クリエイティブコーディングにおけるユーザの内面的利益を考慮したプログラミング環境

石井 行 * 加藤 淳 † 福里 司 *

概要. クリエイティブコーディングに生成 AI 技術を活用することで，ユーザの生産性や効率性の向上が見込める．しかし，既存手法の多くはプログラムを自動生成したり自動修正したりするため，ユーザの試行錯誤や内省の機会が失われてしまう可能性がある．そこで本研究では，ユーザの内面的利益（内在的な能力，自己認識，自己内省，感情的なウェルビーイング）を維持し，その向上を支援することを目的として，生成 AI ベースの対話的インタラクションを活用する手法を提案する．具体的には，ユーザの制作途中のプログラムを踏まえつつ (1) ユーザの思考と内省を促す質問を行う対話式のチャット AI 及び，(2) 対話ログに基づく改善案を画面に提示するビジュアル機能をコードエディタに統合する．本システムにより，ユーザの認知的な負荷を無理なく増加させ，ユーザの思考を促すことで内面的利益を向上させる効果が期待できる．

1 はじめに

人間の創造的活動を促進・補助する技術は，ヒューマン・コンピュータ・インタラクション分野における重要な研究テーマである．その中でも特に，プログラミングによって画像・アニメーションなどを生成する「クリエイティブコーディング」の活動を支援するために，生成 AI 技術を導入する試みが世界的に注目されている．しかし既存研究の大半（例：Spellburst [1]）は，生成 AI によって出力されたプログラムコードをそのまま使用する，またはユーザが書いたコードを自動修正するものである．このアプローチは，生産性や効率性を向上させることはできるものの，ユーザの自己内省の機会を奪ってしまい，生成 AI への過度な依存を引き起こす危険性が存在することが，Wang ら [18] によって指摘されている．Cox ら [16] は，創造性支援ツールの評価方法がアウトプットの生産性や品質に偏重している点に加え，創造的活動を通じて得られた個人の心理的・内面的な成長こそが，人間が創造性を発揮する本質的な価値となることを論じている．また，制作における作業効率の向上は，必ずしもユーザの満足度や学習効果に繋がらない．これらの指摘は，学習プロセスにおける試行錯誤を促すための「意図的な摩擦」を重視する教育者の知見と一致するものである [14]．以上のことから，生産性や効率性だけでなく，ユーザの内面の影響（例：内在的な能力，自己効力感，自己内省，感情的なウェルビーイング）を評価することが重要であると考えられる．

そこで本稿では，クリエイティブコーディング活動におけるユーザの内在的な能力，自己効力感，自己内省，感情的なウェルビーイングを維持し，その向上を支援するために生成 AI を活用するインタラクション手法及び，これを実装したプログラミング環境を提案する（図 1）．具体的には，従来のようにプログラムコードを自動生成・修正するのではなく，(1) ユーザのコードを参照して思考や内省を促す質問を投げかけるチャット機能と，(2) 対話の文脈に基づいた改善案（修正案）を視覚的に確認できるビジュアル機能を，コードエディタに組み込んだシステムを実現した．本システムの有用性を示すために，クリエイティブコーディング経験者による本システムを用いた作品を紹介する．また，現状の課題や解決策を示し，将来への展望について述べる．

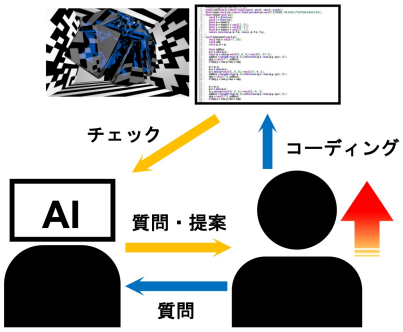

図 1. 本システムのコンセプト図．AI は直接的な「答え」ではなく，「質問」を提示することで，ユーザの思考と内省を促す．

2 関連研究

2.1 クリエイティブコーディング環境

クリエイティブコーディングを支援するための技術は，これまで多数提案されている．例えば加藤ら [10] は，音楽に合わせタイミングよく歌詞が動く

* 早稲田大学
† 産業技術総合研究所

インタラクティブな視覚表現を「リリックアプリ」と名付け，その開発支援フレームワークを提案した．また，Chugh ら [5] は，画面上の出力結果（例：図形）を直接編集し，その結果をプログラムコードに反映させるシステムを考案した．生成 AI 技術を用いた支援システムの例としては，Spellburst が挙げられる [1]．このツールでは，生成 AI 技術による出力結果をノードベースで管理・提示することでアーティストの生産性向上に成功したとされている．しかし，これらの例すべてにおいて，各技術の採用がユーザの内面に与える影響は議論されていない．ユーザ自身の内面的な利益を評価している創造性支援ツールは，Cox らによる評価手法の包括的レビュー [16] によれば，全体の 15%に留まることが報告されている．更に，McNutt ら [15] は，「クリエイティブコーディングにおける芸術的表現」と「学校教育におけるプログラミング教育」の二つの側面から，プログラミングツールが重視する要素について，学生を対象とした調査を通じて議論している．この調査では，生産性を高めるツールは，ユーザの深い理解を促すという教育上の目標とは相反しうることが示唆されている．これらを踏まえ，本研究では，生成 AI に基づくプログラムコードの自動生成ではなく，ユーザが生成 AI と対話しながらプログラミング作業を行うことで，試行錯誤しながら芸術的表現を模索できるシステムを提案する．

2.2 直接的な答え提示による弊害について

生成 AI 技術を用いた「答え」の自動提供は，ユーザの AI への過度な依存や，批判的思考（ある考えについて前提となる事実を明らかにしながら，多角的・論理的に考えること）の能力自体を低下させてしまう危険性がある [19]．更に，生成される結果（答え）は，あくまでも AI が生成できるデザイン範囲に限定されてしまうため，デザイナの想像力を制限してしまう可能性がある．そこで Danry ら [6] は，批判的思考の能力を向上させるために，ユーザに「答え」ではなく「質問」を投げかける手法を提案している．また，AI による認知負荷は「問題を解決する能力」の向上にも寄与することが指摘されている [17]．この知見を受けて本研究では，ユーザの批判的思考力や問題解決能力といった内在的能力を向上させるために，「質問」ベースの対話型 AI 機能を構築する．更に，候補提示型 UI [9][12] を参考に，本研究では，ユーザ自身のコードと対話ログの文脈情報に基づき，ビジュアル案を複数提示する機能も検討する．

2.3 創造的な活動における内省について

クリエイティブプロセスにおける「内省」は重要な要素である一方，その概念の曖昧さから創造性支援に関する研究ではあまり言及されてこなかった [3]．そこで Ford ら [7] は，内省を「現在の制作プロセスへの内省」「自己への内省」「過去の経験への内省」「実験を通じた内省」の 4 種類に分類し，評価尺度を提案している．この分類に沿うと，芸術的表現と計算論的思考を要するクリエイティブコーディング活動では，自己への内省（芸術的表現）とプロセスへの内省（コーディング）の両方が重要となる領域である．換言すれば，「表現したいイメージ」と「それをどうコードに落とし込むか」を両立させるための支援が必要と考えられる．そこで本研究では，従来研究で取り組まれてきたコーディング支援だけでなく，芸術的表現の面にも焦点を当て，とくに「自己への内省」を促す対話方法を検討する．

図 2. 提案ユーザインターフェースのスクリーンショット：(a) コードエディタ，(b) 描画パネル，(c) チャットパネル，(d) ビジュアル候補提示パネル．

3 ユーザインタフェース

本システムは，クロスプラットフォームアプリケーションフレームワークである Qt（バージョン 6.8.3）を用いて，シェーダ言語 OpenGL Shading Language（GLSL）向けのクリエイティブコーディング環境として実装している．対話式のチャット AI には，Google の提供する Gemini 2.5 flash の API を用いた．開発したユーザインタフェースを図 2 に示す．

画面構成としては，標準的なクリエイティブコーディングの環境（コードエディタ，描画結果をプレビューする画面）に加え，AI との対話を行うための

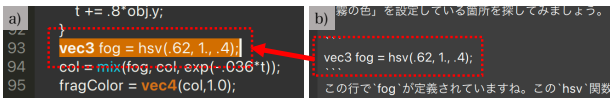

図 3. AI の返答によるプログラムコードのハイライト機能．(a) コードエディタ, (b) チャットパネル．

チャットパネルが常に表示されている．ビジュアル提案機能については，ユーザが必要な時に呼び出すことができる．

3.1 コードエディタ

従来のコードエディタと同様に，ユーザは GLSL コードを入力したり編集したりできる（図 2(a)）．ユーザが入力したプログラムコードは, 文字単位でコンパイルされ，描画パネルに結果が表示される．コードにエラーがある場合，エラーメッセージが表示される．このように，明示的なコンパイル操作不要のライブプログラミング [2][11] を実現している（図 2(b)）．

3.2 チャットパネル

ユーザは，制作途中の作品について，対話式のチャット AI と会話することができる（図 2(c)）．AI への入力には, ユーザのチャット入力とエディタ上の GLSL コードに加え, 過去の対話履歴が含まれる．さらに, AI の振る舞いを規定するために, 以下のような役割とルールを定義したシステムプロンプトを付与している．

- **目的:** ユーザの自己内省や技術的試行, 概念の探求などを促す短い質問を返す.
- **役割:** クリエイティブコーディングにおける「思考/表現のパートナー」として振る舞う.
- **ルール:** 新たなコードスニペットを提示せず, あくまでユーザ自身の思考を促すことに徹する.

このプロンプト設計により, AI は単に答えを提示するのではなく, ユーザの内省を支援するパートナーとしての対話を実現する．また，AI 側がエディタ上のプログラムコードに言及した際，該当する範囲をハイライト（橙色）する機能を有している（図 3）．ハイライト機能を用いることで，チャットパネルとエディタパネルを確認する際の外在的認知負荷を減らすことができる．

3.3 ビジュアル提案機能

チャットボット右部のボタンをクリックすると，現在のコードを基にした改善案が三つ生成され，その出力結果のビジュアルがチャットパネルの下部に表示される（図 2(d)）．これらの改善案を生成するために，AI に対して「現状のコードに対して，対話ログから要約したユーザの意図を反映した完成形のコードを，3 つのバリエーションで生成してください」というシステムプロンプトを与えている．このプロンプトによって，AI は大規模言語モデル（LLM）が有する応答の確率的な性質を基に多様な提案を行う．ユーザが表示されたビジュアル候補の中から一つの案を選択すると，コードの差分を解説する．解説は,「ユーザの現在のコードと選択された候補のコードとの技術的な差分を分析し，どのような変更が加えられたのかを自然言語で簡潔に説明してください」というシステムプロンプトにより生成される．但し, エディタパネル上のプログラムコードを自動的に更新する機能はなく，ユーザ自身が更新する必要がある．これは，ユーザの主体性を担保しつつ，AI とユーザのビジュアルを介した共通理解を確立することを目的としている．

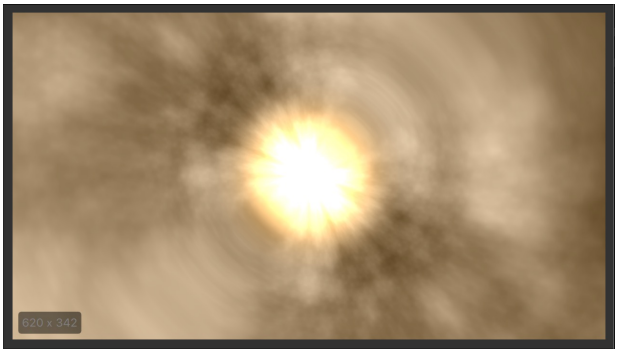

図 4. アイデア出しの段階から本システムを用いたアニメーション作品の例（作品テーマ：幼少期の心象風景）．制作時間：60 分，AI への質問回数：7 回，ビジュアル提案機能の使用回数：1 回.

4 本システムを用いた制作事例

本章では，本システムを用いた制作事例を 2 つ，その制作過程に着目して紹介する．なお，本稿で示す各事例の制作者（第一著者）は，2 年程度の GLSL によるコーディング経験と，p5.js や Processing を用いたクリエイティブコーディング経験を有する．更に C，C++，Python，JavaScript を用いたツール及びゲーム制作経験を有する．

4.1 事例 1：アイデア出しからの制作

図 4 は，アイデア出しの段階から本システムを用いて制作したアニメーション作品（作品テーマ：幼少期の心象風景）[1]である．本事例を通して，制作の初期段階から作品を完成させるまでのプロセスを示す．

4.1.1 初期段階におけるアイディア出し

制作したい作品や目的が十分に定まっていない場合において，AI との対話機能を「アイデア出し」

[1] https://www.shadertoy.com/view/WcBcz1

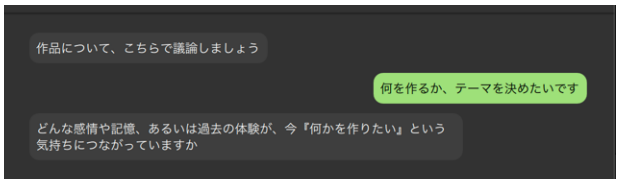

図 5. AI による質問の一例（どんな感情や記憶，あるいは過去の体験が，今「何かを作りたい」という気持ちにつながっていますか？）．

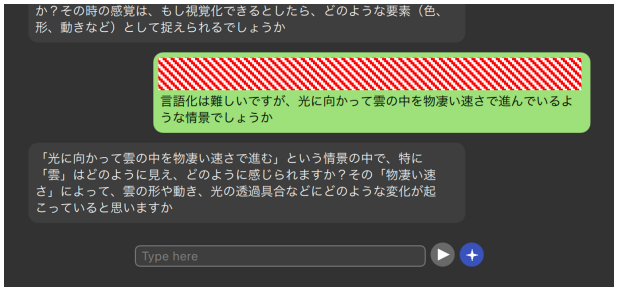

図 6. ユーザの過去や記憶を基にアイディア出しを行った場合（赤枠：プライバシにかかわる内容）．

ツールとして利用することができる．但し，本システムは，ユーザの内省を促すことを目的としているため，AI は作品テーマを直接指示するのではなく，あくまでもユーザの内省を促すような「質問」を提示する．本事例では，AI は「どんな感情や記憶，あるいは過去の体験が，今『何かを作りたい』という気持ちにつながっていますか?」といった質問を提示した（図 5）．このようなユーザの過去の経験や記憶に基づく内省を促す際には，プライバシや心的外傷（トラウマ）に関わる情報に触れる可能性がある点に留意する必要がある（図 6）．

また，ユーザが AI のテキストベースの返答だけで作品の具体像を描けない場合は，ビジュアル提案機能を用いることで新たな着想を得ることもできる．本事例では，図 7(a) のような選択肢が提示された．そして，選択肢をクリックすると,「‘hsv‘関数と ‘mix‘関数を組み合わせることで，fbm_val の値に応じて色相が 0.08（赤みがかったオレンジ） から 0.16（オレンジがかった黄色）の間で滑らかに変化する，暖色系のグラデーションが生成されるようになっています.」という解説が提示された（図 7(b)）．コードそのものが自動で書き換わることはないため，ユーザは解説をヒントに，表示された解説を参考にしながらコーディングを始めることができる．

4.1.2 ユーザと AI による試行錯誤

目的がある程度明確になったら，通常のコーディング作業と同様，エディタパネルにてコーディング作業を行う．ユーザが入力したプログラムコードは

図 7. ビジュアル提案機能の利用例．(a) ビジュアル候補，(b) ユーザのコードと選択肢のコードの差分の解説．

リアルタイムにコンパイルされ，描画パネルに結果が表示されるため，ユーザは試行錯誤しながら作品を制作できる．コーディングの途中で不明点が生じた場合，ユーザはチャット機能を通じて AI に質問を投げかけることができる．本事例では「theta が 2PI から 0 に変わる境界に線が入ってしまいます．滑らかに繋げるにはどうすればいいですか？」というユーザの具体的な質問に対して，AI が「theta の値が 0 から 2PI の間を巡るという『周期性』は，どのような図形に例えることができるでしょうか？その図形上の連続的な位置を，fbm の 2 次元入力として表現できるとしたら，どのようにすれば『離散的な切り替わり』を避けて，なめらかに繋げられると思いますか」という返答を提示した（図 8）．この返答を基に，ユーザは不明点の解決を行うことができる．このように, AI が問題解決に至る思考を促す質問を返答することで，ユーザは不明点を解決するための思考を促され，自己効力感の向上が期待できる．

本作品のアイデア出しの段階から，ユーザが実際にコーディングを開始するまでに約 20 分, その間に AI からは 3 回の質問が提示された．本システムを用いることで，ユーザは従来のコーディング作業と比べ，自己への内省を反映した作品を制作することができ，満足感が得られた．最終的に，本作品の制作時間は 60 分程度，AI への質問は 7 回，ビジュアル提案機能の使用は 1 回であった．

4.2 事例 2：既存作品の改善

本節では, 既存作品の改善において本システムを活用した事例を紹介する．図 9 は, ユーザが作成した途中段階のプログラムコード（作品テーマ：Struggling Structure）のパラメータ調整に本システムを用いたものである．途中段階の作品（図 9(a)）では，コードの各所にアニメーションを制御するパラメータが偏在する．つまり制作者は美的感覚に基づいてパラメータ調整をしなければならず，作品テーマを十分

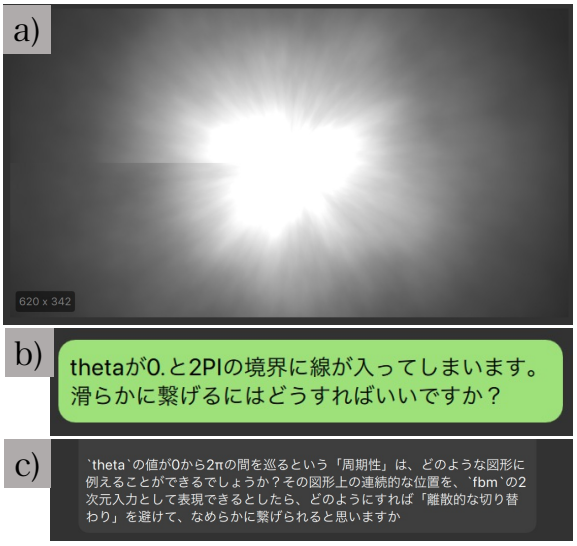

図 8. ユーザと AI による試行錯誤の様子．(a) 制作途中の描画パネル，(b) ユーザによる質問，(c)AI による返答．

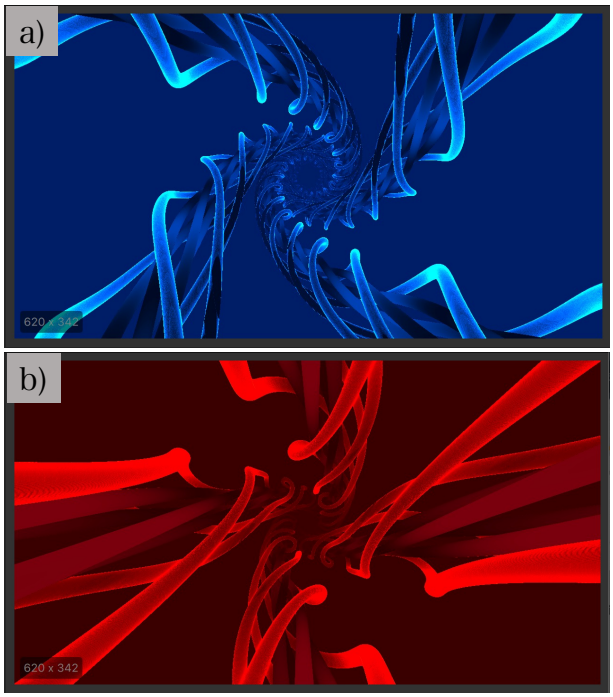

図 9. 本システムを用いたアニメーション作品の例（作品テーマ：Struggling Structure）．(a) ユーザによる制作結果（途中段階），(b) 本システムを用いて改善した結果．制作時間：20 分，AI への質問回数：5 回，ビジュアル提案機能の使用回数：2 回．

に表現できていない状態にあった．

本システムを用いることで，4.1.1 と同様にユーザの内省に関する対話ログが作成される．本事例でユーザは「もがき苦しむような不安定な動き」をイメージしていると AI に伝えた．この対話ログを基に，AI は具体的なコード箇所 $p.xy *= rot2(\ldots)$ を指摘し，「『不安定さ』や『もがき苦しむ』ような感覚をこの回転に加えるには，この cos 関数の引数や係数にどのような調整を加えることが考えられるでしょうか？」といった質問を提示した．これにより，ユーザはアニメーションの動きを制御するパラメータの調整を意図的に行うことができた．このように，本システムはユーザの内省に基づいた試行錯誤を補助する．

本作品の改善（図 9(b)）に 20 分程度を要した．AI への質問は 5 回，ビジュアル提案機能の使用は 2 回であった．

5 議論

本章では，システムを実用した制作事例を通して得られた知見をまとめ（5.1 節），クリエイティブコーディングにおいてユーザの内面的利益を考慮したシステムを設計する上で重要と考えられるデザイン指針（5.2 および 5.3 節）や，今後必要とされる研究について議論する（5.4 節）．

5.1 クリエイティブコーディングにおけるユーザの内面的利益に関する考察

関連研究でも触れたように，クリエイティブコーディングにおいては「表現したいイメージ」と「それをどうコードに落とし込むか」の両立が課題である．「表現したいイメージ」だけを深く考えてもコーディングが難しくなる一方，「コーディング」だけ考えても表現の幅が狭まってしまう．

本システムを使った際（特にアイデア出しの段階）には，チャット機能が，コードによる実現性とは無関係に内省を促す問いを提示する．そのため，第一著者による制作事例においては，通常のコーディング時よりも深く内省を行った実感があった（例：幼少期の心象風景）．その後，AI はイメージをどうコードに落とし込むかを考えるためのサポートとなる問いを提示した（例 1：「もし視覚化できるとしたら，どのような要素（色，形，動きなど）として捉えられるでしょうか？」という抽象的な問いや，例 2：「theta の値が 0 から 2PI の間を巡る ...」という具体的な問い）．第一著者の普段行うクリエイティブコーディングでは，自身のコーディング技術によって，どのような表現が可能かを探求するケースが多かった．その一方，本システムで自身の通常の発想の外にある問いを新たに提示してもらったことで，従来であれば選択肢に入らないような実装方法を発見できた．このように，本システムにより，先行する「表現したいイメージ」と「それをどうコードに

落とし込むか」というギャップを埋める過程に取り組みやすく，また，コーディングに関する学びが得られたようにも感じられた．

一方で，クリエイティブコーディングでは，アイデア出しが終わった後，単線的なコーディング作業だけではない点に注意が必要である．例えば,「表現したいイメージ」が明確な時でも，多くの場合，完成像を写真のように思い浮かべているわけではない．そのため，コーディング作業を通して予想外の新しい効果を発見し，表現の幅が広がるというセレンディピティを体験することがある．これも，クリエイティブコーディングの醍醐味である．しかし．内省した内容を直接プロンプトとして生成 AI を用いた場合，たとえイメージに近い結果が出力されたとしても，そこからユーザ自身が試行錯誤できる範囲が限られてしまう上に，作品への所有感が薄れてしまう危険性がある．本システムの場合は，自分で理解している感覚を持ちつつコーディングを行うことができたため，通常のコーディングと同じ水準で試行錯誤できたという実感が得られる．

このように，本システムは，生成 AI を単なる自動化の手段としてではなく，ユーザの内省を支え，ユーザ自身のコーディングに関する理解を助けるツールとして提供することで，ユーザの主体性を犠牲にせず創作過程の試行錯誤を支援できる可能性を示したと言える．

5.2 ユーザの内省支援に関する示唆

- **対話履歴の視覚化による内省支援機能**：AI との対話履歴をタイムラインやマップとして視覚化し，ユーザが自身の思考プロセスを客観的に振り返り，メタ認知を促す機能を実装する．
- **対話文脈への復帰を支援する UI 設計**：本システムを利用する際，ユーザが AI との対話をしている途中でコーディング作業を再開する様子が繰り返し観察された．そうした場合には，対話の文脈が飛躍してしまうケースがある．そこで，過去の対話コンテキストへ円滑に復帰できるように，未解決の対話スレッドを管理するボタンなど，AI との対話にスムーズに戻れる UI 機能を構築する．
- **外在的認知負荷を軽減する補助機能**:試行錯誤の過程で生まれる複数のプログラムコードのバージョンを管理・比較する機能をはじめ，コーディング作業以外の負荷（外在的認知負荷）を軽減するための補助機能を実装する．

5.3 AI との対話設計に関する示唆

- **自己効力感を維持するための解答提示機能**：本システムは，ユーザ自身の思考によって問題を解決すること（自己効力感の向上に繋げること）を目的としているため，AI からは直接的な答えを提示せず，常に「質問」を投げかけるような形式となっている．しかし，いつまでもユーザが自力では解決できない状況において，答えを提示しないことが「学習意欲の低下」を招いてしまう危険性がある．そこで，一定時間思考しても解決に至らない場合に，ヒントや解答を段階的に提示する機能を設計する．
- **AI による能動的な介入機能**：ユーザのコーディング作業の停滞を検知し，ユーザからのテキスト入力なしに AI から質問や提案を行う機能を実装する．
- **ビジュアルアーティファクトへの対応拡張**：logomotion [13] のように，制作した画面結果を直接認識し，それに基づいた改善案やサポートを行う機能を実装する．

5.4 今後の展望

- **長期利用におけるユーザ行動の分析**:本システムは, ユーザの試行錯誤と内省を支援する機能を有しているが，実際にどのような観点で内面的利益がどの程度向上するのかについては，第一著者による制作事例を通して定性的に検証したのみである．今後，ユーザ実験を通して，各機能が自己効力感の向上や内省の増加へどのように寄与しうるか，より詳細に，Creativity Support Index [4] なども併用しながら長期的に評価する必要がある．

  なお，各機能はユーザの思考スキル向上を促すことを目的としているため，ユーザが本システムを将来的に使用しなくなることは，ユーザの思考スキルが十分に発達し，本システム自体への依存から脱却したことを意味し，必ずしもネガティブな結果とはいえない．そこで，長期的評価においては，ユーザのスキル，モチベーション，そして利用継続・離脱の動向を分析し，思考支援 AI とユーザの理想的な関係性を探求することが必要不可欠であろう．
- **内省を分析するツールとしての展開**：近年，学習や創造の過程における内省を評価するために質問票ベースの研究 [8] が多数発表されている．本システムは，ユーザと AI との対話を通じて内省の過程がログとして詳細に記録できるため，クリエイティブコーディングにおける学習者の内省プロセスを解明するための分析ツールとしてシステム展開が期待される．
- **多言語への対応**:現在は GLSL 言語に限定されているが，今後は Processing や，その JavaScript 版である p5.js にも対応する予定である．

謝辞

本研究の一部は，JSPS 科研費 JP24K20910 及び JST ACT-X JPMJAX22A3 の支援を受けたものである．

参考文献

[1] T. Angert, M. Suzara, J. Han, C. Pondoc, and H. Subramonyam. Spellburst: A Node-based Interface for Exploratory Creative Coding with Natural Language Prompts. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology (UIST)*, pp. 100:1–100:22, New York, NY, USA, 2023. ACM.

[2] S. Burckhardt, M. Fahndrich, P. de Halleux, S. McDirmid, M. Moskal, N. Tillmann, and J. Kato. It's Alive! Continuous Feedback in UI Programming. In *Proceedings of the 34th ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI)*, pp. 95–104, New York, NY, USA, 2013. ACM.

[3] L. Candy. *The Creative Reflective Practitioner: Research Through Making and Practice*. Routledge, 2019.

[4] E. Cherry and C. Latulipe. Quantifying the Creativity Support of Digital Tools through the Creativity Support Index. *ACM Trans. Comput.-Hum. Interact.*, 21(4), June 2014.

[5] R. Chugh, B. Hempel, M. Spradlin, and J. Albers. Programmatic and Direct Manipulation, Together at Last. In *Proceedings of the 37th ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI)*, pp. 129–144, New York, NY, USA, 2016. ACM.

[6] V. Danry, P. Pataranutaporn, Y. Mao, and P. Maes. Don't Just Tell Me, Ask Me: AI Systems that Intelligently Frame Explanations as Questions Improve Human Logical Discernment Accuracy over Causal AI explanations. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems (CHI)*, pp. 352:1–352:13, New York, NY, USA, 2023. ACM.

[7] C. Ford and N. Bryan-Kinns. Towards a Reflection in Creative Experience Questionnaire. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems (CHI)*, pp. 277:1–277:16, New York, NY, USA, 2023. ACM.

[8] C. Ford, A. Noel-Hirst, S. Cardinale, J. Loth, P. Sarmento, E. Wilson, L. Wolstanholme, K. Worrall, and N. Bryan-Kinns. Reflection Across AI-based Music Composition. In *Proceedings of the 16th Conference on Creativity & Cognition (C&C)*, pp. 398–412, New York, NY, USA, 2024. ACM.

[9] T. Igarashi and J. F. Hughes. A Suggestive Interface for 3D Drawing. In *Proceedings of the 14th Annual ACM Symposium on User Interface Software and Technology (UIST)*, pp. 173–181, New York, NY, USA, 2001. ACM.

[10] J. Kato and M. Goto. Lyric App Framework: A Web-based Framework for Developing Interactive Lyric-driven Musical Applications. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems (CHI)*, pp. 124:1–124:18, New York, NY, USA, 2023. ACM.

[11] J. Kato, T. Nakano, and M. Goto. TextAlive: Integrated Design Environment for Kinetic Typography. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, pp. 3403–3412, New York, NY, USA, 2015. ACM.

[12] Y. Koyama and M. Goto. BO as assistant: Using Bayesian Optimization for Asynchronously Generating Design Suggestions. In *Proceedings of the 35th Annual ACM Symposium on User Interface Software and Technology (UIST)*, pp. 77:1–77:14, New York, NY, USA, 2022. ACM.

[13] V. Liu, R. H. Kazi, L.-Y. Wei, M. Fisher, T. Langlois, S. Walker, and L. Chilton. LogoMotion: Visually-Grounded Code Synthesis for Creating and Editing Animation. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems (CHI)*, pp. 157:1–157:16, New York, NY, USA, 2025. ACM.

[14] A. M. McNutt, S. Cohen, and R. Chugh. Slowness, Politics, and Joy: Values That Guide Technology Choices in Creative Coding Classrooms. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems (CHI)*, pp. 54:1–54:16, New York, NY, USA, 2025. ACM.

[15] A. M. McNutt, A. Outkine, and R. Chugh. A Study of Editor Features in a Creative Coding Classroom. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems (CHI)*, pp. 363:1–363:42, New York, NY, USA, 2023. ACM.

[16] S. Rhys Cox, H. Bøjer Djernæs, and N. van Berkel. Beyond Productivity: Rethinking the Impact of Creativity Support Tools. In *Proceedings of the 2025 Conference on Creativity and Cognition (C&C)*, pp. 735–749, New York, NY, USA, 2025. ACM.

[17] J. Sweller, J. J. G. van Merrienboer, and F. G. W. C. Paas. Cognitive Architecture and Instructional Design. *Educational Psychology Review*, 10(3):251–296, 1998.

[18] A. Wang, Z. Yin, Y. Hu, Y. Mao, and P. Hui. Exploring the Potential of Large Language Models in Artistic Creation: Collaboration and Reflection on Creative Programming. *arXiv preprint arXiv:2402.09750*, 2024.

[19] A. Woodruff, R. Shelby, P. G. Kelley, S. Rousso-Schindler, J. Smith-Loud, and L. Wilcox. How Knowledge Workers Think Generative AI Will (Not) Transform Their Industries. In *Proceedings of the CHI Conference on Human Factors in Computing Systems (CHI)*, pp. 641:1–641:26, New York, NY, USA, 2024. ACM.