Unter der Oberfläche? Programmierte Schriftlichkeit in digitaler Lyrik

Wiebke Vorrath

Ausgehend von der Prämisse, die Produktion und Rezeption schriftlicher Erzeugnisse in digitalen Medien geschehe in der Regel ohne die Reflexion ihrer grundlegenden Bedingungen, interessiert digitale Lyriker:innen insbesondere die Frage nach den Funktionslogiken computerbasierter Sprachtechnologien. Sie zielt auf das Sichtbarmachen der Arbeitsweisen von Quellcodes und Algorithmen, die der auf dem Interface lesbaren Schrift zugrunde liegen. Nicht selten initiieren diese auch ökonomische Aktionen wie das Generieren und Anzeigen personalisierter Werbung.[1] Inwieweit poetische Explorationen Antwort auf die Frage geben können, was unter der Oberfläche beziehungsweise hinter dem Interface geschieht, wird in diesem Beitrag anhand einiger theoretischer Überlegungen zu Schriftlichkeit allgemein und zum digitalen Schreiben im Besonderen in Zusammenhang mit drei Beispielen digitaler Lyrik erörtert.

Unter digitaler Lyrik werden nachfolgend Gedichte verstanden, die mit digitalen Technologien und Medien hervorgebracht werden, wobei deren Verwendung nicht nur der Produktion mit einem Textverarbeitungsprogramm sowie der Distribution dient, sondern auch in ästhetischer Weise eingesetzt und künstlerisch reflektiert wird.[2] In den Blick genommen werden generative, teils interaktive Gedichte, die mit Computerprogrammen geschrieben

1 Vgl. Markku Eskelinen: The Four Corners of the E-lit World: Textual Instruments, Operational Logics, Wetware Studies, and Cybertext Poetics. In: *Primerjalna književnost* (Ljubljana) 36.1 (2013), 13–23, hier: 15; Jörg Piringer: datenpoesie. In: *Logbuch Suhrkamp* (2016), o. S. URL: https://www.logbuch-suhrkamp.de/joerg-piringer/datenpoesie/ (letzter Zugriff: 20.01.2022); Saskia Reither: *Computerpoesie. Studien zur Modifikation poetischer Texte durch den Computer*. Bielefeld: transcript 2003, 14.

2 Vgl. Roberto Simanowski: *Textmaschinen – Kinetische Poesie – Interaktive Installation. Zum Verstehen von Kunst in digitalen Medien*. Bielefeld: transcript 2012, 6.

werden, und dadurch auch Fragen nach der Aktivität und Agentialität von Autor-, Leser:innenschaft und Computern, die bisweilen Produzenten und Rezipienten sind,[3] aufwerfen. Zudem wird ein *code poem* vorgestellt, in dem sich »natürliche« Sprache und Programmiersprache mischen, um das reziproke Wirkungsverhältnis der »heute unauflösbar gewordene[n] Verbindung der Sprache der Literatur mit der Sprache der Maschinen«[4] zu diskutieren.

1. Generierte Schriftlichkeit – oder: Wer schreibt das Gedicht?

Computer und Programmiersprachen wurden schon in einem frühen Stadium ihrer Entwicklung für poetische Explorationen verwendet. Als eines der ersten computergenierten Gedichte gilt das Ergebnis eines von Theo Lutz mit FORTRAN geschriebenen Gedichtprogramms, das bereits 1959 unter dem Titel »Stochastische Texte« in der Zeitschrift *Augenblick* publiziert wurde.[5] Auch das Nachdenken darüber, wie sich Schreiben und Schriftlichkeit in Zusammenhang mit computerbasierten Technologien verändern, hat mittlerweile eine gewisse Tradition. So fragte etwa der Medienphilosoph Vilém Flusser in seiner Abhandlung *Die Schrift* von 1987 danach, was das Spezifische am Schreiben ist und stellte für das analoge alphabetische beziehungsweise alphanumerische Schreiben folgende, geradezu dystopische Prognose:

> »Schreiben im Sinne einer Aneinanderreihung von Buchstaben und anderen Schriftzeichen scheint kaum oder überhaupt keine Zukunft zu haben. Es gibt mittlerweile Codes, die besser als die der Schriftzeichen Informationen übermitteln. […] Künftig wird mit Hilfe der neuen Codes besser korrespondiert, Wissenschaft getrieben, politisiert, gedichtet und philosophiert werden können als im Alphabet oder in arabischen Zahlen.«[6]

3 Vgl. Jay David Bolter: Digitale Schrift. In: Gernot Grube / Werner Kogge / Sybille Krämer (Hg.): *Schrift. Kulturtechnik zwischen Auge, Hand und Maschine*. München: Wilhelm Fink 2005, 453–467, 462.

4 Christian Stein: Literatur im I als Code .Code-Poetry und die Ausführbarkeit von Litetur. In: Valentina di Rosa / Jahn Röhnert (Hg.): *Im Hier und Jetzt. Konstellationen der Gegenwart in der deutschsprachigen Literatur seit 2000*. Köln: Böhlau 2019, 237–251, 240.

5 Siehe https://zkm.de/de/werk/stochastische-texte (letzter Zugriff: 20.01.2022).

6 Vilém Flusser: *Die Schrift. Hat Schreiben Zukunft?* Göttingen: European Photography 2002 [1987], 7.

Während Flusser Schreiben weiterhin als eine kulturelle Praxis, in seinen Worten als eine »Geste« versteht, durch die Schriftzeichen in bestimmter Weise geordnet werden, um Gedanken zu übermitteln, was wiederum einen ordnenden Effekt auf die zu übermittelnden Gedanken hat,[7] bleibt die Beschaffenheit der ›neuen Codes‹ in seinen Ausführungen unbestimmt. Das heißt, er geht nicht darauf ein, dass auch Quellcodes mit einer ›Aneinanderreihung von Buchstaben und anderen Schriftzeichen‹ erzeugt werden. Hingegen verdeutlicht er, bei wem er die Kompetenz des künftigen Schreibens verortet, nämlich aufseiten der Maschinen: »[E]s ist etwas Mechanisches am Ordnen, am Reihen, und Maschinen leisten dies besser als Menschen. Man kann das Schreiben, dieses Ordnen von Zeichen, Maschinen überlassen.«[8] In diesem Zusammenhang und bezüglich seines Verständnisses von Schreiben als prozessuale Praktik des Gedanken-Ordnens können Gedichtgeneratoren als Versuche angesehen werden, die Frage zu beantworten, ob Computer tatsächlich die geeigneteren Schreiber, oder in diesem Fall: Verfasser von Gedichten, sind. Des Weiteren werden dabei auch die Fragen verhandelt, welche Inhalte die Maschinen ordnen und unter welchen Bedingungen sie dies tun.

Im einleitend genannten Beispiel »Stochastische Texte« etwa dienten unter anderem Substantive und Adjektive aus Franz Kafkas Roman *Das Schloss* als Datenbasis, aus der das Gedichtprogramm neue Texte generiert, die Worte also zu Gedichten neu angeordnet hat. Ein aktuelleres Beispiel für einen Gedichtgenerator ist Nick Montforts *Taroko Gorge* von 2009, den er mit der Programmiersprache Python 2 geschrieben hat.[9] Die Datenbasis, aus welcher der Generator Begriffe wählt, besteht aus Wortlisten, die von Montfort selbst erstellt wurden. Sie beruhen auf visuellen, verbalisierten Eindrücken seines Aufenthalts im Taroko National Park in Taiwan. Das aufgerufene Programm kann daraus eine Vielzahl an verschiedenen Gedichten produzieren und nimmt dies tatsächlich ununterbrochen vor, auch deshalb bezeichnet Montfort seine Arbeit wohl als »›unbounded‹ nature poem«.[10] Jedes Mal, wenn

7 Vgl. ebd., 10.

8 Ebd.

9 Siehe Nick Montfort: *Taroko Gorge* (2009). URL: https://www.nickm.com/taroko_gorge/ (letzter Zugriff: 20.01.2022).

10 Zitiert nach Mark C. Marino: *Critical Code Studies*. Cambridge, MA / London: MIT Press 2020, 211.

die Webseite mit dem Generator aufgerufen wird, werden neue Gedichte generiert – oder besser gesagt, ein Gedichtfluss. Denn die sukzessiv generierten Verse, deren Struktur die Anordnung der Wortarten und die strophische Form betreffend im Programm festgelegt wurde,[11] ›wandern‹ vertikal über den Bildschirm, ähnlich einem Filmabspann.[12] Nur die Begriffe aus den Wortlisten werden in randomisierter Weise in immer unterschiedlichen Kombinationen in die vorgegebene Struktur eingesetzt.

Unabhängig davon, ob die generierten Gedichte nun qualitativ hochwertiger sind als von Menschen verfasste Texte oder nicht (Antworten darauf fallen möglicherweise aufgrund subjektiver Vorlieben unterschiedlich aus), wird anhand der beiden hier kurz vorgestellten Gedichtgeneratoren Folgendes deutlich: Zwar wählt die Maschine jeweils die Worte aus und ordnet sie in den Gedichten an, allerdings geschieht dies erstens nach den Vorgaben beziehungsweise Parametern der Quellcodes, die von Verfasser:innen geschrieben und festgelegt werden, in diesem Fall von Lutz und von Montfort. Und zweitens stammen die zugrunde liegenden Wörter ebenfalls ›aus der Feder‹ (oder der Schreibmaschine oder dem Keyboard) menschlicher Akteure, hier von Kafka und Montfort. Agentialität in diesen Beispielen lässt sich mithin als Handlungsgefüge zwischen Mensch und Maschine beschreiben,[13] in dem die menschlichen Aktivitäten des Programmierens und Schreibens den Computer im Sinne einer »Handlungsinitiative«[14] in Aktion treten lassen. Natürlich

11 Vgl. ebd., 202. In seiner Publikation stellt Marino eine ausführliche Erklärung der Funktionsweise des Quellcodes von »Taroko Gorge« bereit, siehe insbesondere 210–215.

12 Da sich der vorliegende Beitrag vornehmlich auf Fragen nach der Agentialität digitaler Schriftlichkeit konzentriert, sei an dieser Stelle nur kurz darauf hingewiesen, dass die Besonderheiten der steten Bewegung, der fortlaufenden Veränderung und der Farbgebung (hier hellrosa Schrift auf grünem Grund) des Schrifttexts gewisse Herausforderungen an eine literaturwissenschaftliche Analyse stellen. Eine Untersuchung etwa der aisthetischen Effekte der vorgestellten Beispiele bedürfte eines weiteren Artikels.

13 Die Begriffe Akteur, Aktant und Agentialität werden hier in Anlehnung an die von Erhard Schüttpelz besprochene soziologische Akteur-Netzwerk-Theorie nach Michel Callon und Bruno Latour (u.a.) verwendet, die er zu einer medienwissenschaftlichen Akteur-Medien-Theorie weiterzuentwickeln sucht. Vgl. Erhard Schüttpelz: Elemente einer Akteur-Medien-Theorie. In: Ders. / Tristan Thielmann (Hg.): *Akteur-Medien-Theorie*. Bielefeld: transcript 2013, 9–67, 9f.

14 Schüttpelz wählt diesen Begriff als Übersetzung für *agency*, »weil mit diesem Wort am klarsten gesagt werden kann, dass alles das, was andere Größen in Aktion treten lässt, egal wie stark oder schwach, groß oder klein, als Ausgangspunkt (und Träger) einer

können auch diesem, als maschinellem Aktanten, Handlungsinitiativen zugesprochen werden, nicht nur, weil digitale Technologien Lyriker:innen zu poetischen Experimenten inspirieren, sondern auch weil dessen Aktivität ebenfalls weitere Aktionen nach sich zieht.

An *Taroko Gorge* ist nämlich noch eine andere Begebenheit interessant: Montforts Quellcode wurde von einigen weiteren Personen ›gehackt‹, die eigene Wortlisten erstellt haben, wodurch weitere Gedichtgeneratoren entstanden sind. Das ›Hacken‹ des Programms wird hier aus zweierlei Gründen in Anführungszeichen gesetzt: zum einen, weil Montfort seinen Quellcode selbst zugänglich gemacht hat, und zum anderen, weil der Code für die neuen Generatoren nicht verändert, sondern jeweils mit neuen Wortlisten ausgestattet wurde.[15] Die weiteren Gedichtgeneratoren sind in einer Sidebar auf der Webseite von *Taroko Gorge* gelistet, können angeklickt werden und erzeugen dann Gedichte zu unterschiedlichen, auf ihrer jeweiligen Datenbasis beruhenden Themen. Montforts digital-lyrische Arbeit kann deshalb als interaktiv oder kollaborativ bezeichnet werden, weil die User:innen die endlos produzierten Gedichte nicht nur auf dem Interface lesen, sondern auch selbst als Autor:innen aktiv werden können.

Das Schreiben von Gedichten ist in diesem Beispiel: das Programmieren eines Generators, dessen randomisierte Auswahl von Begriffen aus zuvor erstellten Wortlisten und das ›Hacken‹ des Programms als dezidiert digitale Schreibtechniken, welche die generative Schriftlichkeit gemeinsam erzeugen und bedingen. In diesem Sinne und im Gegensatz zu Flusser, der den Fokus auf Gedankenordnung und -übermittlung legt, geben die Herausgeber des vorliegenden Bands, Martin Bartelmus und Alexander Nebrig, zu bedenken, dass sich »Agentialität im Bereich der Schriftlichkeit […] durch ihre Zeichenhaftigkeit keinesfalls im Zirkulieren und Vermitteln von Informationen [erschöpft]« und verstehen Schriftlichkeit als »das Dazwischen, das kollektive Interagieren nicht-menschlicher und menschlicher Aktanten.«[16] Hinzu kommt außerdem die Frage nach dem lokalen ›Dazwischen-Sein‹ von computerbasierter Schrift: Ist diese auf dem Interface zu verorten, auf dem Schriftzeichen zu lesen sind? Oder zeigt der Bildschirm nur einen Effekt digitaler

›agency‹ (also einer Handlungsinitiative) dargestellt werden kann und soll« (Schüttpelz: Elemente einer Akteur-Medien-Theorie, 10).

15 Vgl. Marino: *Critical Code Studies*, 215f.

16 Martin Bartelmus / Alexander Nebrig: Schriftlichkeit und Agentialität der Schrift. Eine Einleitung. In diesem Band, 11.

Schriftlichkeit, deren Kern ›darunter‹ – verdeckt von der Oberfläche – im von der Maschine ausgeführten Programm liegt?

2. Poetische Quellcodes – oder: Was ist das Gedicht?

Prozesse der Schriftlichkeit scheinen in digitalen Texten also auf mindestens zwei unterschiedlichen Ebenen stattzufinden: erstens im Quellcode und zweitens im Interfacetext,[17] oder in den Worten der Kulturwissenschaftlerin Saskia Reither: »Der Text spaltet sich in einen virtuellen (Programmcode/Matrix) und einen aktuellen (im Augenblick realisierter Bildschirmtext), wobei nur beide zusammengenommen ›den‹ Text ergeben.«[18]

Jay David Bolter konstatiert diesbezüglich, dass »der Computer eine radikal neue Technik des Schreibens« begründet, wobei das »Programmieren […] das wahre digitale Schreiben«[19] ist. Mit der Formulierung des ›wahren‹ digitalen Schreibens zielt Bolter zwar auf den Unterschied von der Verwendung eines Textverarbeitungsprogramms zum Verfassen von Texten und eben dem Programmieren eines Quellcodes ab, aber es klingt noch eine weitere Ebene des Begriffs des Digitalen an: Unter Bezugnahme auf Schriften Friedrich Kittlers bemerkt der Germanist und Informatiker Christian Stein, dass es bei den Bedingungen der Produktion und Rezeption von Literatur »[i]m komprimiertesten Sinne […] um die Codierung einer Information« geht und führt weiter aus:

> »Kittler beschreibt Literatur […] als datenverarbeitende Instanz, die Daten akquiriert, prozessiert, transformiert, transportiert und speichert. Dabei bezieht er sich noch nicht auf die technischen Aufschreibesysteme, sondern auf die Literatur selbst: Literatur ist, bei allem was man sonst über sie sagen könnte, Code.«[20]

Unterschiede zwischen dem ›Code‹ analoger Literatur und den Codes von Computerprogrammen (auf Letzteres bezieht sich in diesem Beitrag die Verwendung des Begriffs »digital«) lassen sich aber in der Art der Datenverarbeitung, an den beteiligten Aktanten und, damit zusammenhängend, an der Ge-

17 Vgl. Reither: *Computerpoesie*, 14f.
18 Ebd., 17.
19 Bolter: Digitale Schrift, 462.
20 Stein: Literatur im | als Code, 238f.

richtetheit sowie der Wahrnehmung der unterschiedlichen Arten von Schrifttexten ausmachen.

Der Quellcode, der zumeist von Menschen geschrieben ist – wobei Computer durchaus eigene Codes verfassen können[21] –, richtet sich an die Maschine. Die darin enthaltenen Befehle werden von einem Compiler prozessiert und anschließend ausgeführt, wodurch der Interfacetext generiert wird. Dieser erscheint mithilfe von Pixeln (also Licht- oder Farbpunkten) auf dem Bildschirm für die Rezeption durch die User:innen. Sie sehen Schriftzeichen und Bilder, die symbolischen Ordnungen in Printmedien ähneln. Tatsächlich handelt es sich aber um Informationen in Form von Bits, also um »jene arbiträren Elemente, deren logische Manipulation die Buchstaben eines alphabetischen Textes oder die diskreten Elemente eines Bildes konstituiert.«[22] Laut Bolter besteht daher bisweilen die Auffassung, dass die für die Wahrnehmung der User:innen bereitgestellten Pixel sekundär sind, während der Code als wesentlich, also primär angesehen wird.[23] Entgegen einer solchen Hierarchisierung argumentiert der Literaturwissenschaftler und digitale Lyriker John Cayley, dass es sich dabei um verschiedene Ebenen der Wahrnehmung handelt, die durch die Struktur von neuen Medien, welche er als »textual media« bezeichnet, adressiert werden:

> »On the level plains of letters and bits, there is no radical disjuncture in the symbolic media when we cross from a region of ›executable‹ text to text ›for human consumption‹. From the human reader's point of view, they are both more or less construable strings of letters; from the processing hardware's point of view they are more or less construable sequences of voltage differences.«[24]

Das heißt, dass ›human readers‹ nicht nur den Interfacetext (also die Worte auf dem Bildschirm), sondern durchaus auch das Skript des Quellcodes als Buchstaben und Satzzeichen lesen und verstehen können, sofern sie die jeweilige Programmiersprache beherrschen. Computer interpretieren hingegen das durch den Compiler in Maschinensprache übersetzte Skript entspre-

21 Vgl. Bolter: Digitale Schrift, 462.

22 Ebd., 461.

23 Zur sogenannten Kode-These siehe ebd., 461f.

24 John Cayley: The Code is not the Text (Unless It Is the Text). In: *Electronic Book Review* (Sept. 2002), 1–21, hier: 8. Vgl. auch ebd., 1.

chend der binären Anweisungen für Stromzufuhr (1) oder -unterbrechung (0), wodurch der Interfacetext generiert wird.[25]

Idealerweise sollten daher die Textformen ›Quellcode‹ und ›Interfacetext‹ in Abhängigkeit zueinander und unter Berücksichtigung ihrer Adressierung in den Blick genommen werden, wofür sich ein Beispiel sogenannter Code Poetry anbietet. Dabei handelt es sich um Texte, die in einer Kombination aus Quellcodes sowie poetischer Sprache geschrieben und als Interfacetext zugänglich sind. Code Poems setzen entweder die syntaktischen Regeln einer Programmiersprache in ästhetischer Weise ein, ohne von Computern ausgeführt werden zu können,[26] oder sie sind zugleich Gedichte und tatsächlich ausführbare Programme. In seiner Monografie *Critical Code Studies* von 2020 erklärt Mark C. Marino diesbezüglich, dass Programmierer:innen Kommentare für sich oder für andere Programmierer:innen in den Quellcode integrieren können, welche zwar nicht prozessiert werden, die Ausführung des Programms aber auch nicht stören.[27] Solche »poegrams«[28] sind etwa auf der Webseite des Projekts *./code --poetry* der Programmierer und Lyriker Daniel Holden und Chris Kerr von 2016 zu finden, die allesamt Gedichte und ausführbare Programme zugleich sind.[29]

In dem nachfolgend besprochenen Code Poem »[by_conspiracy_or_design]« befindet sich ein ›human-readable‹ Text auf der rechten Bildschirmseite, dessen Ausführung durch den Computer auf der linken Seite zu sehen ist – in Form einer visuellen, kinetischen Repräsentation. Das Thema des Code Poems, das in der URL mit »by_conspiracy_or_design« benannt ist,[30] wird durch die Erwähnung der Chemtrails-Verschwörungstheorie im vierten Vers spezifiziert: dem Glauben, dass mit Hilfe von Flugzeugen Chemikalien versprüht werden, die je nach Auslegung der psychologischen Manipulation, der Bevölkerungsreduzierung oder der Wettermodifikation dienen sollen.

25 Vgl. Philippe Bootz: Programmed Digital Poetry: A Media Art? In: *Primerjalna književnost* (Ljubljana) 36.1 (2013), 41–60, hier: 42; Flusser: *Die Schrift*, 139.

26 Ein frühes, berühmtes Beispiel eines nicht ausführbaren Code Poems ist Larry Walls »Black Perl« (1999), dessen exit-Befehl das Programm gleich zu Beginn des Gedichts stoppen würde. Vgl. Stein: Literatur im I als Code, 242f.

27 Vgl. Marino: *Critical Code Studies*, 2.

28 Espen J. Aarseth: *Cybertext. Perspectives on Ergodic Literature*, Baltimore, MD: Johns Hopkins 1997, 11.

29 Siehe https://code-poetry.com/ (letzter Zugriff: 20.01.2022).

30 Siehe Daniel Holden / Chris Kerr: [by_conspiracy_or_design] (2016). URL: https://code-poetry.com/by_conspiracy_or_design (letzter Zugriff: 20.01.2022).

Das Programm generiert eine sich bewegende Spirale aus verschiedenen Satzzeichen, die in Zusammenhang mit dem Gedichtinhalt etwa Kondensstreifen eines Kunstflugs oder eine Luftzirkulation evozieren. Wie die Thematisierung von Verschwörungstheorien auf formaler und inhaltlicher Ebene aussieht, soll folgender Ausschnitt aus dem *poegram* verdeutlichen:

»[…] PROOF,of,chemical,dumps;
wind: across,the,'September'-blue,sky;
bending(to,the,RATIO);as:NATURE=fights_back;
using:/native-American/-wisps,that,clean-up|ALL|that,'bad stuff';
employing(a_simple_formula);the,/ancients/;!KNEW;«[31]

Neben der Großschreibung einiger Wörter, die an aufgebrachte Kommentare von Verschwörungstheoretiker:innen in Internetforen erinnern, wird die Chemtrails-Theorie hier noch mit einer anderen »Theorie« aus esoterischen Kreisen verknüpft. Durch die Hinweise auf *Native Americans* und *the acients knew* wird die Auslegung des Maya-Kalenders hinsichtlich der Vorhersage des Weltuntergangs aufgerufen. Beide Mythen werden weiterhin durch die Vorstellung verbunden, die Natur würde zurückschlagen und sich von allem Schlechten (den Menschen?) reinigen.

Aspekte digitaler Schriftlichkeit spielen hier demnach auf unterschiedlichen Ebenen eine Rolle: Das *poegram* erscheint als ein Kommentar zu Dynamiken schriftbasierter Chatforen darüber, wie derartige Theorien in digitalen Medien kommuniziert, miteinander verknüpft und verbreitet werden. Des Weiteren verdeutlicht das Exzerpt, dass die in den Quellcode integrierten Verse für die Rezeption von menschlichen Leser:innen verfasst wurden und die Kenntnis der Programmiersprache keine Voraussetzung für ihre Lektüre ist. Darüber hinaus zeigt sich das Ergebnis des ausgeführten Programms durch die visuelle Repräsentation (in Form von kinetischen Schriftzeichen), die dem Quellcode als Anweisung für den Computer eingeschrieben ist. Dabei wird die Frage nach der Verortung der digitalen Schrift elegant umgangen, denn der Text beziehungsweise das Gedicht ist Quellcode und Interfacetext in einem und auf dem Bildschirm, also auf der Oberfläche, lesbar. Auch in diesem Beispiel ist die Agentialität programmierter Schriftlichkeit als Handlungsgefüge zwischen menschlichen und maschinellen Aktanten zu beschreiben, wobei die Rezipient:innen dem Computer bei der Arbeit zusehen können. Aller-

31 Holden / Kerr: [by_conspiracy_or_design], V. 7–11.

dings kann die Funktionsweise, zum Beispiel die Frage, welche syntaktischen Regeln und Schriftzeichen des Quellcodes die Anweisungen an die Maschine enthalten, nur bei Kenntnis der Programmiersprache nachvollzogen werden.

3. Algorithmen verstehen – oder: Wer liest wen?

Während in Code Poetry die Quellcodes und die Interfacetexte in Kombination vorliegen und daher gemeinsam untersucht werden können, bestehen in der Regel diverse Probleme der Zugänglichkeit: Die Quellcodes sind für User:innen zumeist gar nicht sichtbar, sondern bleiben unter der Oberfläche verborgen und viele Seitenbetreiber:innen sowie Internetfirmen halten sich ihre Algorithmen betreffend bedeckt. In den Fällen, in denen sie doch offen zugänglich sind, kann zudem die Mehrheit der User:innen ihre Strukturen, ihr Vokabular und ihre Funktionsweisen nicht verstehen.[32] In diesem Zusammenhang gibt Marino Folgendes zu bedenken:

> »If code governs so much of our lives, then to understand its operations is to get some sense of the systems that operate on us. […] Like other systems of signification, code does not signify in any transparent or reducible way. And because code has so many interoperating systems, human and machine-based, meaning proliferates in code.«[33]

Zwei Aussagen sind an diesem Zitat beachtenswert: Zum einen wird Programmiersprachen ein bedeutungsgenerierendes Potenzial zugesprochen, das heißt, Marino nimmt Abstand von der Ansicht, dass sie sich durch Eindeutigkeit auszeichnen und nur richtig oder falsch kennen.[34] Vielmehr könnten auch Quellcodes ebenso wie Produkte anderer Zeichensysteme in ihrer Interpretation durch menschliche Akteure Polysemien aufweisen. Zum anderen werden die Codes selbst als Aktanten aufgefasst, die weite Teile unseres Lebens beeinflussen oder gar steuern. Daher sei eine *digital literacy* im Sinne des Erlernens von Programmiersprachen heutzutage unabdingbar, um unsere digitalisierte Welt verstehen zu können.[35] Auch verhält es sich so,

32 Vgl. Bolter: Digitale Schrift, 465; Cayley: The Code is not the Text, 2.
33 Marino: *Critical Code Studies*, 4.
34 Vgl. Stein: Literatur im I als Code, 239.
35 Vgl. Marino: *Critical Code Studies*, 21.

dass Quellcodes und Algorithmen nicht nur gelesen werden – vielmehr können sie umgekehrt die User:innen »lesen«, wie der Literaturwissenschaftler und Lyriker Craig Dworkin folgendermaßen auf den Punkt bringt:

> »Although we may imagine ourselves as customers consuming the product of Facebook's platform, our data, harvested as we use the site, is the actual product sold by the company; […]. Similarly, one does not so much google something, as one is googled in the process; again, Google's business model is predicated less on providing customers with data than on gathering their data as raw material. […] [C]onsumers have themselves been commodified.«[36]

Während also User:innen Interfacetexte im Internet lesen, können die in den zugrundeliegenden Quellcodes vorhandenen Algorithmen so programmiert sein, dass sie das Leseverhalten der User:innen interpretieren und darauf reagieren. Das User:innen-Verhalten wird dabei in Form von Daten gesammelt, gespeichert und ausgewertet. Auf dieser Grundlage werden unter anderem bestimmte Inhalte ausgewählt und angezeigt, allen voran gezielte Werbungen. Das jeweilige Leseverhalten beeinflusst demnach den Interfacetext, der sich je nach gesammelten Daten unterschiedlich präsentiert.[37] Derartige Prozesse laufen hintergründig ab und werden von den User:innen entweder nicht bemerkt oder deshalb hingenommen, weil die Nutzung vieler digitaler Services gegenwärtig kaum mehr zu umgehen ist. Eine ähnliche Beobachtung beschreibt der digitale Lyriker Jörg Piringer als Ausgangspunkt seines Projekts *datenpoesie* von 2016. Ihm zufolge nutzen Autor:innen heutzutage zwar mehrheitlich digitale Technologien zum Verfassen und Lesen von Texten oder zum Bewerben von Veranstaltungen, jedoch ohne deren Einfluss auf die eigenen Schreib- und Lesepraktiken zu reflektieren.[38] In seinen poetischen Explorationen interessieren ihn hingegen die Funktionsweisen von Quellcodes und Algorithmen, die er nachvollziehen und sichtbar machen will.

Beispielsweise beschäftigt sich Piringer im generativen Gedicht »allgemein erklar menschenrecht« mit sogenannten Stoppwörtern. Diesbezüglich erklärt er, dass Sprachverarbeitungsalgorithmen von Suchmaschinen wie

36 Craig Dworkin: Poetry in the Age of Consumer-Generated Content. In: *Critical Inquiry* 44 (2018), 674–705, hier: 677.

37 Vgl. ebd., 679; Eskelinen: The Four Corners of the E-lit World, hier: 15.

38 Vgl. Piringer: datenpoesie, o. S.

Google unter anderem auf Datenreduktion ausgerichtet sind, weil idealerweise alle Webseiten in einem Suchindex zusammengefasst werden sollen. Hinsichtlich der äußerst hohen Anzahl von Internetseiten muss daher Speicherplatz gespart werden. Um dies zu erreichen, entfernt der Algorithmus zunächst Stoppwörter, also Wortarten, die für eine Suchanfrage als nicht relevant erachtet werden, wie etwa Artikel, Präpositionen und Konjunktionen. Zur weiteren Vereinfachung werden die verbleibenden Wörter zudem auf ihren Wortstamm reduziert.[39]

Dieses algorithmische Vorgehen wendet Piringer auf die *Allgemeine Erklärung der Menschenrechte* an, welche somit die Datenbasis seines Gedichts darstellt. Dadurch entsteht ein seltsam anmutender Text, dessen erste drei Zeilen folgendermaßen lauten: »da anerkenn angebor wurd gleich unverausser recht mitglied gemeinschaft mensch grundlag freiheit gerechtig fried welt bildet nicht anerkenn veracht menschenrecht akt barbarei […]«.[40] Indem also Begriffe wie ›Anerkennung‹, ›unveräußerlichen‹ oder ›Gerechtigkeit‹ derart verkürzt werden, können sie mit allen Wörtern gefunden werden, die den jeweiligen Stamm enthalten. Piringers Beispiel illustriert demnach das stark reduzierte Sprach-, genauer gesagt Schriftverständnis von Suchmaschinen, die in der Regel schriftliche Texte durchsuchen. Der in den Worten des digitalen Lyrikers »verstümmelte[]«[41] Text ist darüber hinaus nicht in der Lage, den Sinn des Originaltexts wiederzugeben: nämlich die Verkündung, dass »[a]lle Menschen […] frei und gleich an Würde und Rechten geboren«[42] sind. »allgemein erklar menschenrecht« legt nicht nur eine Funktionslogik der meistgenutzten Suchmaschine offen, sondern erscheint auch als kritischer Kommentar über den reduktionistischen Umgang der Maschine mit Sprache respektive Schrift.

Da Internetfirmen und Seitenbetreiber die von ihnen verwendeten Algorithmen überwiegend geheim halten, kann nur eingeschränkt untersucht werden, wie beispielsweise Daten gesammelt, gezielte Werbung generiert oder das Leseverhalten beeinflusst wird. Allerdings können algorithmische Verfahren zumindest teilweise nachvollzogen werden, wie die poetische

39 Vgl. ebd.

40 Ebd.

41 Ebd.

42 Siehe https://www.bpb.de/internationales/weltweit/menschenrechte/38624/erklaerung-der-menschenrechte (letzter Zugriff: 20.01.2022).

Exploration Piringers verdeutlicht. Diese offenbart Aktivitäten von Programmen, die normalerweise im Hintergrund ablaufen, indem die prozessuale Schriftverarbeitung, welche generell als Teil digitaler Schriftlichkeit zu betrachten ist, im Interfacetext sichtbar gemacht wird. So konstituiert sich Agentialität in diesem Beispiel ebenfalls zwischen Mensch und Maschine, da sich Piringer die algorithmische Arbeitsweise des Computers sozusagen aneignet und mit ihr einen poetischen Text generiert, der Automatismen im digitalen Schriftgebrauch herausstellt.

4. Resümee

Die vorgestellten lyrisch-digitalen Arbeiten reflektieren Besonderheiten von programmierter Schriftlichkeit und der damit zusammenhängenden Agentialität auf verschiedene Weise: Gedichtgeneratoren veranschaulichen beispielsweise, dass die daraus entstehenden Gedichte ein Produkt menschlicher und maschineller Aktivitäten sind, da die Generatoren und Wortlisten von Menschen verfasst werden, die den Computer in Aktion treten lassen. Auch von der Maschine gehen Handlungsinitiativen aus, da sie User:innen zum kreativen Umgang mit digitalen Technologien inspirieren, die dann selbst als Autor:innen aktiv werden. In Code Poetry können hingegen die Textformen ›Quellcode‹ und ›Interfacetext‹, deren Prozesse in der Regel auf unterschiedlichen Ebenen stattfinden, in Abhängigkeit zueinander betrachtet werden. Dadurch, dass *poegrams* Gedichte und ausführbare Programme zugleich sind, deren Skript und dessen Ausführung auf dem Interface gemeinsam nachvollzogen werden können, kann der Maschine bei der Arbeit zugesehen werden. Auch in generativen Gedichten, in denen algorithmische Verfahrensweisen etwa von Suchmaschinen nachempfunden werden, können Aktivitäten von Programmen zugänglich gemacht werden. Bei der Herausstellung ihrer Funktionslogiken, die normalerweise im Hintergrund ablaufen, wird nicht nur der maschinelle Umgang mit Schrifttexten deutlich, sondern auch, dass Codes selbst als Aktanten auftreten, die das Leseverhalten von User:innen beeinflussen.

Während verschiedene Prozesse programmierter Schriftlichkeit bei der Nutzung digitaler Technologien mehrheitlich verdeckt ablaufen, vermag es digitale Lyrik demnach, diese an die Oberfläche zu bringen. Dabei werden auch digitale Schreibtechniken beleuchtet, wie das Programmieren von Quellcodes, deren Ausführung durch den Computer und damit zusammenhängend

algorithmische Verfahren, ebenso wie Fragen nach der Agentialität computerbasierter Textproduktion. Diese lässt sich mithin als komplexes Handlungsgefüge zwischen Autor-, Leser:innenschaft und Computern beschreiben, als ein gegenseitiges Einwirken menschlicher und maschineller Aktanten, durch das digitale Schriftlichkeit prozessual und kollektiv hervorgebracht wird.

***

Dieser Artikel ist Teil des Projekts *Poetry in the Digital Age*, für das Fördermittel des Europäischen Forschungsrats (ERC) im Rahmen des Programms der Europäischen Union für Forschung und Innovation »Horizont 2020« bereitgestellt wurden (Finanzhilfevereinbarung Nr. 884177).