SpiderTailed：視覚的プロパティを利用した Presentation Failure 検出手法[†]

岡嶋　隆人[*]

SpiderTailed: Presentation Failure Detection Method Using Visual Properties

Takato Okajima

1　はじめに

今日のアプリケーションではユーザインターフェースとして主に GUI（Graphical User Interface）が採用されており，必要不可欠なものとなっている．そのため，GUI コンポーネントの描画結果が仕様を満たさない場合，大きな問題となる．

本研究ではこのような GUI コンポーネントの描画結果が仕様を満たさない問題を Presentation Failure [1]と呼ぶ．一般的に，Presentation Failure の検知は，目視によりテスト対象のアプリケーションの GUI の描画結果を確認することにより行われているが，画面数に比例してコストが増加する問題がある．

前述の問題を解決するため，Presentation Failure を検知する手法はこれまでに数多く提案されているが，その多くは画像を直接比較することによって実現されているため，大きな差異があると，Presentation Failure の検出は困難であるという問題がある．

2　目的

本研究では，前述の問題を解決するため，回帰テストにおいて，バージョン間の Web ページの描画結果からコンピュータビジョン技術によって抽象化された視覚的プロパティを抽出し，それぞれから抽出した視覚的プロパティを比較することにより，Presentation Failure を検知する手法 SpiderTailed と，その利用例として，GUI の見た目を採点対象とした OJS（オンラインジャッジシステム）SELERESGP を提案する．

本研究では，前述の提案手法を実装し，目視による作業と比較した結果について述べる．

[†]本研究の一部は以下において発表した
・The 14th International Joint Conference on Knowledge-Based Software Engineering (JCKBSE 2022),
[*]電子情報メディア工学専攻　2218004　橋浦研究室

3　提案手法

SpiderTailed の概要を図 1 に示す．SpiderTailed は Python と各種ライブラリを用いて実装されている．

SpiderTailed では，始めに異なる 2 つのバージョンの Web ページを入力として受け取り，以下 4 つの処理を順に実行する．

1)　スクリーンショットの取得
2)　視覚的プロパティの抽出
3)　視覚的プロパティの比較
4)　比較結果の出力

3.1　スクリーンショットの取得

この手順では，まず，テスト対象の Web ページから Web ページから HTML パーサの 1 つである Beautiful Soup を用いて DOM を取得する．次に，取得した DOM を各要素セレクタを抽出し，抽出したセレクタとブラウザのレンダリング結果を基に Web フロントエンドテストツールの Selenium を用いてスクリーンショットを取得する．

3.2　視覚的プロパティの抽出

この手順では，前の手順で取得したスクリーンショットから，コンピュータビジョン技術を用いて視覚的プロパティを抽出する．SpiderTailed ではコンピュータビジョン技術を使用するために，コンピュータビジョンライブラリの OpenCV を利用している．

抽出する視覚的プロパティの一覧を表 1 に示す．以下に視覚的プロパティの抽出方法について述べる．

幅(width)と高さ(height)の視覚的プロパティの抽出には，各要素のスクリーンショットの幅と高さを用いた．

絶対座標(x-location，y-location)の視覚的プロパティの抽出には，テンプレートマッチングと特徴点マッチングを用いる．具体的には，<body>要素の画像を検索空間に，各要素の画像を検索対象としてマッチングを行う．

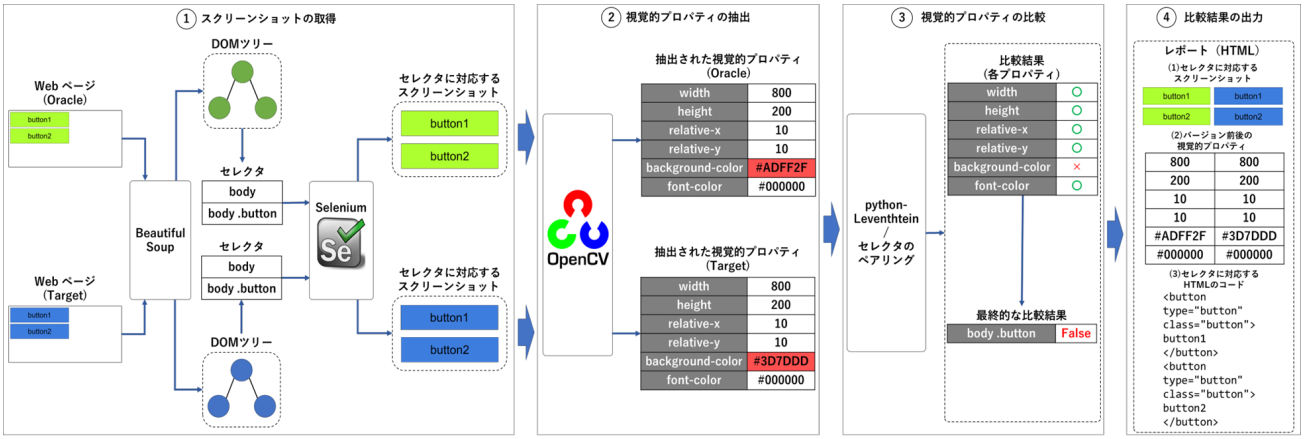

図 1:SpiderTailed の概要

SpiderTailed では，この処理により取得された座標を絶対座標の視覚的プロパティとする．

相対座標(relative-x，relative-y)の抽出には，絶対座標の抽出と同じくテンプレートマッチングと特徴点マッチングを用いる．絶対座標と異なるのは，抽出対象の要素の親要素のスクリーンショットを検索空間として用いることである．

背景色(background-color)の抽出には，ヒストグラムを用いる．SpiderTailed では，各要素のスクリーンショットから RGB 各色のヒストグラムを取得し，その中で最も占有比率の高い色を背景色の視覚的プロパティとする．

フォントカラー(font-color)の視覚的プロパティの抽出には, ヒストグラムとテキスト検出技術を用いる. SpiderTailed では，まず、テキスト検出技術を用いて各要素のスクリーンショットからテキストを検出する．次に、検出されたテキストの外接矩形で切り出した画像から、背景色を除いた RGB 各色のヒストグラムを計算する．そして，最も高い占有率の色をフォントカラーの視覚的プロパティとする．

表 1：抽出する視覚的プロパティの一覧

#	プロパティ名	意味
1	width	幅
2	height	高さ
3	x-location	絶対座標(x)
4	y-location	絶対座標(y)
5	relative-x	相対座標(x)
6	relative-y	相対座標(y)
7	background-color	背景色
8	font-color	文字色

3.3 視覚的プロパティの比較

この手順では，前の手順で抽出された視覚的プロパティの比較を行う．SpiderTailed ではまず，ジャロ・ウィンクラー距離を用いて 2 つの Web ページのセレクタのペアを作成する．

次に，作成したセレクタのペア毎に，絶対座標を除く視覚的プロパティを比較し，値が一致しない場合，その要素を Presentation Failure として検出する．

3.4 比較結果の出力

この手順では，前の手順で求めた結果を基に，ユーザにテスト結果を提示するためのレポートを生成する．レポートは HMTL 形式であり，各セレクタの視覚的プロパティやスクリーンショット，HTML コードが記載される．

3.5 提案手法の利用例

本研究では，SpiderTailed の利用例として，GUI の見た目を採点対象とした OJS SELERESGP を提案する．SELERESGP の概要を図 2 に示す．

SELERESGP は Web フレームワークの Django と JavaScript のライブラリを使用し, Web アプリケーションとして実装した．SELERESGP でユーザは課題の確認や提出，採点結果の確認を行うことができる．

4 評価

4.1 実験 1：SpiderTailed の評価

SpiderTailed の有効性を確認するため実験を行った．実験は人為的に Presentation Failure を挿入した Web ページに対して SpiderTailed と目視でそれぞれで Presentation Failure の検出を行うというものである．

被験者は日本工業大学と日本工業大学大学院の学生計 12 名とし，事前に作成した 2 種類の Web ページに対して実験を行い，その結果に対して考察を行った．

4.2 実験の 1 の RQ の設定

SpiderTailed のの評価の明確化のため，以下の 3 つの RQ を設定した．

RQ1: 目視と比較した場合の SpiderTailed の特徴は？

RQ2: SpiderTailed の所要時間は実用的か？

RQ3: SpiderTailed の課題は？

RQ1 に解答するために，本実験における SpiderTailed と目視のテスト結果を表 2 のように 4 種類に分類する．さらに，これらの分類結果を基に，正解率，適合率，再現率，特異度，F 値を計算し，比較を行う．

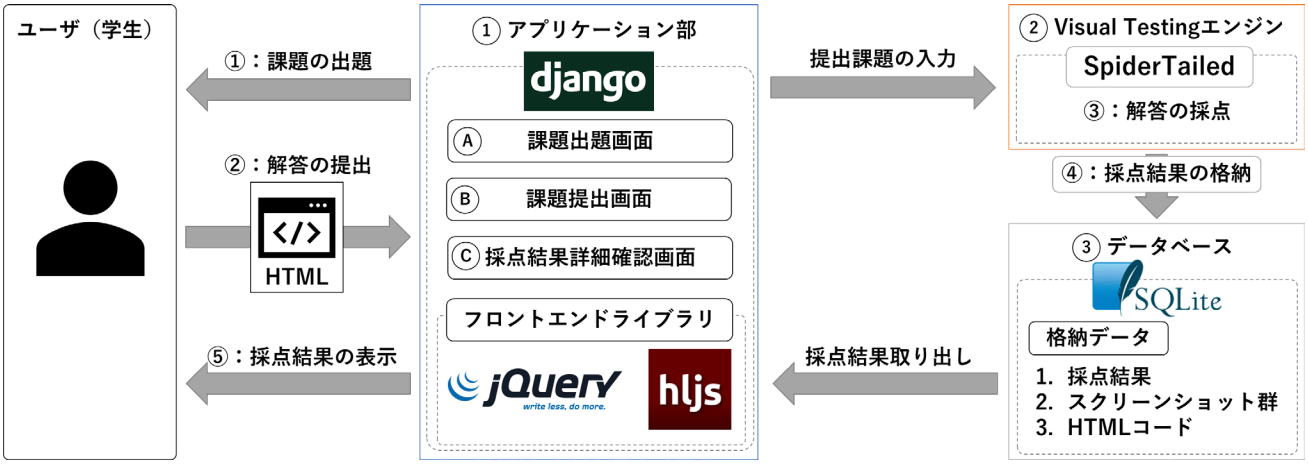

図 2:SELERESGP の概要

RQ2 に解答するために，SpiderTailed を 10 回実行し，記録された所要時間と記録された目視のテストの所要時間と比較する．

RQ3 に解答するために，実験において誤検知(FP，FN)と判断された GUI 要素において，どの視覚的プロパティが誤検知の原因となったかを調査する．

表 2：実験 1 のテスト結果の分類

		対応するセレクタの Presentation Failure の有無	
		あり	なし
SpiderTailed/目視	Presentation Failure である	真陽性(TP)	偽陽性(FP)
	Presentation Failure でない	偽陰性(FN)	真陰性(TN)

4.3 実験 2：SELERESGP の評価

SELERESGP の評価を行うため，実験を行った．実験はユーザに実際に SELERESGP を使用して課題の Web ページを作成，提出してもらうというものである．

被験者は日本工業大学の学部生と大学院生計 11 名とし，SELERESGP を用いて計 3 問の課題を解いてもらった．

4.4 実験 2 の RQ の設定

SELERESGP の評価の明確化のため，以下の 2 つの RQ を設定した．

RQ4: SELERESGP と目視の採点の一致率は？

RQ5: 課題の解答にかかった所要時間は？

RQ4 に解答するために，本実験において SELERESGP に提出された解答を確認後，著者が目視による採点を行い，表 3 のように 4 種類に分類する．また，分類結果から，正解率，適合率，再現率，特異度，F 値を計算し，その結果について考察を行った．

RQ5 に解答するために，本実験において，学生が課題提出までにかかった所要時間を計測した．

表 3：実験 2 の採点結果の分類

		目視による採点結果	
		合格	不合格
SELERESGP の採点結果	合格	真陽性(TP)	偽陽性(FP)
	不合格	偽陰性(FN)	真陰性(TN)

5 実験結果と考察

5.1 RQ1: 目視と比較した SpiderTailed の特徴

実験 1 の結果を基に，正解率などを求めた結果を求めた結果を表 4 に示す．

表 4：SpiderTailed と目視の正解率などの比較

#	精度名	SpiderTailed	目視(総和)
1	正解率	0.797	0.785
2	適合率	0.644	0.777
3	再現率	1.000	0.582
4	特異度	0.679	0.903
5	F 値	0.783	0.665

SpiderTailed は，正解率，再現率，F 値では目視を上回る結果となった．しかし，偽陽性が多く発生したため，適合率や特異度は下回る結果となった．

これらのことから，SpiderTailed は目視と比べ偽陰性が低いテストが可能であることが示唆された．

5.2 RQ2: SpiderTailed の所要時間は実用的か？

実験 1 の所要時間の比較を表 5 に示す．本実験において，SpiderTailed は平均 221 秒でテストを完了し，目視と比べても高速であった．

表 5：SpiderTailed と目視の所要時間の比較

	SpiderTailed	目視
所要時間(平均/秒)	221	480

これらのことから，SpiderTailed は安定して高速なテストを行うことができることが示唆された．

5.3 RQ3: SpiderTailed の課題は？

実験 1 で誤検知の原因となった視覚的プロパティを調査した結果，具体的な誤検知の原因は 2 種類だった．

1 つ目は，親要素の影響によるものである．これは，親要素の幅や高さの変更による影響が，抽出対象に及んでしまったためである．

2 つ目は，SpiderTailed のアルゴリズムに起因するものである．SpiderTailed では，抽出結果を条件によって一意に決定している．そのため，誤った視覚的プロパティが抽出結果として選択されることがあった．

5.4 RQ4: SELERESGP と目視の採点の一致率は？

実験 2 で，SELERESGP と目視の採点結果を基に，正解率などを求めた結果を表 6 に示す．

表 6：採点の一致率

正解率	適合率	再現率	特異度	F 値
0.70	0.88	0.47	0.94	0.62

本実験において，再現率は低下する傾向が，適合率や特異度は上昇する傾向にあった．これは，SELERESGP に組み込まれた SpiderTailed のテスト特性が，採点結果にも出たためだといえる．

これらのことから，SpiderTailed は改良を行うことで，OJS の採点にも応用が可能であることが示唆された．

5.5 RQ5: 課題の解答にかかった所要時間は？

実験 2 において被験者全員が課題 1 で制限時間として設定した 45 分を使用した．この結果の原因は 2 つのものが考えれらる．

1 つ目は，被験者の HTML への理解力の不足である．HTML では，ボックスモデルやなどの概念を理解する必要があり，知識の不足が考えられる．

2 つ目は，課題の設計方法の工夫である．本実験における課題はすべて Web ページの作成だったが，学生の理解力や CSS のプロパティ毎など段階に応じいた学習をできるようにする必要がある．

これらのことから，SELERESGP における課題の出題はレベル分けなどの工夫が必要なことが示唆された．

6 関連研究

Web ページの回帰テストにおいて，異なるバージョン間の Web ページの描画結果の差異を検出する既存ツールとして reg-suit[2]がある．reg-suit は，バージョン前後の Web ページの差分画像とコードの差異の 2 つをユーザにフィードバックすることにより，回帰テストの実行を支援する手法を提案している．

Halle ら[3]は Web ページにおける GUI の仕様記述言語 Cornipickle を実装し，記述された仕様と実際の Web ページの DOM のプロパティを比較することで Presentation Failure を検知する手法を提案している．

Mahajan ら[1]は Web ページのモックアップ図と実装版のスクリーンショットを比較し，知覚差分手法(PID)を用いて Presentation Failure を自動で検出する手法を提案している．

Tanno と Adachi[4]は正しい画面と比較対象の画面の差分を取得し，その結果をテスタにフィードバックする半自動的な手法を提案し，これにより所要時間の短縮が可能であることを指摘している．

本研究は既存の研究と比較して，2 つの Web ページを用意するだけで実行でき，全自動で Presentation Failure を検出することができる．また，GUI の状態を独自の言語で記述したりする必要がないことも特徴である．

7 結論

本研究では Web ページのスクリーンショットから各 GUI 要素の視覚的プロパティを抽出し，バージョン間で比較することにより Presentation Failure を検知する手法 SpiderTailed とその利用例としての OJS SELERESGP を提案した．実験の結果，目視よりも高速で正解率，再現率と F 値の高いテストが可能であり，改良を行うことで OJS にも利用可能であることが示唆された．

参 考 文 献

1) S. Mahajan and W. G. J. Halfond, “Detection and Localization of HTML Presentation Failures Using Computer Vision-Based Techniques,” Proc.of IEEE 8th International Conference on Software Testing, Verification and Validation (ICST 2015), pp.1-10, May. 2015.
2) reg-suit, <reg-viz.github.io/reg-suit >, (accessed:2022/01/17)
3) H. Sylvain, B. Nicolas, G. Francis, L. B. Gabriel and B, Oussama, “Declarative Layout Constraints for Testing Web Applications,” Journal of Logical and Algebraic Methods in Programming, Vol.85, pp.737- 758, Aug. 2016.
4) H. Tanno and Y. Adachi, "Support for Finding Presentation Failures by Using Computer Vision Techniques," Proc. of IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW 2018), pp. 356-363, Apr. 2018.

指導教授	審査委員（主査）	准教授	橋浦　弘明
	審査委員（副査）	教授	新井　啓之
	審査委員（副査）	准教授	松田　洋