研究報告

メディア表現とクリエイティブコーディング教育

高　尾　俊　介

Media Creation and Creative Coding Education

TAKAO Shunsuke

Abstract： In recent years, education-focused information technology（ICT）and media literacy educations have been increasingly important. They have also become a problem for the educational system that is directly connected to the inclusion of programming education in the compulsory subjects for elementary school students, which will be implemented in 2020. The reason why programming literacy is so critical nowadays is that programming is a fundamental principle in this information-based society. It is expected the importance of programming will grow even more in a future where AI（artificial intelligence）develops rapidly. The need for programming has been expanding quickly in the field of media expression, as discussed in this paper. A media-based environment where we are living now continues to transform at a rate never seen before in history. Various media that had been utilized as communication tools until now have been digitalized and integrated into digital devices and the internet.

In this paper, we will present the modern social conditions and the challenges in the area of media expression, in particular, IT education field. We will also describe practical examples of STEAM education and creative coding to solve those challenges.

Key Words： creative coding, STEAM education, programming, new media art

要旨：近年，教育を中心とした情報通信技術（ICT）教育・メディアリテラシー教育の必要性が叫ばれている．これは 2020 年から行われる小学校プログラミング教育の必修化の流れに接続された教育システム上の問題でもある．プログラミングリテラシーがなぜ今求められているかといえば，現在プログラムが情報社会を支える根源的な原理だからだ．AI（人工知能）が高度に発達した未来へ向けて，重要性は今後ますます高まることが予測されている．本稿で取り上げるメディア表現分野においてもプログラミングの重要性はこれまで以上に大きくなっている．私たちの生活するメディア環境そのものが歴史上なかった速度で変容し続けており，近代までコミュニケーションの道具として活用されてきたさまざまなメディアが情報化される形でコンピュータに代表されるデジタルデバイスとインターネットに統合され，置き換えられたことに起因している．

本稿では現代の社会状況と，メディア表現領域でも特に ICT 教育分野が直面する課題について述べる．それを解決するための STEAM 教育やクリエイティブコーディングの実践と課題について述べる．

キーワード：クリエイティブコーディング，STEAM 教育，プログラミング，メディア・アート

1. はじめに

2020年から始まる小学校プログラミング教育の必修化[1]と連動して，情報技術（ICT）・メディアリテラシー教育の必要性が高まっている．プログラミングを含めたコンピュータリテラシーがなぜこれまで以上に求められているかといえば，プログラムが情報社会を実装するための原理だからだ．家電製品から自動車，飛行機，人工衛星まで，私たちの身の回りの多くのモノにコンピュータが搭載され，大規模なネットワークと通信しながら動作している．コンピュータがこのように社会に不可欠なものとなった現代においてなお，機械学習や人工知能（AI）が高度に発達した未来へ向けて，人間とコンピュータをつなぐ言語としてのプログラミングの重要性はこれまで以上に高まることが予測されている．

メディア表現分野においてもプログラミングによって可能になる表現の比重は増してきているが，それは私たちの生活するメディア環境自体がかつてない速度で変容・拡大し続けてきていることが直接的原因である．同時に，多くのメディアが情報化されていく過程で，コンピュータに代表されるデジタルデバイスとインターネットへと統合され，さらに計算機能を付加されながら置き換えられることで、新しい表現の萌芽が埋め込まれ，芽吹きつつある時代とも言える．

本稿では現代の社会状況と，メディア表現領域でも特にクリエイティブコーディング教育が直面する課題について述べる．なお本稿ではプログラムを「コンピュータによって作成されたプログラム」，コードを「プログラムとして記述されたコード」として議論を進める．

2. 背　　景

1990年代後半からパーソナルコンピュータとインターネット環境が，続く2000年代以降，iPhoneに代表されるスマートフォンが急速に普及した．誰もが1台以上のデジタルデバイスを所有し，絶えず誰か／何かと通信する日常生活が当たり前となった．

情報化社会によって引き起こされた最も重要な変化は，出来事が瞬時にメディアを通じて拡散され，波及していく圧倒的速度であり，現在はフェイクニュースやSNS上でのトラブルなど個人のメディアリテラシーが求められる時代とも言える．一方，デジタル中毒と呼ばれる過剰使用の問題も提起されている．デジタルネイティブと呼ばれるインターネット登場以降に生まれた世代は，SNSなどの情報環境に順応可能と言われているが，情報格差（デジタルディバイド）は世代を越えて広がっており，教育の場でも学生間の情報格差の広がりはスマートフォンの普及が関連していると見られている．携帯性が高く常時起動が基本のスマートフォンはコンピュータ以上にパーソナルなデバイスであるものの，現在ICT分野で期待されているデータの可視化やRPA（ロボットによる業務自動化）といった新規事業分野の開発･運用の場では，今後もコンピュータに頼らざるを得ないのが現状である．

こういった格差の解消やメディア関連の知識・技能の取得を目的に行われるコンピュータリテラシー教育は通常の授業形式の他，学生の習熟度に応じた内容を主体的に学ぶことができるようなMOOC[2]やeラーニングといった，オンライン非同期型の学習支援サービスは民間企業を中心にプログラミング教育に限らず展開されている．またYouTubeに代表される動画共有サイトでも個人のクリエイターらによるチュートリアル動画が多数公開され始めている[3]．このように非同期型の教育支援環境は反転授業やアクティブラーニングとともに広がりつつある．

---

[1] 2018年3月に文部科学省が公開した小学校プログラミング教育の手引（第一版）には「プログラミング的思考」を育むこと，プログラムの動作や利点，情報社会との接続について理解すること，つまりプログラミング一般に関するリテラシー教育がその目的とされている．
http://www.mext.go.jp/component/a_menu/education/micro_detail/__icsFiles/afieldfile/2018/03/30/1403162_01.pdf（2018年10月20日 確認）

[2] MOOC（Massive Open Online Course）の代表的なものとしてCoursera，edXなどが知られている．

[3] 後述するクリエイティブコーディングに関連する活動としてはニューヨーク大学のITP（Interactive Telecommunication Program）で教鞭をとるDaniel Shiffmanが開設するチャンネル「Coding Train」がある．ここではクリエイティブコーディングに関連するトピックやチュートリアル動画が週ごとに更新されている．
http://thecodingtrain.com/（2018年10月20日 確認）

3. STEAM 教育

前述のような背景の中, 新たな教育手法として STEAM（スティーム）教育が注目されている．これは Science（科学），Technology（技術），Engineering（工学），Mathematics（数学）を統合的に学ぶ STEM（ステム）教育に，Art（芸術）を加えたものである．STEAM の前身である STEM 教育は，1990 年代にアメリカ国立科学財団（NSF）が提唱したことがきっかけとなり，2013 年にホワイトハウスが国家戦略として採用しており，現在アメリカでは幼稚園から高等学校卒業までの 12 年間の教育プログラムに組み込まれている．アメリカで STEM 教育が導入された背景には，社会全体の ICT 化によってこれまで専業とされてきた STEM 分野の知識や技能が，他分野へと接続する形で新規事業やイノベーション創出に関連する重要な能力と認められたためである．更に STEM から派生する STEAM 教育は 2013 年，ロードアイランド・スクール・オブ・デザイン（RISD）の学長であった John Maeda が提唱したもので，その中心的な理念とされるものは以下の 3 つである．[4]

1. STEM 教育の中心に Art（芸術／デザイン）を配置する．
2. すべての公教育で芸術やデザイン分野の統合を促進する．
3. イノベーションを推進するアーティストやデザイナーを雇用するように雇用主に影響を与える．

1 の芸術やデザインが本来備えている領域横断性をもって，STEM 教育における個別分野の統合を目指している点は注目に値する．STEAM 教育における芸術はプログラミングや後述のクリエイティブコーディングに代表される計算機科学と芸術の橋渡しを行うような表現分野に限らず，音楽や美術，演劇，ダンス，写真や映像のほか，建築や園芸，クリエイティブ・ライティングなど広範な表現分野を対象としている．教育活動の中で STEM 教育と創造性を接続するような実践を行うことで，3 にあるように，社会活動を通じてアートやデザイン活動の重要性を説きながら，育成されたアーティストやデザイナーが職業専門家として活躍できるように，雇用創出への働きかけをも行おうとしている．

STEAM 教育における芸術は，例えば数理による生成的表現を探求するジェネラティブ・アートのように，コードが実行されることによって生み出される表現，あるいはアプリケーション開発に関連するようなインタラクション･デザインといった「コンピュータを介する新しい表現」としてのメディア表現分野に芸術を限定してはいない．対してメディア表現は，例えば初期国内のコンピュータ・アートがからくりやエッシャーのだまし絵のような古典芸術から着想を得たように，これまでも，言うなればメディアそれぞれが持つメディア性に着目し，それに応じた表現技法によってメディアそのものを批評的に扱うことを主題に研究・制作が行われてきた分野である．また領域を越境する範囲は広く，近年では 3D プリンタやレーザーカッターといったデジタル工作機器を用いるデジタルファブリケーション分野や，生物学的見地からバイオメディアについての考察を行うバイオアートの活動も活発である．双方ともにファブラボやバイオラボというような研究所兼工房が活動の母体となり，研究活動をひろく公知・共有していくことを重視している．

もともと Art（芸術）の語源はラテン語の「アルス」にあり，さらにこれはギリシア語の「テクネ」を翻訳したものである．「テクネ」はテクニックやテクノロジーのように技術を意味し，そこには多くの実証的側面があるものだった．このような観点からメディア表現分野は芸術と科学の融合という意味で，STEM 教育とアートやデザインを接続する文脈を先立って実践してきたとも言える．さらに近年, 芸術と科学の結節点にコンピュータがあり，その間を文字通りコードが走っているという状況がある．

[4] STEM to STEAM http://stemtosteam.org/take-action/（参照 2018-10-20）

4. コンピュータとコード

プログラミングそれ自体の起こりは，1940 年代以降登場した「電子計算機」としてのコンピュータを制御することをきっかけとしている．1961 年アメリカの宇宙開発で起こった史実をもとにした映画「Hidden Figures（邦題：ドリーム）」には計算手（コンピュータ）と呼ばれる専門職が関連する複雑な計算を人力で行っている様子が史実に基づいて描かれている．主要な登場人物の一人であるドロシーは，当時最新のコンピュータ[5]の導入によって将来計算手が解雇されることを見越して，自ら FORTRAN のパンチカード式プログラミングを習得する．コンピュータを扱うためのプログラミングは計算の正確性とその検証が重要であり（プログラム上のバグか，プログラマによるコードが誤りかを切り分ける必要があるため），プログラミングには予め数学の知識が前提となっていた．事実，プログラミング教育の初期は線形代数や行列，微分積分といった数学的概念をコードに落とし込む形式での演習が一般的なものとして進められていたため，プログラミングそのものが高度に専門化された技術となっており，多くは実学的活用に限られていた．

5. クリエイティブコーディング

2004 年に刊行された「Creative Code: Aesthetics + Computation」の中で，当時 MIT のメディアラボで准教授として所属していた Maeda は, 数理に基づくビジュアルデザインに関する研究をまとめている．1999 年に Maeda はアーティストやデザイナー，学生などコンピュータ・サイエンスやプログラミングの専門的知識を持たない人々を対象に，簡潔なコードでコンピュータの持つ計算性を利用してコンピュータグラフィックスを作成できるアプリケーション「Design by Numbers」を開発した．和訳「数によるデザイン」となるこのアプリケーションは，基本理念として教育学で言う構成主義的な考え方をコーディングに導入しており，その特徴はコードの編集画面と実行結果が並置されたアプリケーション画面に見ることができる．1）コードの実行結果を表示, 2）検証し, 3）書き換える，という 3 つのサイクルを通じて，ユーザーはプログラミングを段階的に学ぶことができる．

Maeda の率いたデザイン研究グループ「Aesthetics & Computation Group（ACG）」の学生だった Ben Fry と Casey Reas が共同で開発した，Java をベースとした IDE（統合開発環境）「Processing」は，以後クリエイティブコーディング環境として派生していくその他の環境が共通して持つ重要な性質（マルチプラットフォーム，簡潔な記述，GPL ライセンスによるコミュニティベースの開発，setup と draw による基本構文の原理，色や基本図形などグラフィック描画の規格化）を備えていた点で革新性があった．その後も派生した高速・高機能に特化した「openFrameworks」やフィジカルコンピューティングのための開発環境「Arduino」といった，クリエイティブコーディングをさらに拡張する他の開発環境へ多大な影響を与えている．

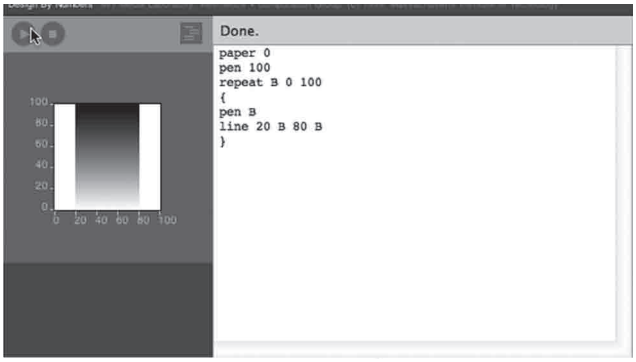

図 1　Design by Numbers スクリーンショット

[5] 大企業の基幹業務に使用される用途で開発された大型コンピュータ．ここでは IBM 7090.

これまで紹介してきたクリエイティブコーディング環境はコードベースと呼ばれる，テキストでプログラムを記述・保存する形式であるが，もう一つの潮流としてノードベースのプログラミング環境がある．これはオブジェクトとよばれるさまざまな機能を持った箱状の図形をノードと呼ばれる線で結線する形式で，GUI のコンテクストに依拠してプログラムを記述する．ノードベースのプログラミングの歴史としては 1990 年代から開発が始まっている「Max/MSP」や「TouchDesigner」などが代表的なものとして知られており，その他コードに似た制御構文のブロックを組み合わせる「Scratch」のようなものもある．これらはコードベースのプログラミング環境と比較すると高度な機能を少ない手順で利用できるなど初学者にとって利点も多い[6]．2018 年に発売された「Nintendo Labo」に同梱された「Toy-Con ガレージ」というゲームソフトでは画面をタッチして，コントローラーや本体に内蔵されたセンサーやカメラの情報を取得してプログラムを作成することができる．

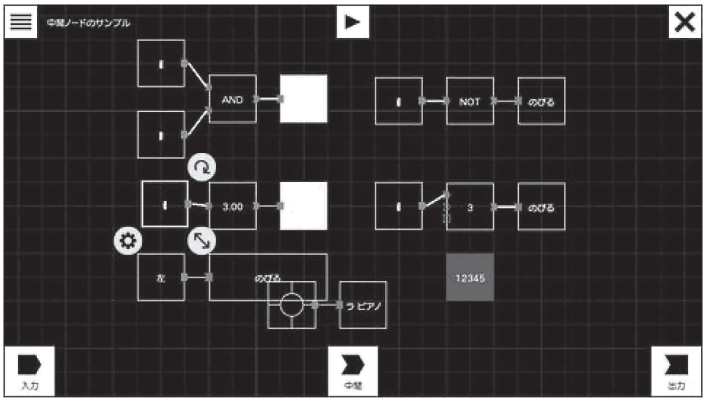

図 2　Toy-Con ガレージ スクリーンショット

その他では，コードベース，ノードベースの 2 つのクリエイティブコーディングの系譜の間を縫うかたちで，2000 年代初頭から Macromedia/Adobe が開発した「Director」や「Flash」といったタイムラインベースのオーサリングソフトを用いたインタラクティブコンテンツの隆盛があった．これらのソフトでの制作物はウェブブラウザ上で実行可能なため，親和性の高いウェブでの技術交換が盛んに行われることで，クリエイティブコーディングにコミュニティの揺籃期につながった．

数理を用いて美的な表現へと応用すること自体は，さまざまな言語や開発環境での実装や機能的翻訳が可能である．そのため，コミュニティの形成とともにクリエイティブコーディングという言葉の定義は，商業的・機能的なアプリケーションやインタラクティブコンテンツの開発手法のような合目的性を重視するものではなく，個人の表現や感覚，自由さや軽妙さに着目しながら行う創造的行為として，計算機科学の領域からコードを解放し，さらには文学的観点から捉え直すというより思想的定義へと移行したとも言える．そのため，コードベース／ノードベースの環境の違いは本質的な差異として捉えられるものではない．また，本来のソフトウェア開発であればバグとして処理されるエラーや，グリッチと呼ばれる画像や音声のようなデジタルデータを意図的に破壊することで，本来再現性が保証されているデジタルメディアに不完全性のリアリティとその美学を見いだすような試みも登場した．

6. 国内外におけるクリエイティブコーディング教育

日本におけるクリエイティブコーディング教育は国際情報科学芸術アカデミー［IAMAS］で 1999 年に行われた Max/MSP のサマースクール，2001 年の秋に武蔵野美術大学で開催された Maeda ら開発者によって行われた Design By. Numbers（DBN）や Processing のワークショップが最初期のものと考えられる．以降美術系大学でインタラクションザインやメディアアート，アルゴリズミックデザイン関連の分野を中心にクリエイティブコーディング教育が導入されるようになった．また 2000 年代中頃に起こったウェブ 2.0 以降はオンラインでの資料公開や

[6] Learnable Programming: Blocks and Beyond,https://arxiv.org/pdf/1705.09413.pdf

書籍の刊行を通じてクリエイティブコーディングに関する知識や情報が流通することとなった．特に田所淳はその最初期から，自身のウェブサイト上でクリエイティブコーディングに関する情報を公開しており，その布教から国内のコミュニティの醸成につながった．また多摩美術大学の久保田晃弘は自身の著作や多数の翻訳・監訳による刊行物を通じて，クリエイティブコーディングに関連する思想や哲学を敷衍している．

国外でのクリエイティブコーディング教育は，国内の取り組みと比較してより活発に行われている．例えばオンライン上でプログラムの公開・共有が可能なウェブサービス GitHub 上では，さまざまな大学のクリエイティブコーディングにまつわるシラバスや授業資料，サンプルコードが多数公開されており，学生のみならず教育者もその取組を参照することが可能である[7, 8]．また openFrameworks の開発者でアーティストの Zachary Lieberman が中心となって設立された School for Poetic Computing（SFPC）は，アーティストらによるコミュニティ主体の教育組織として知られている．

7. クリエイティブコーディング教育が直面する課題

クリエイティブコーディングを教育活動と接続する試みのなかでは，いくつかの課題も見つかっている．第一に学習リソースの問題がある．日本語での教育活動やそのリソースの公開・共有についてはまだまだ十分とはいえない．また小中高校と大学で目標とする学修段階とその接続なども十分に検討されていないという現状がある．プログラミング教育必修化では「プログラミング的思考」を開発することに重点が置かれることになっているが，こういった能力をどのように育てるかについて継続的に議論を進め，また効果的な学習教材の作成やその検証，公開を推進していく必要がある．

第二に，参加者間のジェンダーギャップやアクセシビリティに関する問題も挙げられる[9]．プログラミング専業化にまつわる歴史的経緯もあってか，初期段階から女性やセクシャルマイノリティといった少数者が，コミュニティへ参加するにあたってのアクセシビリティが十分でないため，参入障壁が生まれるケースがいまだ多くあり，ギャップを是正する必要がある．そのための取組みの例としてはアルゴリズムによる生成的な映像や音楽表現に関するイベントを主催している「Algorave」の活動が挙げられる．ここでは，LGBT やノンバイナリジェンダーと呼ばれる性自認を男性・女性のどちらでもない第 3 の性別と考える人々たちへ向けた，クリエイティブコーディングに関するワークショップが行われている．少数者を積極的に支援するようなワークショップや機会を設け，積極的に参入を促進することが狙いである．こういった機会創出による参入障壁を下げる方策と合わせて，さまざまなハラスメントをなくすための防止策や多様性を確保するためのガイドラインを策定していくことも重要である．例として Processing 財団が 2017 年から主催を始めたコミュニティ間の接続や活動を刺激するためのイベントである「Processing Community Day」では，技術に関連した議論と同等以上に，教育の機会や多様性についてのディスカッションが盛んに行われ，規範となるガイドラインの策定を通じて，今まで以上により健全なコミュニティ形成を目指している．

図 3 Algorave Womens' workshop Osaka の様子

[7] 代表的なものとして，カーネギーメロン大学准教授のゴラン・レヴィンはクリエイティブコーディング及びニューメディアアート分野に関する自身の講義資料を GitHub に公開している．https://github.com/golanlevin/lectures/（参照 2018-10-20）
その他 GitHub 上でのクリエイティブコーディングに関する情報資料としては以下のものがある．https://github.com/terkelg/awesome-creative-coding（参照 2018-10-20）

[8] 著者は 2018 年間に 4 回の異なるクリエイティブコーディング関連のワークショップに参加した．全てのワークショップで女性の参加者数は 20% に満たないものであった．

[9] 代表的なものとして，カーネギーメロン大学准教授のゴラン・レヴィンはクリエイティブコーディング及びニューメディアアート分野に関する自身の講義資料を Github に公開している．https://github.com/golanlevin/lectures/（参照 2018-10-20）

また，コミュニティ間の障壁として横たわる言語間の断絶も根深いものとしてある．プログラミング関連のコミュニティではドキュメントと呼ばれる仕様書は英語で記載され，ユーザーの質問や意見はオンラインフォーラムでも英語でのコミュニケーションすることが国際的な通例である．しかし前述の「Processing Community Day」は 100 以上の都市で開催されており，発表もすべて英語が基本だが，国内開催唯一の東京会場でのみ日本語での開催となっている．言語の問題は，例えば openFrameworks などのコミュニティでも同様の指摘がなされており，国内のクリエイティブコーディング関連のコミュニティが共通で抱える問題と言える．前述の教育的リソースの問題やガイドラインの策定においても，言語の問題が国際的進歩の速度と同調していく際のボトルネックとなっている．技術情報を調べ，翻訳するような局所的インプットは行われているものの，国内の活動を情報発信し共有していくといった情報発信・交換といった協調的姿勢は十分とは言えない現状がある．近い将来，言語間の即時翻訳が技術的に可能になることもありえるが，フォーラムを含めた多くの場で実装していくにはまだまだ時間がかかり，その間に国際的進歩から取り残されないためには，これまで述べてきた課題の根底に言語の問題が根ざすことを自覚し，英語での情報発信を積極的に推奨し，国際的に共有していく姿勢をコミュニティの中で育成していく必要がある．

8. ま と め

STEAM 教育そのものは，先にも述べた通り芸術をクリエイティブコーディングのようないわゆるコンピュータを用いた表現に限定して捉えていない．そのため，今後より広範な芸術分野がテクネとして機能することで，STEM が急速に教育的統合を進めていくことが予想される．他方で，クリエイティブコーディングは STEAM の先行例として表現分野における数理と創造的表現の間のさらなる結節点を探求していくための道具となるだろう．

また，発展過程でコミュニティ内の多様性やアクセシビリティが担保されることは，クリエイティブコーディングがコンピュータのデスクトップ上で達成した「個人が自由に表現を行う場」という環境を，言うなれば現実社会でも同様に実装し直すこと，と言えるだろう．その実現へ向けて，今後も研究・教育活動を続けていく．

参 考 文 献

1. Richardson, Andrew（2016）. Data-driven Graphic Design: Creative Coding for Visual Communication, London, UK: Fairchild Books.
4. Greenberg, Ira（2007）. Processing: Creative Coding and Computational Art, NY, US: Apress.
5. Reas, Casey, John Maeda（2007）. Processing: A Programming Handbook for Visual Designers and Artists, MA, US: The MIT Press.
6. Moradi, Iman, Ant Scott, Joe Gilmore, Christopher Murphy（eds）（2009）. Glitch: Designing Imperfection, NY, US: Mark Batty Publisher.
7. Bohnacker, Hartmut, Benedikt Gross, Julia Laub, Claudius Lazzeroni（ed）（2012）. Generative Design: Visualize, Program, and Create with Processing, NY, US: Princeton Architectural Press.
8. Maeda, John（2004）. Creative Code: Aesthetics + Computation, London, UK: Thames & Hudson.
9. Maeda, John（2001）. Design by Numbers, MA, US: The MIT Press.
10. Reas Casey，Chandler McWilliams（2010）. Form+Code in Design, Art, and Architecture, NY, US: Princeton Architectural Press.
11. 久保田晃弘・畠中実（2018）．メディア・アート原論 あなたは、いったい何を探し求めているのか？，フィルムアート社.
12. 久保田晃弘（2017）．遙かなる他者のためのデザイン 久保田晃弘の思索と実装，ビー・エヌ・エヌ新社.
13. 久保田晃弘（2018）．久保田晃弘：コードを記述し，実行し，保存する，情報科学芸術大学院大学紀要第 9 巻　2017 年.
14. 田所淳・比嘉了・久保田晃弘（2010）．Beyond Interaction －メディアアートのための open- Frameworks プログラミング入門，ビー・エヌ・エヌ新社.
15. William H. Press，WilliamT.Vetterling，SaulA.Teukolsky，BrianP.Flannery（1993）. ニューメリカルレシピ・イン・シー日本語版－ C 言語による数値計算のレシピ，技術評論社 .

映 像 資 料

A. Design By Numbers. 2013. https://vimeo.com/72611093（参照 2018-10-20）
B. History of the Future, Art & Technology from 1965 - Yesterday | Casey Reas | The Gray Area Festival. 2013. https://youtu.be/mHox98NFU3o（参照 2018-10-20）
C. Dan Shiffman - Codeland - Creative Coding: An art and code showcase - NYC 2017, 2017, https://www.youtube.com/watch?v=68JUaszsvmU（参照 2018-10-20）