutlicht: Die relevanten Informationen sind durch den Zustandsraum und die Überführungsregeln definiert. Sie stellen die Grundlage für die simulierten Zeitentwicklungen dar. „Ein solches Regelwerk ist gerade dann vollständig spezifiziert, wenn es für jede der 8 möglichen Nachbarschaftskonstellationen das Update (also den von der Zentralzelle im nächsten Zeitschritt angenommenen Wert) festlegt.“<sup>42</sup> Für jede der 8 Konstellationen gibt es wieder nur zwei Möglichkeiten (0 und 1) und somit  $2^8=256$  unterschiedliche Entwicklungsregeln für elementare zelluläre Automaten.<sup>43</sup> Diese Regeln beruhen auf dem Binär- bzw. Dualsystem. Im Gegensatz zum gängigen Dezimalsystem, in dem die Ziffern 0 bis 9 verwendet werden, werden Zahlen im Dualsystem nur mit den Ziffern des Wertes 0 und 1 dargestellt.<sup>44</sup>

Aus einigen Regelwerken ergibt die Zeitentwicklung eindimensionaler zellulärer Automaten eine fraktale Struktur.<sup>45</sup> Das Charakteristikum der Selbstähnlichkeit ist dabei hingegen nicht wie bei mathematisch exakten Fraktalen, die Resultat eines unendlichen Iterationsprozesses sind, gegeben, da eine räumliche Begrenzung vorliegt.<sup>46</sup> Für die vorliegende Arbeit sind besonders die Regelwerke zellulärer Automaten interessant, die

<sup>40</sup> Ebd., S. 117.

<sup>41</sup> Vgl. Gerhardt; Schuster: Das digitale Universum, S. 29.

<sup>42</sup> Hütt: Datenanalyse in der Biologie, S. 117.

<sup>43</sup> Vgl. Gerhardt; Schuster: Das digitale Universum, S. 29.

<sup>44</sup> Wolfram: A new kind of science, S. 116 f.

<sup>45</sup> Vgl. Hütt: Datenanalyse in der Biologie, S. 236.

<sup>46</sup> Ebd.

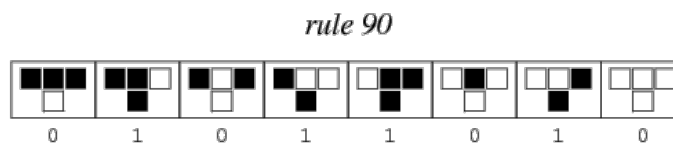
eine fraktale Struktur erzeugen. Als veranschaulichendes Beispiel soll im Folgenden die Zeitentwicklung des eindimensionalen zellulären Automaten der Regel 90 dienen, die ein fraktales Objekt – das Sierpinski-Dreieck – erzeugt.<sup>47</sup>

## 5. Darstellung des Sierpinski-Dreiecks nach Wolfram

### 5.1 Rule 90

Der in diesem Teilkapitel betrachtete zelluläre Automat gehört zwar zu den simpelsten, nicht-trivialen Automaten, führt aber dennoch zu einem interessanten und komplexen Muster.

*Abbildung 4: Vollständiger Regelsatz*



Eine Zelle kann wie bereits erläutert nur den Wert 1 (schwarz) oder den Wert 0 (weiß) als Zustand annehmen. Abbildung 4 zeigt das Regelwerk der Regel 90. Die acht möglichen Zellzustände von drei benachbarten Zellen sind in der oberen Zeile angegeben. Die untere Zeile definiert die Regel für die Zeitentwicklung des zellulären Automaten, indem sie den Wert im nächsten Zeitschritt angibt, der aus der jeweiligen Nachbarschaftskonstellation resultiert. Wird eine Zelle an der Position  $i$  im Zellraum zum Zeitpunkt  $t$  mit  $z_i(t)$  beschrieben, ergibt sich für die Entwicklungsregel dieses Automaten die nachstehende Formel:<sup>48</sup>

$$z_i(t+1) = (z_{i-1}(t) + z_{i+1}(t)) \bmod 2$$

Der Wert einer Zelle berechnet sich somit aus der Summe der Werte ihrer beiden unmittelbaren Nachbarszellen im vorhergehenden Zeitschritt modulo 2. Bei der

<sup>47</sup> Ohi, Fumio; Takamatsu, Yoshikazu: Time-space pattern and periodic property of elementary cellular automata. sierpinski gasket and partially sierpinski gasket. In: Japan Journal of Industrial and Applied Mathematics H. 18/11. Jg. (2001), S. 59-73, S. 60.

<sup>48</sup> Vgl. Wolfram, Stephen: Cellular automata and complexity. Colorado 1994, S. 9.

Vorschrift  $x \bmod 2$  wird der Rest einer Zahl  $x$ , der bei der Division durch 2 entsteht, angegeben. Die gleiche Entwicklungsregel lässt sich auch folgendermaßen ausdrücken:

$$z_i(t+1) \begin{cases} 0, \text{ wenn } (z_{i-1}(t) + z_{i+1}(t)) \text{ gerade} \\ 1, \text{ wenn } (z_{i-1}(t) + z_{i+1}(t)) \text{ ungerade} \end{cases}$$

Zur Veranschaulichung wird im Folgenden die Entwicklung einer Beispiel-Konfiguration für einen Zeitschritt gemäß der gegebenen Modulo 2 Regel dargestellt.

```

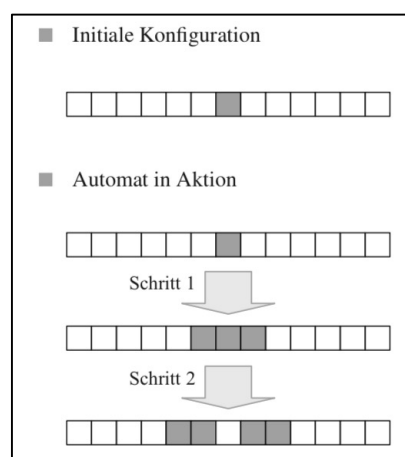
0 1 0 1 1 0 1 1 0 1 0 1 0 1 1 1 0 0 0 1 0
0 0 1 1 0 1 1 0 0 0 0 0 1 0 1 1 0 1 0

```

Die Bezeichnung Regel 90 ist auf Stephen Wolframs Binär-Dezimal-Darstellung für die Regeln eindimensionaler zellulärer Automaten zurückzuführen. Im Fall der Regel 90 erhält man also die Werte: 01011010.<sup>49</sup>

Die Zeitentwicklung des Regel 90-Automaten ergibt eine Approximation bzw. Annäherung des Sierpinski-Dreiecks, wenn eine auf 1 gesetzte Zentralzelle verwendet wird.

*Abbildung 5: Zeitenwicklung eines eindimensionalen zellulären Automaten unter Verwendung der Regel 90<sup>50</sup>*



<sup>49</sup> Vgl. Wolfram, Stephen: Cellular automata and complexity. Colorado 1994, S. 9.

<sup>50</sup> Hütt: Datenanalyse in der Biologie, S. 118.

## 5.2 Programmierung

Ich habe den nachstehenden zellulären elementaren Automaten mit der Programmiersprache Java generiert und die einzelnen Schritte kommentiert (grau gefärbte Schrift). In der beigelegten Datei (s. E-Mail: SierpinskiTriangle.jar) können sowohl die Größe der Zellen als auch die Anzahl der Generationen eingestellt werden. Es besteht sogar die Möglichkeit zwischen verschiedenen Regelwerken zu wählen. Im Folgenden beschränke ich mich aber nur auf die zeitliche Entwicklung des mit der Regel 90 umgesetzten Automaten.

Abbildung 6: Programm „SierpinskiTriangle“ Teil 1

```
1  import javax.swing.JFrame;
2
3  /*
4   * Klasse, welche fuer die Berechnung des SierpinskiDreiecks zustaeendig ist
5   */
6
7  public class SierpinskiTriangle {
8      private Boolean[][] area; //Matrix aus boolschen Werten (Zelle an oder aus)
9      private static int median; //Mittelwert der Feldgrosse, ab welchem das berechnen ausgefuehrt
10         wird
11      private static int[] ruleBinary; //Liste welche die Regeln zur berechnung der naechsten
12         Generation enthaelt
13      private int generations; //Anzahl der Generationen
14      private int size; //Groesse des Zeichengeldes
15
16      //Initialisierende Methode fuer die SierpinskiTriangle Klasse
17      public SierpinskiTriangle(int rule, int size, int generations) {
18          initializeSierpinskiArea(size, generations); //Initialisiert das Zeichenfeld mit der
19             Groesse "size" und fuer "generations" generationen
20          setupRules(rule); //Berechnen der Teilregeln fuer die Zahl in rule
21      }
22      //Aufrufmethode, mit welcher sich eine andere Klasse das berechnete Feld zurueckgeben lassen
23      kann
24      public Boolean[][] getArea(){
25          return area; //Gibt den Inhalt in area zurueck
26      }
27      //Zeichenmethode fuer das SierpinskiDreieck
28      public void drawSierpinski() {
29          area[median][0] = Boolean.TRUE; //Zuerst wird die erste Zelle auf wahr (an/schwarz)
30             gesetzt
31          Boolean prev, curr, next; //Variablen fuer die Vorgaengergeneration von Zellen
32          for (int generation = 0; generation < generations-1; generation++) {
33              //Eine Schleife, welche ueber die Anzahl der vorgegebenen Generationen traversiert
34              for (int cell = 0; cell < size; cell++) { //Eine Schleife fuer jede Zelle in dieser
35                 Generation
36                  prev = getPrev(generation, cell); //Lade den Zustand der Zelle des vorherigen
37                     linken Nachbarn
38                  curr = getCurr(generation, cell); //Lade den Zustand der Zelle der aktuellen
39                     Zelle in der vorherigen Generation
40                  next = getNext(generation, cell); //Lade den Zustand der Zelle des vorherigen
41                     rechten Nachbarn
42                  area[cell][generation+1] = drawNextGenerationCell(prev,curr,next);
43                  //Zeichne den aus den drei vorherigen Zustaenden abgeleiteten Zustand in die
44                     aktuelle Zelle
45              }
46          }
47      }
48  }
```

Abbildung 7: Programm "SierpinskiTriangle" Teil 2

```
40 //Hauptzeichenmethode fuer die Matrix
41 private Boolean drawNextGenerationCell(Boolean prev, Boolean curr, Boolean next) {
42     Boolean output = Boolean.FALSE; //Variable fuer die Ausgabe des berechneten Ergebnisses
43     /*
44      * Der nachfolgende Abschnitt repraesentiert die Abfrage der drei Zustaeude:
45      * 1. Zuerst wird geschaut ob der vorherige linke Nachbar schwarz oder weiss war
46      * 2. Dann wird geschaut ob die Zelle selbst in der vorherigen Generation schwarz
47      *    oder weiss war
48      * 3. Zuletzt wird geschaut ob der vorherige rechte Nachbar schwarz oder weiss war
49      * Da sich die Nachfolgeschritte aus den acht Stellen der Binaerdarstellung der Regel
50      * ergeben werden
51      * diese aus der vorher berechneten Regel liste ausgelesen. Jede Stelle in der Liste
52      * repraesentiert
53      * eine Regel fuer die Zelle danach
54      */
55     if (prev == Boolean.TRUE){
56         if (curr == Boolean.TRUE){
57             if (next == Boolean.TRUE){[ ]} else
58             if (next == Boolean.FALSE){
59                 output = getBool(ruleBinary[1]);
60             }
61         } else
62         if (curr == Boolean.FALSE){
63             if (next == Boolean.TRUE){
64                 output = getBool(ruleBinary[2]);
65             } else
66             if (next == Boolean.FALSE){
67                 output = getBool(ruleBinary[3]);
68             }
69         }
70     } else
71     if (prev == Boolean.FALSE) {
72         if (curr == Boolean.TRUE){
73             if (next == Boolean.TRUE){
74                 output = getBool(ruleBinary[4]);
75             } else
76             if (next == Boolean.FALSE){
77                 output = getBool(ruleBinary[5]);
78             }
79         } else
80         if (curr == Boolean.FALSE){
81             if (next == Boolean.TRUE){
82                 output = getBool(ruleBinary[6]);
83             } else
84             if (next == Boolean.FALSE){
85                 output = getBool(ruleBinary[7]);
86             }
87         }
88     }
89     return output; //Am Ende wird die Ausgabe zurueck gegeben
90 }
91
92 }
```

Abbildung 8: Programm "SierpinskiTriangle" Teil 3

```

93 //Dies ist eine Hilfsfunktion, welche die Stellen der Binaerdarstellung der Regel als
94 //Boolsche Werte (schwarz/weiss) zurueck gibt
95 private Boolean getBool(int i) {
96     if (i == 1) { //wenn 1 dann
97         return Boolean.TRUE; //True = Schwarz
98     }
99     else { //ansonsten
100         return Boolean.FALSE; //False = Weiss
101     }
102 }
103 //Hilfsfunktion um den vorherigen rechten Nachbarn der Zelle auszulesen
104 private Boolean getNext(int generation, int cell) {
105     if (cell+1 >= size) { //Abfrage ob das Ende des Rasters erreicht ist
106         return Boolean.FALSE; //Falls ja wird die vorherige rechte Nachbarzelle als weiss
107         gehandhabt
108     } else {
109         return area[cell+1][generation]; //Ansonsten wird der eigentliche rechte vorherige
110         Nachbar zurueck gegeben
111     }
112 }
113 //Hilfsfunktion um den vorherigen Zustand der aktuellen Zelle auszulesen
114 private Boolean getCurr(int generation, int cell) {
115     return area[cell][generation]; //Gibt den Zustand des vorherigen Zustands der aktuellen
116     Zelle zurueck
117 }
118 //Hilfsfunktion um den vorherigen linke Nachbarn der Zelle auszulesen
119 private Boolean getPrev(int generation, int cell) {
120     if (cell-1 < 0) { //Abfrage ob das Ende des Rasters erreicht ist
121         return Boolean.FALSE; //Falls ja wird die vorherige linke Nachbarzelle als weiss
122         gehandhabt
123     } else {
124         return area[cell-1][generation]; //Ansonsten wird der eigentliche linke vorherige
125         Nachbar zurueck gegeben
126     }
127 }
128 }

```

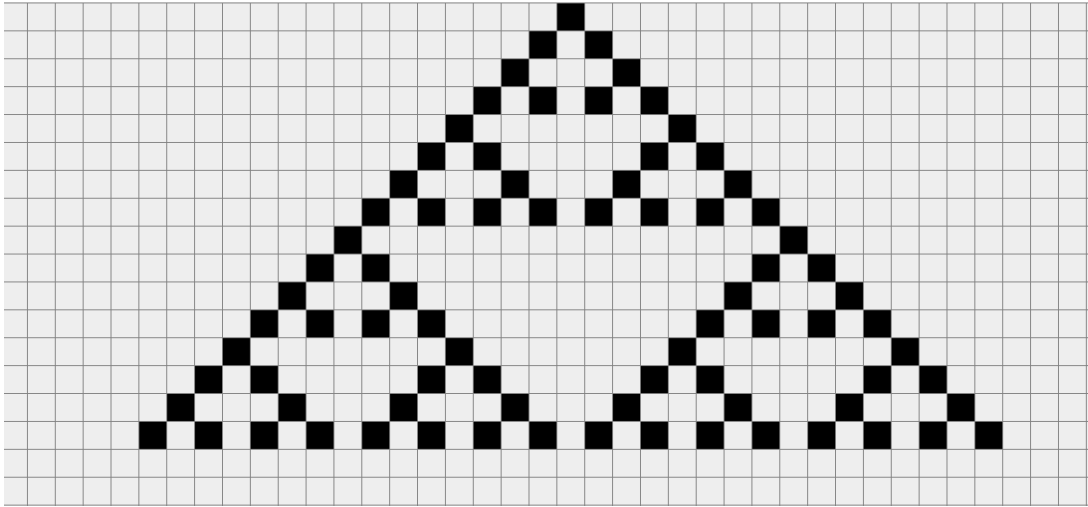
Abbildung 9: Programm "SierpinskiTriangle" Teil 4

```

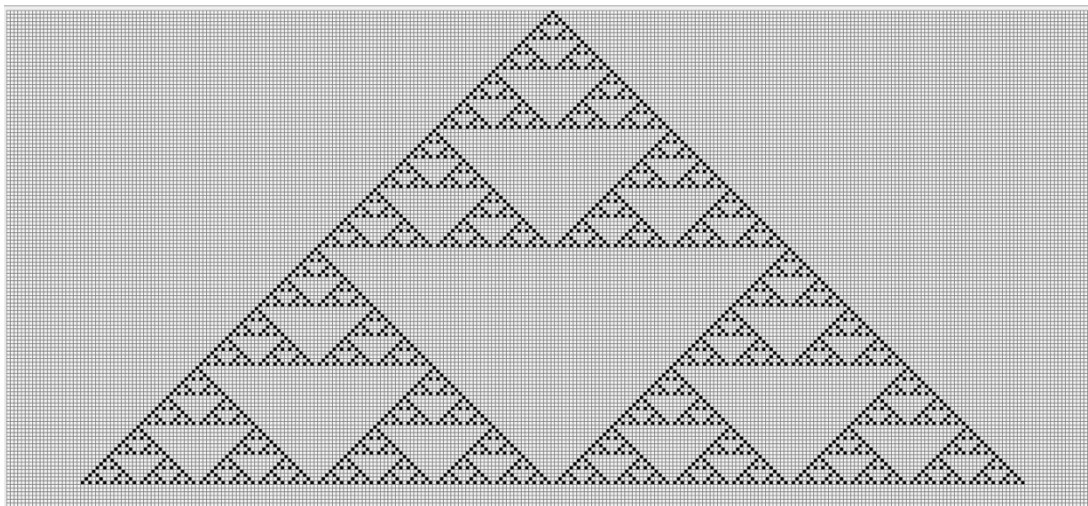
125 //Hilfsfunktion, welche die Zahl "rule" in dessen Binaerdarstellung umrechnet
126 private static void setupRules(int rule) {
127     ruleBinary = new int[8]; //Erstellen einer 8 Stelligen Liste
128     int value = rule; //Uebergeben des Werts aus rule zu value
129     int pos = 7; //Erste Position der Liste, welche beschrieben wird
130     /*
131     * Um eine Dezimalzahl ins Binaersystem umzurechnen teilt man diese so lange durch 2,
132     * bis der Rest dieser Division 0 ergibt
133     * Den Rest der Division durch 2 erhaelt man mit modulo 2 (In Java ist es das % Zeichen).
134     */
135     while (pos > -1) { //Die Schleife wird so lange wiederholt bis alle 8 Stellen belegt
136         wurden
137         ruleBinary[pos] = value%2; //Belege die Stelle pos mit dem modulo 2 von value
138         value = value/2; //Belege value mit dem ohne Rest durch 2 dividierten value
139         pos--; //Zaehle die im naechsten Schritt zu befuellende Zelle runter
140     }
141     /*
142     * Diese vereinfachte Version der Umrechnung funktioniert nur fuer Zahlen von 0 bis 255,
143     * allerdings wird dieser Wert in diesem Programm
144     * nie erreicht, da die auswaehlbaren Regeln auf 8 beschraenkt wurde und keine groesser
145     * als 255 ist
146     */
147 }
148 //Initialisieren der Berechnungsflaeche des Fraktals
149 private void initializeSierpinskiArea(int s, int g) {
150     size = s; //Groesse des Fraktals wird mit s belegt
151     generations = g; //Anzahl der Generationen wird mit g belegt
152     area = new Boolean[size][generations+1]; //Neue Berechnungsflaeche wird erstellt
153     median = size/2; //Mittelwert fuer die erste schwarze Zelle wird erstellt
154     for (int i = 0; i < size; i++) {
155         area[i][0] = Boolean.FALSE; //Setze alle Werte der ersten Generation auf weiss
156     }
157     area[median][0] = Boolean.TRUE; //Setze den Wert der Zelle an der Stelle median auf
158     schwarz
159     /*
160     * Nach diesen Schritten wurde die erste Generation erstellt
161     */
162 }
163 }

```

*Abbildung 10: Zeitentwicklung nach 16 Generationen*



*Abbildung 11: Zeitentwicklung nach 128 Generationen*





## 6. Pascal- und Sierpinski-Dreieck

Das entstandene Muster erinnert interessanterweise stark an die Struktur einer bekannten mathematischen Figur und zwar an das Pascalsche Dreieck der Binomialkoeffizienten.<sup>51</sup> Die Entstehung des einen Muster unterscheidet sich im Grunde nicht von der des anderen: Färbt man alle ungeraden Zahlen des Pascalschen Dreiecks schwarz und die geraden weiß, erhält man exakt dasselbe Muster wie bei dem betrachteten zellulären Automaten. Trägt man die Binomialkoeffizienten der binomischen Reihe in die Gitterplätze eines Pascalschen Dreiecks, verfährt man nach einem simplen Bildungsgesetz: Der Wert jeder Zelle ergibt sich aus der Summe der Werte der beiden darüberliegenden Zellen in der vorangegangenen Zeile. Enthalten beide Zellen gerade oder ungerade Zahlen, ist die daraus resultierende Summe eine gerade Zahl. Bestehen die Zellen hingegen aus einer geraden und einer ungeraden Zahl, ist das Ergebnis darunter folglich eine ungerade Zahl.<sup>52</sup> Denkt man sich an Stelle von den konkreten Zahlenwerten der Binomialkoeffizienten die Farben schwarz und weiß für ungerade und gerade Werte, lässt sich die erläuterte Bildungsregel auf den eindimensionalen Automaten übertragen:

„Hat ein Gitterplatz im Pascalschen Dreieck (eine Zelle) in der darüberliegenden Zeile (im vorherigen Zeitschritt) genau einen angrenzenden Gitterplatz, von schwarzer Farbe (genau eine Nachbarszelle mit Zustandswert 1), wird der Gitterplatz schwarz gefärbt (bekommt den Wert 1). Andernfalls bleibt der Platz weiß (auf dem Zustandswert 0). Die geometrische Figur dieses mathematischen Konstrukts des Pascalschen Dreiecks lässt sich plötzlich in einer anderen Sprache untersuchen, nämlich der der zellulären Automaten.“<sup>53</sup>

---

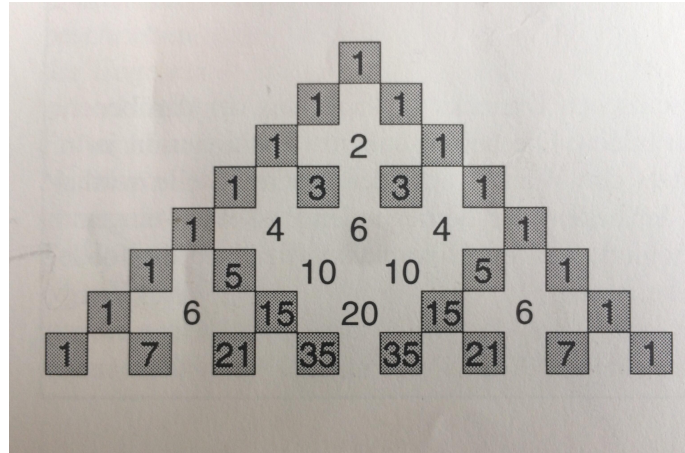
<sup>51</sup> Vgl. Wolfram, Stephen: Cellular automata and complexity. Colorado 1994, S. 9.

<sup>52</sup> Vgl. ebd.

<sup>53</sup> Hütt: Datenanalyse in der Biologie, S. 140.



Abbildung 12: Durch Färbung der ungeraden Zahlen des Pascalschen Dreieck ergibt sich das Sierpinski-Dreieck<sup>54</sup>



## 7. Fazit

Fraktale sind ein relativ neues Gebiet der Mathematik. Wir haben es mit einem geometrischen Konzept zu tun, für das es bisher keine eindeutige Definition gibt. So wird unter anderem gesagt, Fraktale seien Objekte mit fraktaler Dimension. Diese Aussage ist doppelt missverständlich, da die Dimension eine irrationale Zahl sein kann wie beim Sierpinski-Dreieck, aber auch eine ganze Zahl wie bei den raumfüllenden Kurven. Zusammenfassend lässt sich sagen, dass ein Fraktal annähernd selbstähnlich ist (einzelne Teile sehen aus wie das große Ganze), es wird durch einen iterativen Prozess erzeugt, es ist von den Ausgangsbedingungen abhängig und es ist komplex, auch wenn es unter Verwendung eines einfachen Algorithmus beschrieben werden kann. Wie auch Fraktale basieren zelluläre Automaten auf iterativen Prozessen, da für ihre Erzeugung immer wieder derselbe Algorithmus bzw. dasselbe Regelwerk angewandt wird. Dies erklärt, warum mit Hilfe zellulärer Automaten eine fraktale Struktur wie die des Sierpinski-Dreiecks dargestellt werden kann.

Hier ist ein medienepistemologischer Umbruch anzumerken: Denn durch die Etablierung des Computers und die damit verbundenen computergraphischen Möglichkeiten, erhielt die fraktale Geometrie Einzug in zahlreiche Forschungsfelder. Es konnten zum ersten

---

<sup>54</sup> Ebd., S. 67.

Mal Objekte generiert werden, die sich manuell nicht erzeugen lassen, da die mathematischen Prozesse zu „komplex“ sind. Das gleiche gilt auch für die Erforschung von zellulären Automaten. So erklärt Stephen Wolfram selbst in seinem Werk *A New Kind Of Science*: „By the end of the 1970s [...] the situation had changed, and large amount of computer time were becoming readily available. And this is what allowed me [...] to begin my experiments on cellular automata.“<sup>55</sup>

Doch auch durch den Computer kann nur eine annähernde Darstellung eines Fraktals erreicht werden, da ein Programm endlich ist. Es kann also nur eine gewisse Anzahl von Iterationsstufen dargestellt werden. Ein „wahres“ Fraktal hingegen ist Produkt eines unendlichen Iterationsprozesses. Ohne die Leistungsstärke moderner Computer wären Fraktale jedoch unmöglich zu erforschen. Sagen wir, dass eine Grenze, ein Baum oder das Gefäßsystem Fraktale sind, meinen wir eigentlich, dass es fraktale Modelle gibt, die sich an diese Strukturen mit einem hohen Maß an Genauigkeit annähern können. In der realen Welt gibt es keine Fraktale, so wie es keine geraden Linien oder perfekte Kreise gibt. Aufgrund der Tatsache, dass sie die Realität annähern, helfen uns mathematische Modelle, sie besser zu verstehen: „More generally, fractals, cellular automata, chaos, catastrophe and a host of other mathematical developments seem to show that the world, nature and even life itself may, for all messiness and complexity, be computable.“<sup>56</sup>

---

<sup>55</sup> Wolfram: A new kind of science, S. 45.

<sup>56</sup> Schmidt, Artur P.: Zelluläre Automaten. Verfügbar unter: <https://www.heise.de/tp/features/Zellulaere-Automaten-3446142.html>, abgefragt am 28.01.2018.

## 8. Literaturverzeichnis

**Barnsley**, Michael F.: *Fraktale. Theorie und Praxis der Deterministischen Geometrie*. Heidelberg 1995.

**Bräuer**, Kurt: *Chaos, Attraktoren und Fraktale*. Mathematische und physikalische Grundlagen nichtlinearer Phänomene mit Anwendungen in Physik, Biologie und Medizin. Berlin 2001.

**Falconer**, Kenneth: *Fractal Geometry: Mathematical foundations and applications*. Chichester 2003.

**Gerhardt**, Martin; **Schuster**, Heike: *Das digitale Universum*. Zelluläre Automaten als Modelle der Natur.

**Haftendorn**, Dörte: *Mathematik sehen und verstehen*. Schlüssel zur Welt. Heidelberg 2010.

**Hütt**, Marc-Thorsten: *Datenanalyse in der Biologie*. Heidelberg 2001.

**Jürgens**, Hartmann; **Peitgen**, Hans-Otto; **Saupe**, Dietmar: *Bausteine des Chaos*. Fraktale. Stuttgart 1992.

**Mandelbrot**, Benoît B: *Die fraktale Geometrie der Natur*. Basel 1987.

**Martin**, Olivier; **Odlyzk**, Andrew M.; **Wolfram**, Stephen: *Algebraic properties of cellular automata*. In: Communications in Mathematical Physics. Jg. 20, 1984, Nr. 93.

**Ohi**, Fumio; **Takamatsu**, Yoshikazu: *Time-space pattern and periodic property of elementary cellular automata*. Sierpinski gasket and partially sierpinski gasket. In: Japan Journal of Industrial and Applied Mathematics. Jg. 11, 2001, Nr. 18.

**Pagon**, Dušan; **Zeitler**, Herbert: *Fraktale Geometrie. Eine Einführung. Für Studienanfänger, Studierende des Lehramts, Lehrer und Schüler.* Wiesbaden 2000.

**Uni Bremen:** *Ausgesuchte Fraktale und ihre Selbstähnlichkeitsdimension.* URL: [http://www.math.uni-bremen.de/didaktik/ma/ralbers/Veranstaltungen/MaDenken1313/Material/Dim\\_Skript\\_K2b.pdf](http://www.math.uni-bremen.de/didaktik/ma/ralbers/Veranstaltungen/MaDenken1313/Material/Dim_Skript_K2b.pdf), abgefragt am 20.01.2018.

**Redaktion Schule und Lernen:** *Schülerduden. Mathematik 2 - Ein Lexikon zur Schulmathematik für das 11 bis 13. Schuljahr.* Mannheim 2000, S. 86.

**Schmidt**, Artur P.: *Zelluläre Automaten.*  
URL: <https://www.heise.de/tp/features/Zellulaere-Automaten-3446142.html>, abgefragt am 28.01.2018.

**Scholz**, Daniel: *Pixelspiele. Modellieren und Simulieren mit zellulären Automaten.* Heidelberg 2014.

**Westermann**, Thomas: *Mathematische Begriffe visualisiert mit Maple für Lehrer und Dozenten.* Heidelberg 2001.

**Martin**, Olivier; **Odlyzk**, Andrew M.; **Wolfram**, Stephen: *Algebraic properties of cellular automata.* In: *Communications in Mathematical Physics.* Jg. 20, 1984, Nr. 93.

## 9. Abbildungsverzeichnis

<b>Abbildung 1:</b> Die ersten drei Iterationsschritte des Sierpinski-Dreiecks .....	8
<b>Abbildung 2:</b> Zeitentwicklung eindimensionaler zellulärer Automaten .....	11
<b>Abbildung 3:</b> Zeitschritt-Verfahren für eindimensionale zelluläre Automaten.....	12
<b>Abbildung 4:</b> Vollständiger Regelsatz.....	13
<b>Abbildung 5:</b> Zeitenwicklung eines eindimensionalen .....	14
<b>Abbildung 6:</b> Programm „SierpinskiTriangle“ Teil 1.....	15
<b>Abbildung 7:</b> Programm "SierpinskiTriangle" Teil 2.....	16
<b>Abbildung 8:</b> Programm "SierpinskiTriangle" Teil 3.....	17
<b>Abbildung 9:</b> Programm "SierpinskiTriangle" Teil 4.....	17
<b>Abbildung 10:</b> Zeitentwicklung nach 16 Generationen.....	18
<b>Abbildung 11:</b> Zeitentwicklung nach 128 Generationen.....	18
<b>Abbildung 12:</b> Durch Färbung der ungeraden Zahlen des Pascalschen Dreiecks ergibt sich das Sierpinski-Dreieck.....	20