

SpecifyUI: Supporting Iterative UI Design Intent Expression through Structured Specifications and Generative AI

Yunnong Chen, Chengwei Shi, Liuqing Chen
2025/09

arXiv

研究の概要

- | 自然言語のプロンプトに依存した現在のAIデザインツールの限界を超えるアプローチを提案
- | AIと人間の間に「SPEC」という構造化された設計図を挟むことで、意図通りのデザインを生成し、安全に修正できるシステム「SpecifyUI」を開発

背景：AIによるUIデザインの進化と現在

- | 近年、LLMの進化により、テキストや画像から一瞬でUIのプロトタイプを生成できるようになった。
- | デザイン初期のアイデア出しのスピードは格段に上がった。

課題1:意図の表現の難しさ

UIデザインの意図は抽象的（レイアウトのバランスや視覚的階層など）

デザイナーは視覚的に思考しますが、既存のAIツールではこれを自然言語に翻訳して入力する必要があり、伝達にズレが生じる

複数の参考画像から特定の要素のみを抽出・合成するような複雑な意図を、テキストのみで正確に伝達することは困難

課題2:反復的修正における制御損失

- | UIデザインは、反復的な修正を通じて段階的に洗練されるプロセス
- | プロンプトベースのツールでは、一部のテキストを変更するだけで、意図していないレイアウト全体の構造的変化が引き起こされる事象が頻発する
- | デザインの一貫性を維持したまま、局所的に微調整を行う機能がない（制御損失）

専門家への事前調査

表1. 形成的研究における参加者の人口統計情報、年齢順に整理。

ID	性別	年齢	職種	経験年数
P1	男性	40	UIディレクター	>10 年
P2	男性	35	UIエンジニア	>10 年
P3	男性	32	UIエンジニア	5～10年
P4	女性	26	UIデザイナー	1～3年
P5	女性	27	UIデザイナー	5～10年
P6	女性	29	UIディレクター	5～10年

実務における課題を明確化するため、6名のUIデザイナーを対象に、既存のAIツールを用いた評価調査を実施

「視覚的な要素（スタイルや階層構造）をテキストで記述することの困難さ」が共通の課題として挙げられた

「修正のたびに変更しない部分のプロンプトまで再記述しなければならず、作業が非効率かつエラーを誘発しやすい」という構造的欠陥が確認された

「SPEC（構造化仕様）」の導入

課題を解決するため、本研究では「SPEC」という新たな中間表現を提案

SPECは、自然言語の「曖昧さ」とコードの「厳格さ」を埋める、パラメータ化された階層的なデータフォーマット

プロンプトの代わりにこのSPECを設計仕様として人間とAIで共有することにより、意図の反映と制御性を両立

SPECの構造1:グローバル仕様

グローバル仕様:全体のデザイン原則

- レイアウト構造
- カラーシステム
- 形状言語
- 利用シナリオ

グリッドの列数や余白サイズ、主要な配色のカラーコード、角丸の半径などを、定量的な数値と意味的タグのセットとして保持

SPECの構造2:ページ構成

ページ構成：UIの階層関係

ページを「セクション」に分割し、さらにその内部の「コンポーネント」へとツリー状に分解して管理

各コンポーネントは機能的役割を持つが、スタイル等は「グローバル仕様」から継承される

提案システム「SpecifyUI」

主な3つのプロセス

- **SPEC Initialization**

参考画像の領域分割と、それに伴うSPECの自動抽出および合成

- **SPEC Edition**

抽出されたSPECに対する、階層を意識した局所的かつ直接的なパラメータ編集

- **UI Generator**

最終的なSPECに基づく、UIコードの生成

プロセス1：SPECの自動抽出

- 入力された参考UI画像を独自の物体検出モデル（Co-DETR）を用いて、意味的な領域ごとに分割
- 分割された各領域をLLMに解析させ、レイアウトやコンポーネントの種類、配色などを抽出し、自動的にSPECフォーマットへと変換
- ユーザーは複数の参考画像から特定の要素を選択し、柔軟に合成することが可能になる

プロセス2：制御性を担保した反復的編集

ユーザーが編集意図を入力すると、システムはそれを構造化されたコマンドに変換

SPECのツリー構造内の特定のノードをピンポイントで更新することが可能

変更意図のない他のレイアウトや構造を破壊することなく、安全かつ予測可能な非破壊的編集を実現

プロセス3：LLMによるコード生成

コード生成に特化したLLM（Qwen3-Coder）がSPECから実働するReactコードを出力

RAG技術を活用し、事前に構築された2,000件のSPEC-UIコードデータベースから類似の構造を参照することで、出力の精度と安定性を向上

「デバッグエージェント」がコンパイルエラーを検知・自己修正するフィードバックループを実行

定量的評価

提案手法が参考画像の意図をどの程度再構成できるか、既存のプロンプト手法と比較する定量評価を実施

- MSE（ピクセル誤差）
- CLIP（意味的類似度）
- SSIM（構造的忠実度）

SpecifyUIはベースライン手法と比較して再構成エラーを11.4%削減し、構造的忠実度を14.4%向上させた

方法	MSE↓	CLIP↑	構造的類似度↑
直接プロンプト	57.4214	0.7024	0.688
テキスト強化プロンプト	53.2755	0.7058	0.7017
自己修正プロンプト	50.8570	0.7553	0.787
統合SPEC (SPEC+RAG+領域クロップ)	40.9930	0.8871	0.854
SPEC+RAG	44.8892	0.8598	0.8167
SPEC+領域クロップ	42.1189	0.8744	0.8302
のみSPEC	45.1431	0.8628	0.8166

ユーザー評価

- | 16名のプロのUIデザイナーを対象に、業界標準ツールとSpecifyUIの比較評価実験を行った
- | 設計の品質、意図の伝達しやすさ、制御性などの全項目において、SpecifyUIが高い評価を獲得
- | 操作時のタイピングによる文字入力量が平均で約73%削減された

まとめ

- | AIによるUI生成プロセスを「単発のプロンプト依存型」から、「構造化仕様を介した反復的な協働型」へとシフトさせる新たな枠組みを提示
- | 曖昧な設計意図を明確なパラメータとして抽出・編集可能にしたことで、表現力と制御性の両立に成功