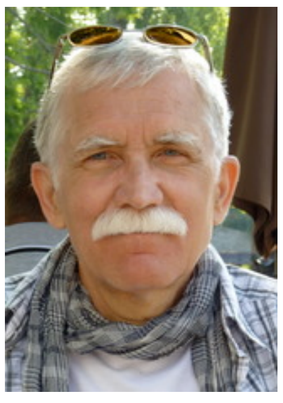
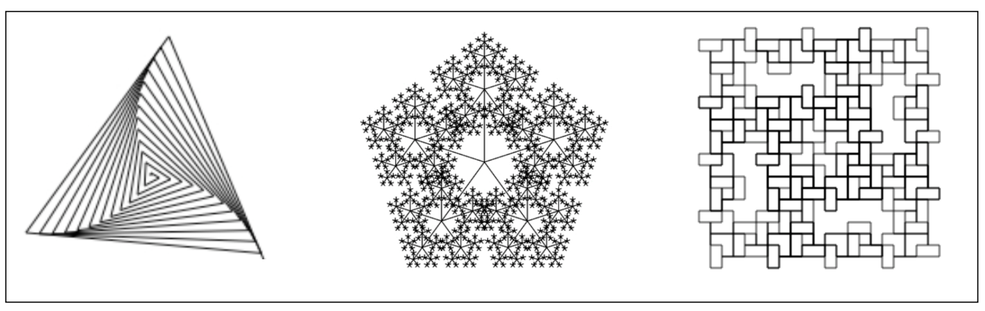
Recode. Remix. Design

Kunst Codieren durch visuelle Programmierung mit Snap!

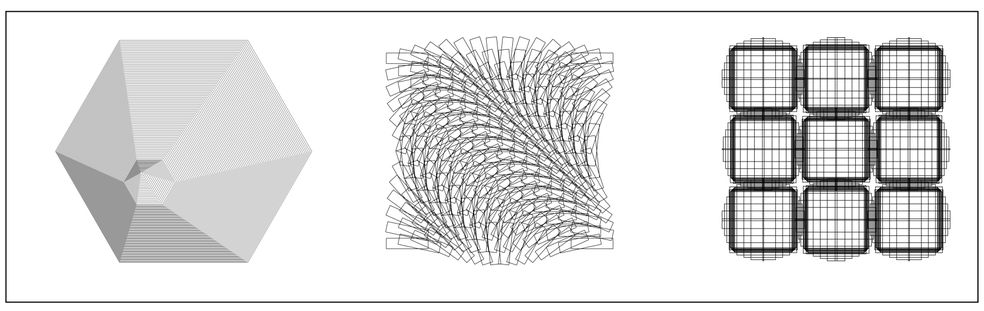
Die Ausstellung *Recode. Remix. Design* ist eine Hommage an die Pioniere der frühen Computerkunst der 60er-Jahre. Mit seinen Exponaten knüpft der Unterrichtstechnologe und Mediendidaktiker Joachim Wedekind bewusst an deren Arbeiten an. Dabei interpretiert und und transformiert er ihre Themen mit den Technologien der Gegenwart.



Bereits Anfang der 80er Jahre hat Joachim Wedekind die *Programmiersprache Logo* kennen gelernt. Für einen Studienbrief *Schildkrötengrafik* im Rahmen eines Projekts zum Thema *Lehren und Lernen mit dem Computer* entwickelte er zum ersten Mal Programme für Logo-Grafiken mit ihrer ganz eigenen Ästhetik. Bei der späteren Beschäftigung mit der frühen Computerkunst entdeckte er dann überraschende Parallelen der Bildeigenschaften:



Beispiele Schildkrötengrafik



Beispiele Computerkunst

Bei beiden finden sich typische Grafikelemente wie Linien und Polygonzüge, Quadrate, Kreise und Ellipsen, sowie Linien und Schraffuren.Die Bildeigenschaften sind jeweils geprägt durch einfache Grundelemente und deren Wiederholung und Variation sowie vom gelenkten Zufall für die Kennwerte.

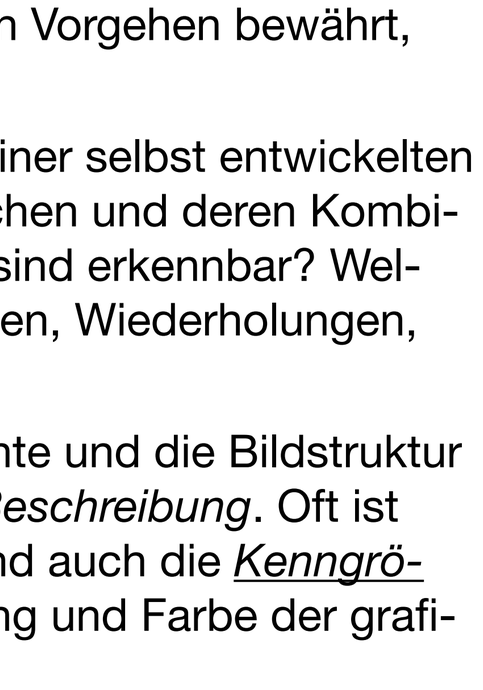
Das war die Grundlage für die *Codierte Kunst* von Joachim Wedekind, zu Beginn mit dem *Recoding* prägender Beispiele, gefolgt vom *Remixing* und schließlich dem *Design* eigener ästhetischer Objekte.

Den Begriff *Codierte Kunst* macht zweierlei deutlich: Erstens liegen ihr immer *Algorithmen* zugrunde. Zweitens müssen diese Algorithmen in einer *Programmiersprache* ausformuliert, also *codiert* werden. Leider sind die Algorithmen der frühen Computerkunst fast nie dokumentiert worden und so bleiben meist nur die Bilder selbst als Vorlagen. Für Joachim Wedekind hat sich dabei ein Vorgehen bewährt, das sich an Frieder Nakes Prinzip *Think the Image, Don't Make It!* orientiert:

Analyse Am Anfang steht immer die genaue *Analyse* einer *Bildvorlage* oder einer selbst entwickelten *Bildidee*. Welche Grundelemente sind enthalten (Punkte, Linien, Flächen und deren Kombination), welche Farben und welche sich wiederholenden Muster und Strukturen sind erkennbar? Welche Operationen werden auf sie angewendet (z. B. geometrische Transformationen, Wiederholungen, Variationen) und welche Rolle spielt der Zufall?

Algorithmus Im zweiten Schritt ist genau zu beschreiben, wie die Bildelemente und die Bildstruktur erstellt werden sollen, d. h. wir benötigen eine *algorithmische Beschreibung*. Oft ist dies einfach, in anderen Fällen kann es eine hohe Hürde sein. In dieser Phase sind auch die *Kenngrößen* zu definieren, wie die Anzahl und Positionierung sowie die Größe, Ausrichtung und Farbe der grafischen Elemente.

Implementation Gemäß dem Motto *Erst denken, dann programmieren!* ist die *Programmierung* der letzte Schritt, damit die Umsetzung von Maschinen ausgeführt werden kann. Joachim Wedekind hat als Werkzeug die *visuelle Programmierumgebung Snap!* verwendet. Das Beispiel rechts erzeugt das Bild *Schotter*, eine Ikone der frühen Computerkunst. Seine Programme stellt Joachim Wedekind als freie Lernmaterialien zur Verfügung, mit dem Dritte experimentieren können.



Vertiefende Informationen zum Thema finden Sie in seinem *Buch Codierte Kunst* (J. Wedekind, 2018) und auf der *Webseite* zur Ausstellung.

Zur Webseite

ComputingEducation

Computer science explained!

Zum Buch