N-016

プログラミング教育支援のための大規模言語モデルを利用した音声からの説明図生成
Generation of Explanatory Diagrams from Speech Using Large-Scale Language Models to Support Programming Education

湊 敢太郎[1)]　山崎 禎晃[2)]　大原 剛三[2)]
Kantaro Minato　Tomoaki Yamazaki　Kouzou Ohara

1　はじめに

昨今，デジタル化の進展により産業全体における情報技術の重要性が高まり，プログラミングスキルを持つ人材の需要は年々増加している．日本でも 2020 年度から小学校でプログラミング教育が必修となり [7]，教育現場におけるプログラミング指導の重要性が高まっている．

しかし，プログラミング教育の現場では，教員や TA が学習者の個別質問に対応する際，双方がもつ知識や思い描くイメージの違いにより口頭による説明だけでは意思疎通が困難な場面が存在する．特に，配列のインデックスエラーやポインタの参照関係，メモリ管理などの抽象的概念を言葉だけで説明することは極めて困難である．こうしたプログラミング学習で必要とされる特有の知識の存在は，学習者のプログラミングスキルの習得の妨げとなりやすい．たとえば，実行時エラーや論理エラーが発生した場合，処理過程や変数の扱いを理解している必要がある [9]．しかし，学習者がコンピュータの動作を「ブラックボックス」として認識してしまう事で，主体性が抑制されてしまう例が報告されている [8]．そこで，プログラミングの脱ブラックボックス化を図る方法の一つとして，プログラミング概念の視覚化が研究されている．

プログラミング概念を視覚化する多くの研究では，プログラムの動作理解の促進を目的とし，そのための視覚化ツールを提案している [2]．Python Tutor[5] や VisuAlgo[4] のような視覚化ツールは，プログラムの実行過程を段階的に可視化することで，学習者の理解を助けている．しかし，これらのツールは事前に定義された教材や一般的なアルゴリズムの視覚化に長けており，授業中に発生する個別的な質問や学習者固有のエラー状況に対する即時対応は困難である．たとえば，学習者が配列の初期化について質問した場合，既存の視覚化ツールでは一般的な配列の説明しか提供できず，その学習者が抱える固有の質問や理解度に即した説明図を即座に生成することはできない．そのため，実際の教育現場で発生するさまざまな質問に対応するためには，質問内容や問題に応じた個別の説明図を生成することが必要であると考えられる．

近年では，プログラミング教育支援において，大規模言語モデル（Large Language Model: LLM）を活用する動きがある [1]．LLM はコード生成やエラー解析の性能も高く，自由応答が可能であるため，学習者や問題に応じた教育的フィードバックの生成が可能である [3, 6]．しかし，これらの研究ではテキストによるフィードバックを基本としており，プログラミング概念の説明図を視覚化する試みはなされていない．

1）　青山学院大学大学院理工学研究科
2）　青山学院大学理工学部

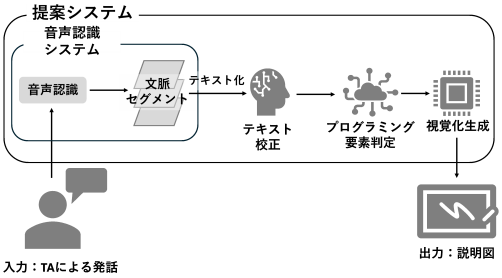

図 1　提案手法の概要図

そこで本研究では，口頭で説明される専門的な概念の理解を助けるために，指導員の音声説明から説明図を自動生成するシステムの構築を目指す．本システムは，音声認識，校正，図示生成の 3 つのコンポーネントで構成され，指導員の発話音声からプログラミングに関する文を検出し SVG 形式の説明図を生成する．実験では，C 言語のプログラミング実習授業で収集した対話音声に対して，口頭説明での視覚化が必要な事例に対して提案システムの有効性を検証した．

2　提案手法

提案システムは，音声認識，校正，説明図生成の 3 つの主要コンポーネントから構成される．本システムでは，まず教員の発話音声（例：「配列のインデックスが範囲外になってエラーが出ています」）をリアルタイムで取得し，音声認識コンポーネントによってセグメント単位でテキストに変換する．次に，校正コンポーネントが音声認識の誤りを修正し，「配列のインデックスが配列サイズを超過したため範囲外エラーが発生」のように口語表現における曖昧な表現を修正し，説明図の生成に適した文章に整形する．最後に，説明図生成コンポーネントが校正済みテキスト内のプログラミング用語の存在を判定し，検出された場合は説明構造を分析して SVG 形式の説明図を自動生成する．各コンポーネントの役割は次のようである．音声認識コンポーネントは，プログラミング用語独自の表現による認識精度の低下によって発生する誤ったセグメントの分割に対処するために，複数のセグメントを意味的にまとまった文脈セグメントに集約する．校正コンポーネントは，音声認識で発生する専門用語の誤認識，口語で生じる冗長な表現の削除するために LLM を用いて校正を実施する．最後に説明図生成コンポーネントでは，説明に対して図の生成が必要であるかを判定し，テキストによる説明に基づいてプログラミング概念の説明図を生成する．図 1 に提案手法の概要図を示す．

2.1　音声認識システム

音声認識システムでは，教員の連続的な音声説明を処理するため，以下の手順で音声をテキストに変換する．まず，マイクから取得した音声を発話の切れ目でセグメ

表 1　視覚的要素の説明と特徴

項目	説明
視覚的階層性	重要な概念を強調し，関連する要素を適切にグループ化
情報の流れ	説明の順序を考慮した要素の配置
視認性	適切な余白とコントラストの確保

ントに分割する．たとえば，「配列のインデックスが範囲外になって」「エラーが出ています」のように，連続する発話を時間軸で区切る．各セグメントに対して音声認識を実行し，「配列のインデックスが範囲外になって」のようにテキストに変換する．認識されたテキストは TextBuffer に蓄積され，3 つのセグメントが集まると 1 つの文脈として統合される．これにより，「配列のインデックスが範囲外になって エラーが出ています」という断片的なテキストを「配列のインデックスが範囲外になってエラーが出ています.」のような連続した文章として校正システムに送る．この処理により，教員の長時間の説明を適切な単位で区切りながら，最低限の文脈を保持したテキスト変換が可能となる．

2.2　校正システム

音声認識システムでは，プログラミング用語の誤認識や口語的表現により，「はいれつ」→「配列」の誤変換や「えーっと，その，配列のサイズが」のような冗長な表現が生成される．校正システムでは，これらの問題を解決するため，音声認識結果を教育的説明に適した形式に変換する．

たとえば，音声認識結果「えーっと，はいれつのインデックスが，その，範囲外になってエラーが出てます」を「配列のインデックスが配列サイズを超過したため範囲外エラーが発生」のように，プログラミング用語を正規化し，簡潔な文章に整形する．

具体的には，LLM を用いて以下の処理を実行する：

- 専門用語の正規化: 音声認識で誤認識されたプログラミング用語（「はいれつ」→「配列」）を正しい表記に修正する．
- 口語表現の整理:「えーっと」「その」などの冗長表現を除去し，簡潔な説明文に変換する．
- 教育的構造化: 問題提起から解決策提示までの論理的な流れを明確化し，学習者が理解しやすい文章構造に整理する．

これにより，後続の図示生成システムが適切に動作するための，構造化された教育的テキストを生成する．

2.3　説明図生成システム

図示生成システムの最初のステップは，入力テキストからプログラミング要素を判定する処理である．この処理は LLM を用いてプロンプトとなる校正済みの文章内になにかしらプログラミング用語が存在しているかを判定する．プログラミング要素を検出した場合は説明図を生成するステップに移行し，検出しなかった場合はその時点でその音声に対する処理を終了する．

説明図の生成では，構造的で一貫性のある図を説明するために，表 1 に示す 3 つの項目をプロンプトに含めて LLM に図を生成させる．与えられた説明に対して，重要な概念を階層的にグループ化し，情報をどのように配置すべきかを中間的に出力させることで構造化された説明図の生成を促す．また，説明図の各コンポーネントの色合いも決定させることで重要度によって色の使い分けを期待する．

3　評価実験

3.1　実験設定

本実験では使用可能な公開データセットが存在しなかったため青山学院大学理工学部 1 年次の学生が受講する 2024 年度の実習において全 15 回ある授業のうち第 13 回および第 14 回の授業を対象とした．該当授業において，8 名の Teaching Assistant (TA) にイヤホンを装着してもらい，授業内の質問対応の音声を取得した．この講義は必修授業であり，C 言語プログラミングの基礎を学ぶものとなっており，第 13 回の授業内容は構造体，第 14 回の授業は乱数の授業となっている．質問対応の始まりから対応終了までを一つの音声データとして数え，合計 8 人の TA から構造体は 99 件，乱数は 43 件の合計 142 件の音声データを録音した．取得した音声は発言者が特定出来ないように匿名化した上で使用している．

3.1.1　データセットの構築手順

収集した音声データの前処理として，OpenAI の Whisper v1 モデルを用いて全音声の文字起こしを実施した．Whisper は多言語対応の高精度音声認識モデルであり，日本語の専門用語を含む音声に対しても比較的安定した認識性能を示すことが確認されている．文字起こし結果をもとに以下の基準でデータセットを分類した：

- 成功データ基準：プログラミング用語（`scanf`、`fgets`、`malloc` 等）が正確に認識されており、かつ問題提起から解決策提示までの論理的流れが確認できる音声
- 対照データ基準：上記基準を満たさない一般的な質問対応音声

分類作業の結果，構造体のデータでは確実に図生成が可能と判断される音声が 6 件，生成可能性のある音声が 8 件の計 14 件を成功データとして抽出した．残りの 85 件からランダムサンプリングにより 10 件を対照データとして選定した．乱数のデータについても同様の手順で成功データ 10 件，対照データ 10 件を構築した．

3.1.2　評価指標の設計

生成された説明図の品質を定量的に評価するため，プログラミング教育の観点から重要な要素を考慮した 9 項目の評価基準を設計した（表 2）．各項目は二値評価（1：満たしている，0：満たしていない）とし，全項目の平均値を生成精度として定義した．この評価手法により，二値評価（1：満たしている，0：満たしていない）でシステム性能を定量化できる．評価項目の設計においては，(1) 教育的観点（概念理解の促進，論理的構造の明確化），(2) 技術的観点（SVG 形式の要件充足，エラー処理の適切性），(3) 実用的観点（視認性，具体性）の 3 つの側面を総合的に考慮した．各項目は可能な限り主観性を伴わない範囲で設定したが，本評価には一定の主観性を含まれる．そのため，各項目の評価は著者により目視によって行った．

表 2　評価項目一覧

評価項目	詳細内容
図の生成状況	説明内容に関連する図が生成されているか
問題・解決の一貫性	問題とその解決策が論理的に結びついているか
概念・関数の表現	音声認識した概念・関数が図に存在するか
表現形式の適切性	図解・テキスト・コードの形式が適切か
関数の説明	関数の動作や役割が適切に説明されているか
エラー条件の明示	想定エラーと対処法が説明されているか
実装例の具体性	具体的な使用方法が示されているか
SVG 要件の充足	指定した SVG 基本要件を満たしているか
変数処理の適切性	変数が正しく認識・処理されているか

表 3　構造体：生成データの統計分析

項目	全データ	成功データ	対照データ
生成枚数	69 枚	41 枚	28 枚
最大値	1.000	1.000	1.000
最小値	0.182	0.182	0.273
平均値	0.663	0.687	0.627

表 4　乱数：生成データの統計分析

評価指標	全データ	成功データ	対照データ
生成枚数	43 枚	31 枚	12 枚
最高値	1.000	1.000	0.909
最小値	0.273	0.273	0.545
平均値	0.709	0.645	0.773

3.2　定量的評価結果

本研究では，構造体および乱数の 2 つの授業データを用いてシステムの有効性を評価した．各データセットにおいて，成功データと対照データに対してシステムを適用し，生成された説明図の品質を定量的に分析した．

3.2.1　構造体授業データの分析

構造体授業データでは，総音声データ 20 件に対し 69 枚の画像が生成された．表 3 に示すように，データセット別では成功データから 41 枚（67%），対照データから 28 枚（33%）の画像が生成され，成功データでより多くの視覚化が実施された．生成精度については，成功データで平均 0.687，対照データで平均 0.627 となり，成功データがやや高い精度を示した．図 2 と図 3 に示す生成精度の分布では，成功データは 0.6 から 0.8 の間にピークを持ち，対照データは 0.4 から 0.6 に集中する傾向が確認された．注目すべきは，両データセットともに最高精度 1.0 を達成したケースが存在する一方，最低精度 0.182 は成功データから記録されたことである．これは，成功データとして分類された音声であっても，音声認識エラーや文脈の断片化により低品質な図が生成される可能性を示唆している．

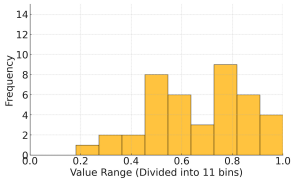

図 2　構造体：成功データ精度分布

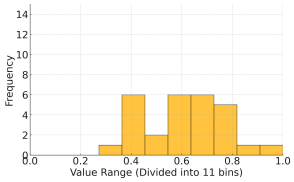

図 3　構造体：対照データ精度分布

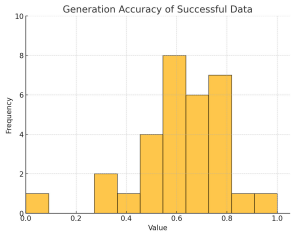

図 4　乱数：成功データ精度分布

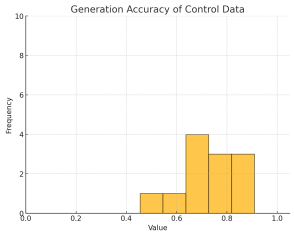

図 5　乱数：対照データ精度分布

3.2.2　乱数授業データの分析

乱数授業データでは，総音声データ 20 件に対し 43 枚の画像が生成された．表 4 に詳細を示すように，データセット別では成功データから 31 枚（72%），対照データから 12 枚（28%）の画像が生成された．しかし，生成精度については成功データで平均 0.645，対照データで平均 0.773 となり，構造体授業とは異なる傾向を示した．図 4 と図 5 の分布図からも，この傾向の違いが視覚的に確認できる．この逆転現象の原因として，成功データに含まれる長時間音声（20 分超）での文脈維持の困難性が挙げられる．長時間音声では説明の流れが断片化されやすく，結果として生成精度が低下する傾向が観察された．

両実験の比較から，以下の知見が得られた．第一に，プログラミング用語の正確な認識と文脈の一貫性が生成精度に大きく影響することが明らかになった．第二に，音声の長さや内容によって生成精度が変動し，長時間音声では文脈維持が困難になる場合があることが確認された．第三に，成功データとして分類された音声でも，必ずしも高精度の図が生成されるとは限らないことが判明した．

3.3　定性的評価結果

3.3.1　高品質生成例の分析

図 6 に示す最高精度生成例では，プログラミング教育における理想的な視覚化が実現されている．この例では，配列インデックスエラーという一般的なプログラミング課題について，概念導入部における配列の基本構造と正常なアクセスパターンの図示から始まり，問題提起部では範囲外アクセスが発生する状況の視覚化を行い，最終的に解決策提示部において適切な境界チェックの実装例を示すという教育的構造が確認された．この 3 段階構成により，学習者は問題の本質から解決策まで段階的に理解できる設計となっている．特に，抽象的な配列やメモリの概念を具体的な図形表現で示すことで，初学者にとって理解困難なポインタや配列操作の概念化を効果的に支援している．

3.3.2　失敗パターンの系統的分析

一方，低精度の生成結果からは複数の問題が明らかになった．図 7 に示すように，最も深刻な問題は図の生成

図 6　構造体：成功データ最高精度生成例

が途中で終了する技術的不具合である．これは SVG 生成プロセスの安定性に関わる課題であり，実用化に向けて改善が必要である．失敗パターンとして，SVG 要素の構文エラー，座標計算の範囲外エラー，テキスト要素の重複配置などが確認された．また，プログラミング用語が不十分な音声では，必要な要素が不足し，適切な領域分割がなされない問題も発生した．さらに，音声データに第三者の発話が混入した場合，校正システムが全ての要素を処理しようとするため，本来の説明の文脈が破綻することが確認された．この問題は実際の教室環境では頻繁に発生する可能性があり，実用化に向けて話者分離技術等の導入が必要であると考えられる．

視覚的品質の観点では，高精度結果において表 1 の視覚的階層性，情報の流れ，視認性が適切に実現されている．重要な概念が視覚的に強調され，関連要素が適切にグループ化されることで，認知的負荷の軽減が図られている．

これらの分析から，現在のシステムがノイズ等の外部要因に対して脆弱であることが判明した．特に，複数話者環境や長時間の連続説明において，文脈の維持と適切な図生成の両立が困難であることが課題として浮上した．今後は，話者分離技術の導入や，文脈の一貫性を保持するためのより堅牢な校正手法の開発が必要である．

4　おわりに

本研究では，プログラミング教育における指導音声からの説明図自動生成システムを提案した．提案システムは，音声認識，校正，説明図生成の 3 つの主要コンポーネントから構成され，教員の音声説明をリアルタイムに視覚的表現へと変換することでプログラミング教育における理解促進を支援する．その際，生成図の品質を定量化するために独自に定義した平均生成精度指標（0〜1）を導入した．定量分析の結果，構造体授業では成功データで平均 0.687，乱数授業では平均 0.645 の生成精度を達成した．また，定性分析では，適切な音声入力に対して概念を段階的に視覚化する説明図が生成されることが確認された．一方で，長時間音声での文脈維持や外部ノイズへの対応など，実用化に向けた課題も明らかになった．定量評価に使用した指標は最低限の項目を評価するだけのものであったが，課題内容の違いを反映して数値が変動し，少なくとも同一条件下での性能傾向を把握する手がかりとして機能していた．今後は複数評価者によるアンケート評価や，学習者を対象とした教育効果測定などの客観的評価など学習成果や認知負荷の観点からの検証も必要である．本研究は技術性能の評価に留まったが，利用する LLM やプロンプト，対象とするプログラミング概念による傾向の違いの検証も必要である．

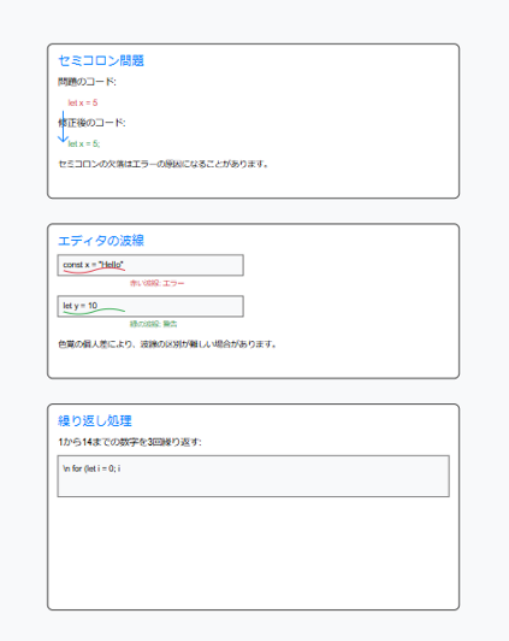

図 7　構造体：成功データ最低精度生成例

参考文献

[1] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

[2] Lilia García-Mundo, Juan Vargas-Enríquez, Marcela Genero, and Mario Piattini. The impact of program visualization tools on teaching programming concepts: A systematic literature review. *IEEE Access*, 11:7391–7412, 2023.

[3] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. *arXiv preprint arXiv:2205.11916*, 2022.

[4] Monika Mladenovi'c, Žana Žanko, and Marin Agli'c Čuvi'c Mladenovi'c. Learning algorithms with unified and interactive web-based visualization. *Computer Applications in Engineering Education*, 29(1):1–15, 2003.

[5] Monika Mladenovi'c, Žana Žanko, and Marin Agli'c Čuvi'c Mladenovi'c. Online python tutor: Embeddable web-based program visualization for cs education. *Computer Applications in Engineering Education*, 29(1):145–159, 2013.

[6] Frank F Wu, Kenneth Lai, Jane Hsu, Cheng-Yu Hsieh, Lei Chen, and Matei Zaharia. Automatic feedback generation for programming assignments using large language models. *arXiv preprint arXiv:2106.10701*, 2021.

[7] 文部科学省. 小学校プログラミング教育の手引（第三版）. https://www.mext.go.jp/content/20200218-mxt_jogai02-100003171_002.pdf.

[8] 文部科学省. 小学校段階におけるプログラミング教育の在り方について（議論の取りまとめ）. https://www.mext.go.jp/b_menu/shingi/chousa/shotou/122/attach/1372525.htm, 6 2016.

[9] 松河 秀哉 and 稲垣 知宏.「視覚的顕在化」に着目したプログラミング学習教材の開発と評価. 日本教育工学会論文誌, 37(1):41–50, 2013.