Compressed-a11y: 視覚的文脈の再構成と冗長性削減による GUI エージェント観測の効率化

竹下理斗 [1]　川田拓朗 [2]　大橋巧 [2]　北田俊輔 [2]　彌冨仁 [1,2]
[1] 法政大学 理工学部　[2] 法政大学大学院 理工学研究科
{michito.takeshita.4t,takuro.kawada.3g,takumi.ohashi.4g}@stu.hosei.ac.jp
shunsuke.kitada.0831@gmail.com　iyatomi@hosei.ac.jp

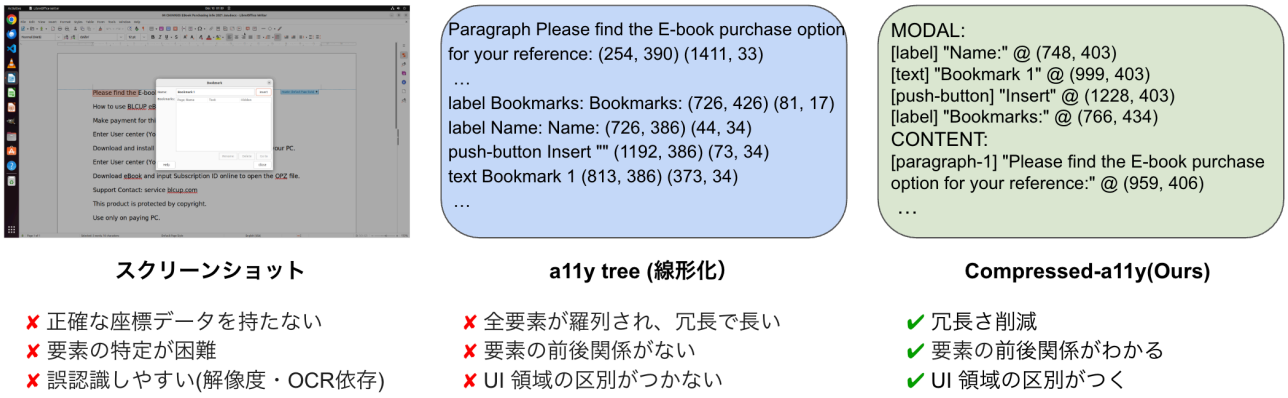

図 1: 既存の観測表現（スクリーンショット，a11y tree）と提案手法（Compressed-a11y）の比較．Compressed-a11y は，GUI 構造を保持しつつ a11y tree を圧縮し，GUI エージェントの grounding を高める観測表現である．

概要

ローカル環境での GUI エージェント実用化には，限られた計算資源下での効率的な画面観測が重要である．Accessibility tree に基づく画面構造の観測は，大規模言語モデル（Large Language Models; LLM）による操作対象の特定に有効な一方，情報の冗長性や視覚的文脈の欠如，状況理解の正確性の限界が課題である．本研究では，LLM が効率的に画面構造を理解可能な観測表現として，ノード削減と空間配置の再構成により情報を圧縮した Compressed-a11y を提案する．評価実験の結果，Compressed-a11y は入力トークン数を最大 90% 削減し，多くのタスクで高い成功率を達成した．本研究は，ローカル環境における効率的な GUI 観測手法の実現に貢献する．

1　はじめに

ローカル環境で動作するマルチモーダル対応の大規模言語モデル（Large Language Models; LLM）（以下，ローカル LLM）を用いたエージェントは，実世界において GUI を含むソフトウェア環境を自律的に操作し，タスクを遂行できる可能性を持つ．現在の GUI エージェント研究では，莫大な数のパラメータによって高度な理解力と推論能力を実現するクラウドベースのクローズドモデルが主流である [1, 2]．一方で，これらのモデルを実社会の多様なタスクに適用するには，データのプライバシー保護，通信に伴う遅延，ならびに継続的な運用コストの面で解決すべき課題が残されている．これに対しローカル LLM は，デバイス内で完結したセキュアかつ低遅延な推論が可能であり，これらの実運用上の課題を解決する現実的な解として期待されている [3, 4]．

しかし，ローカル LLM を用いた GUI エージェントの実用化には未だ課題が多く，とりわけその性能は，複雑な画面情報をいかに正確かつ効率的に処理できるかという環境観測の設計に大きく依存している [5]．従来，GUI の観測には，スクリーンショットに基づく画像情報 [5, 6] のほか，Web ページを構成する要素をプログラムから操作可能な構造として表現した Document Object Model [7] や画面上の UI 要素の意味や操作性を機械可読な階層構造として表現した Accessibility（a11y） tree [8] によるテキスト情報，あるいはそれらを統合したマルチモーダルな表現 [9] が用いられてきた．

特に a11y tree は，操作対象の特定（grounding）に有用である一方で，膨大なノード数に起因する情報の冗長性や，要素の重なり順といった視覚的文脈が明示されないという構造的な問題を抱えている．このような特性は，計算リソースやコンテキスト長に厳しい制約のあるローカル LLM において，観測情報の処理や画面構造の推論を著しく困難にする要因となる [10]．こうした課題に対し，a11y tree の冗長性を低減する手法や，タスクに応じて情報を抽出する手法が提案されている [7, 11, 12]．しかし，GUI が持つ空間的・階層的構造を十分に保持した観測表現の確立には至っていない．

本研究では，a11y tree を基にした観測表現である Compressed-a11y を提案し，ローカル LLM が画面構造を効率的に理解可能にすることを目指す．本手法は，ルールベースにより不要なノードを削減すると同時に，a11y tree では欠落しがちな要素間の空間的配置や UI の意味的なまとまりを明示的に再構成する．これにより，ローカル LLM の推論負荷を最小限に抑えつつ，正確な grounding に必要な情報を過不足なく提供することを可能にする．

Compressed-a11y の有効性を検証するため，GUI 操作を含むタスクに対応可能なローカル LLM として Qwen3-VL [13] を用い，GUI エージェント評価ベンチマーク OSWorld [11] において評価を行った．タスク成功率やトークン数といった定量的指標に基づく評価を通じて，ローカル LLM を用いた GUI エージェントにおける効率的な環境観測設計の一つの方向性を提示することを本研究の目的とする．

2 関連研究

視覚的判断や暗黙知を要する GUI 操作の難易度 [14, 11]，とりわけエージェントによる grounding の困難さ [15] を克服するため，Set-of-Mark に代表される画像へのラベル重畳 [16] や，要素の機能を言語化するキャプション付与 [17] など，視覚情報を補完する手法が提案されている．これらの手法は，指示と操作対象との対応付けを改善する一方で，画面全体を俯瞰した構造の理解や，複数要素から成る意味的単位の把握を直接の対象とはしていない．

画像ベースの手法に代わる観測表現として，a11y tree を活用する研究も注目されている [7]．しかし，a11y tree は GUI 要素の名称や役割，画面上の座標が明示されるため，grounding には優れるものの，GUI 特有の空間的配置や重なり関係，表示状態の遷移といった視覚的制約が十分に反映されないという課題がある．例えば，ユーザーの注意を現在の文脈に集中させるために画面前面に一時的に表示される UI や，操作に応じて補足情報を一時的に前面に提示する UI の前後関係が構造に反映されず，最前面の操作要素の誤認を引き起こす恐れがある．また，視覚的な近接性が論理構造では分断されるため，配置に基づく文脈把握が困難になる．さらに，情報の冗長性がコンテキスト長を圧迫し，ノイズの中から真に操作すべき要素を選別する精度を低下させる [18]．

このような a11y tree に内在する課題に対処するため，階層構造をモデル入力向けのテキスト列へ変換する線形化により冗長性を低減する手法 [7, 11] やタスク指示に基づき必要な情報を LLM で事前に抽出・要約する手法 [12] も提案されている．これらの手法は重要な進展である一方で，GUI に内在する空間的・階層的な性質を明示的に扱う点については，さらなる検討の余地が残されている．この課題に対処するため，本研究では既存手法が十分に扱えていない GUI の空間的配置や階層構造を明示的に保持する観測表現を設計する．

3 手法

本研究では，ローカル LLM による GUI 操作を対象として，a11y tree を基にした新たな観測表現 Compressed-a11y を提案する．図 1 に，Compressed-a11y と，既存の代表的な観測表現であるスクリーンショットおよび線形化された a11y tree との比較を示す．Compressed-a11y は，線形化された a11y tree を入力として生成される観測表現であり，以下の 3 段階からなる処理パイプラインによって構築される．

3.1 モーダル検出

メインコンテンツとは独立した一時的な前面 UI（以下，モーダル）を検出するため，時間的差分と空間的凝集性に基づく二つのアプローチで候補領域を抽出し，検証によりモーダル領域であるかを判定する．具体的には，以下の 3 ステップを採用する．

Step 1: 時間的差分に基づくモーダル候補抽出
ユーザー操作に応答して新たに出現する要素を捉えるため，直前および現在の画面状態における a11y tree の差分から新規要素の候補群を得る．この時，前回の観測から画面全体のスクロールやウィンドウの移動が発生していた場合，構成要素の座標が一様に変化するため，単純な差分検出では誤認が生じ

る．そのため，前後フレーム間で同一とみなせる要素の位置変化から画面全体の移動量を推定して座標を補正し，新しく出現した要素のみを抽出する．

Step 2: 空間的凝集性に基づくモーダル候補抽出 時間的差分情報が利用できない初期状態においても，ボタンや入力フォームといった典型的なモーダル構成要素の空間的凝集性に基づき，視覚的な配置特徴からモーダル候補を抽出する．

Step 3: 妥当性検証と安全フィルタ 抽出された候補群に対し，領域内要素のロールや機能に関連するキーワードに基づきスコア化し，所定の閾値を超過した場合にモーダルとして検出する．なお，検出されたモーダルは，時間的差分が生じなくなった後も，a11y tree 上に存在する限り同一要素として追跡し，検出結果の一貫性を維持する．検出されたモーダル領域は，メインコンテンツとは独立したレイヤーとし，後続の構造化処理（3.3 節）に反映する．

3.2 観測表現の正規化と冗長情報の除去

エージェントの判断に必要な情報を損なわず，高い情報密度を持つ観測表現を構築するため， a11y tree に内在する冗長性や不要な情報を削減する処理と，指示内容に応じた動的な最適化を組み合わせた以下の４ステップを行う．

Step 1: ルールベースによるノイズ除去 a11y tree からタスク遂行に不要なノイズを静的に除去し，画面描画のみを目的とする不可視要素や，OS が自動生成する不要なメタデータ（クラス名や内部 ID など）をルールベースで除外する．

Step 2: 視覚的重複の統合 a11y tree 特有の情報冗長性に対処するため，要素中心間の距離とラベル文字列の類似性に基づき，座標が近接し意味的に重複する要素群を統合する．一定距離内に位置し，かつラベルが同一または包含関係にある要素を同一の操作対象とみなし，入力フォーム → ボタン → 見出し → 静的テキストの順でロール優先度を適用した単一の要素として再定義する．

Step 3: 属性の選別と座標の簡略化 統合された各要素に対し，ロール，ラベル，値といったタスク遂行に不可欠な属性のみを抽出し，過度な改行や空白を除去することで，LLM が解釈しやすい簡潔な形式へ整形する．また，要素の位置情報については，従来の「左上座標とサイズ」に代えて中心座標を用いる．これにより，要素間の相対的な位置関係を維持しつつ，座標記述に必要なトークン数を削減し，コンテキスト効率を向上させる．

Step 4: 指示内容に基づく動的最適化 すべての a11y tree 上の静的テキストを一律に保持すると LLM の処理効率が低下するため，指示文を解析してタスク遂行に関連するキーワードを抽出し，当該キーワードを含む要素の前後を優先的に保持する動的フィルタリングを適用する．キーワードが検出されない場合には，情報欠損を避けるため，あらかじめ設定した文字数分の要素列を先頭から出力する．

3.3 意味的領域分割と構造化

LLM が画面のレイアウト構造を正しく解釈できるように，3.1 節で検出し，3.2 節で正規化されたモーダル領域を除いた要素列を入力として，以下の 3 ステップから成る構造化処理を行う．

Step 1: 意味的領域への分割 a11y tree からアプリケーションのドメイン（Chrome，GIMP，VLC など）を特定し，ドメインに対応した UI 要素パターンに基づいて，画面端に固定配置されるメニューバーやステータスバーなどのシステム UI 領域とメインコンテンツ領域を分離する．これにより，テキスト上での情報の混在が解消され，LLM は画面全体のレイアウト構造と各領域の役割を認識しやすくなる．

Step 2: 空間的整列による一次元化 a11y tree 上のノード順序は視覚的な配置と一致しないため，各要素の座標情報に基づき，Y 軸方向（上から下）を優先し，同一行内とみなせる範囲では X 軸方向（左から右）の順で再整列することで，視覚的な閲覧順序を反映した並べ替えを行う．

Step 3: 論理構造の復元 整列されたノード列から視覚的なまとまりを復元するため，縦方向のギャップに基づいてブロックを形成し，その中で横方向のギャップにより分割を行う．この縦横両方向の分割により，並列配置された要素群もテキスト上で独立したブロックとして保持される．また，見出し属性を持つ要素はブロックの開始点として扱い，視覚的配置と論理的な階層構造の両立を図る．

4 実験

4.1 実験設定

提案手法 Compressed-a11y の有効性を検証するため，ローカル環境で動作するマルチモーダル対応の LLM として Qwen3-VL-32B [13] を用い，統合的な GUI 操作ベンチマークである OSWorld [11] により評

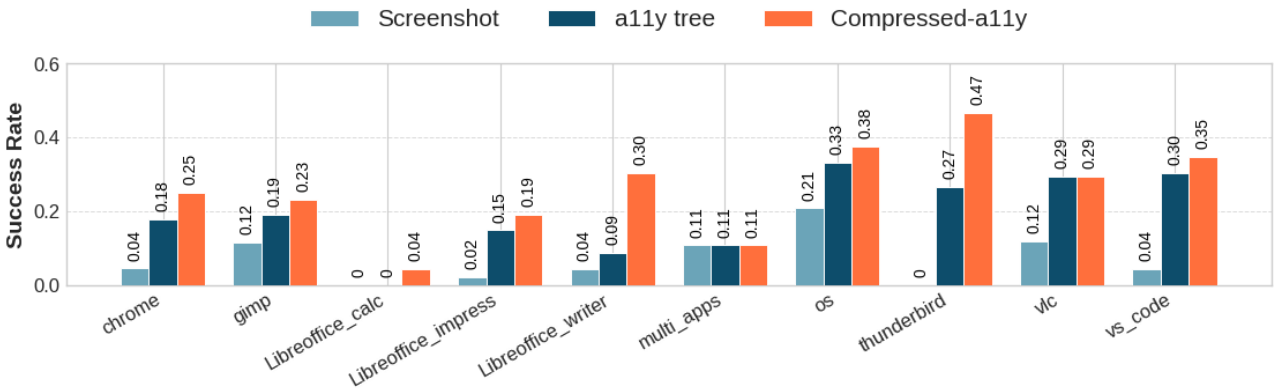

図 2: 各ドメインにおける観測表現の成功率比較

価を行った．環境依存のエラーにより実行不可能であったタスクを除外し，Web ブラウジング（Chrome: 45 タスク），オフィスワーク（Writer: 23 タスク，Calc: 46 タスク，Impress: 47 タスク），メール管理（Thunderbird: 15 タスク），メディア編集（GIMP: 26 タスク，VLC: 17 タスク），ソフトウェア開発（VS Code: 23 タスク），OS 基本操作（24 タスク），および複数アプリケーション横断タスク（93 タスク）から成る多様なアプリケーションドメインを対象とした．OSWorld は実環境上で動作するため非決定性を含むことから，各タスクにつき 2 回試行し，いずれかが成功した場合を成功と判定した．

本実験では，平均トークン数とタスク成功率の 2 指標を用い，前者では Compressed-a11y と線形化 a11y tree を，後者では Compressed-a11y と LLM の入力コンテキスト上限を考慮してトリム化した線形化 a11y tree およびスクリーンショットを比較することで，提案手法の有効性を検証した．

4.2 実験結果・分析

a11y tree を線形化した場合と Compressed-a11y を用いた場合の Qwen3-VL への入力トークン数を比較した結果，Compressed-a11y はすべての対象アプリケーションにおいてトークン数を削減できることが確認された．例えば，Chrome では約 5,500 トークンから約 1,700 トークンへ，LibreOffice Calc では約 30,000 トークンから約 3,400 トークンへと大幅な削減が見られた．このように，削減率はアプリケーションごとに異なるものの，全体としてはおおよそ 35% から 90% 程度の削減が達成された．特に，UI 要素数が多く構造が複雑なアプリケーションほど，Compressed-a11y によるトークン数削減の効果が顕著であった．なお，ドメインごとの詳細なトークン数比較結果については付録 A に示す．

次に，図 2 に各ドメインにおける観測表現手法のタスク成功率を示す．Compressed-a11y は多くのドメインにおいて，他の観測表現より高いタスク成功率を示した．一方，スクリーンショットは全体的に成功率が低く，UI の意味的情報を十分に表現できていないことが示唆される．ただし，multi_apps ドメインでは Compressed-a11y による成功率の向上は見られなかった．これは，本手法が単一アプリケーションドメインを前提とした UI の意味的分割を仮定しており，複数ドメインが混在する状況ではその前提が成り立たないためと考えられる．以上より，Compressed-a11y は多くのドメインにおいて有効である一方，複数ドメインが同時に存在する環境では依然として課題が残ることが示された．

5 おわりに

本研究では，GUI エージェントが扱う冗長な a11y tree を対象として，GUI の階層構造および空間的まとまりを保持したまま圧縮する観測表現 Compressed-a11y を提案した．OSWorld を用いた評価により，提案手法は従来手法と比較してタスク成功率を向上させるとともに，推論時のトークン消費量を大幅に削減可能であることを確認した．これらの結果から，Compressed-a11y の設計は，GUI における重要な構造情報を保持したまま a11y tree の冗長性を削減することで，ローカル LLM による GUI の構造理解を効率的に支援する環境観測設計の一つの方向性を示唆している．

今後の課題として，多様な GUI ドメインやマルチウィンドウ環境への対応，および複雑な画面構成における圧縮精度の向上が挙げられる．

参考文献

[1] Chi Zhang, Zhao Yang, Jiaxuan Liu, Yucheng Han, Xin Chen, Zebiao Huang, Bin Fu, and Gang Han. AppAgent: Multimodal Agents as Smartphone Users. In **Proceedings of the ACM CHI Conference on Human Factors in Computing Systems (CHI)**, 2025.

[2] Chaoyun Zhang, Liqun Li, Shilin He, Xu Zhang, Bo Qiao, Si Qin, Minghua Ma, Yu Kang, Qingwei Lin, Saravan Rajmohan, Dongmei Zhang, and Qi Zhang. UFO: A UI-Focused Agent for Windows OS Interaction. In **Proceedings of the Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)**, 2025.

[3] Xunyu Zhu, Jian Li, Yong Liu, Can Ma, and Weiping Wang. A Survey on Model Compression for Large Language Models. **Transactions of the Association for Computational Linguistics (TACL)**, Vol. 12, pp. 1556–1577, 2024.

[4] Yue Zheng, Yuhao Chen, Bin Qian, Xiufang Shi, Yuanchao Shu, and Jiming Chen. A Review on Edge Large Language Models: Design, Execution, and Applications. **ACM Computing Surveys (CSUR)**, Vol. 57, No. 8, pp. 1–35, 2025.

[5] Kevin Qinghong Lin, Linjie Li, Difei Gao, Zhengyuan Yang, Shiwei Wu, Zechen Bai, Stan Weixian Lei, Lijuan Wang, and Mike Zheng Shou. ShowUI: One Vision-Language-Action Model for GUI Visual Agent. In **Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)**, 2025.

[6] Runliang Niu, Jindong Li, Shiqi Wang, Yali Fu, Xiyu Hu, Xueyuan Leng, He Kong, Yi Chang, and Qi Wang. ScreenAgent: A Vision Language Model-driven Computer Control Agent. In **Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence (IJCAI)**, 2024.

[7] Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Boshi Stevens, Samuel andus, Huan Sun, and Yu Su. Mind2Web: Towards a Generalist Agent for the Web. In **Advances in Neural Information Processing Systems (NeurIPS)**, 2023.

[8] Shuyan Zhou, Frank F Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xian Cheng, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. WebArena: A Realistic Web Environment for Building Autonomous Agents. In **Proceedings of the International Conference on Learning Representations (ICLR)**, 2024.

[9] Boyuan Zheng, Boyu Gou, Jihyung Kil, Huan Sun, and Yu Su. GPT-4V(ision) Is a Generalist Web Agent, If Grounded. In **Proceedings of the 41st International Conference on Machine Learning (ICML)**, 2024.

[10] Yuhao Yang, Yue Wang, Dongxu Li, Ziyang Luo, Bei Chen, Chao Huang, and Junnan Li. Aria-UI: Visual Grounding for GUI Instructions. In **Findings of the Association for Computational Linguistics (Findings of ACL)**, 2025.

[11] Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh Jing Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, Yitao Liu, Yiheng Xu, Shuyan Zhou, Silvio Savarese, Caiming Xiong, Victor Zhong, and Tao Yu. OSWorld: Benchmarking Multimodal Agents for Open-Ended Tasks in Real Computer Environments. In **Advances in Neural Information Processing Systems (NeurIPS)**, 2024.

[12] Imene Kerboua, Sahar Omidi Shayegan, Megh Thakkar, Xing Han Lù, Massimo Caccia, Véronique Eglin, Alexandre Aussem, Jérémy Espinas, and Alexandre Lacoste. LineRetriever: Planning-Aware Observation Reduction for Web Agents, 2025. https://doi.org/10.48550/arXiv.2507.00210.

[13] Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, et al. Qwen3-VL Technical Report. 2025. https://doi.org/10.48550/arXiv.2511.21631.

[14] Jing Yu Koh, Robert Lo, Lawrence Jang, Vikram Duvvur, Ming Chong Lim, Po-Yu Huang, Graham Neubig, Shuyan Zhou, Ruslan Salakhutdinov, and Daniel Fried. VisualWebArena: Evaluating Multimodal Agents on Realistic Visual Web Tasks. In **Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)**, 2024.

[15] Wenyi Hong, Weihan Wang, Qingsong Lv, Jiazheng Xu, Wenmeng Yu, Junhui Ji, Yan Wang, Zihan Wang, Yuxuan Zhang, Juanzi Li, Bin Xu, Yuxiao Dong, Ming Ding, and Jie Tang. CogAgent: A Visual Language Model for GUI Agents. In **Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)**, 2024.

[16] Jianwei Yang, Hao Zhang, Feng Feng, Xue Yang, Weijie Ye, and Pengchuan Zhang. Set-of-Mark Prompting Unleashes Extraordinary Visual Grounding in GPT-4V. In **Proceedings of the International Conference on Learning Representations (ICLR)**, 2024.

[17] Kanzhi Cheng, Qiushi Sun, Yougang Chu, Fangzhi Xu, Yantao Li, Jianbing Zhang, and Zhiyong Wu. SeeClick: Harnessing GUI Grounding for Advanced Visual GUI Agents. In **Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)**, 2024.

[18] Izzeddin Gur, Hiroki Furuta, Austin Huang, Mustafa Safdari, Yutaka Matsuo, Douglas Eck, and Aleksandra Faust. A Real-World WebAgent with Planning, Long Context Understanding, and Program Synthesis. In **Proceedings of the International Conference on Learning Representations (ICLR)**, 2024.

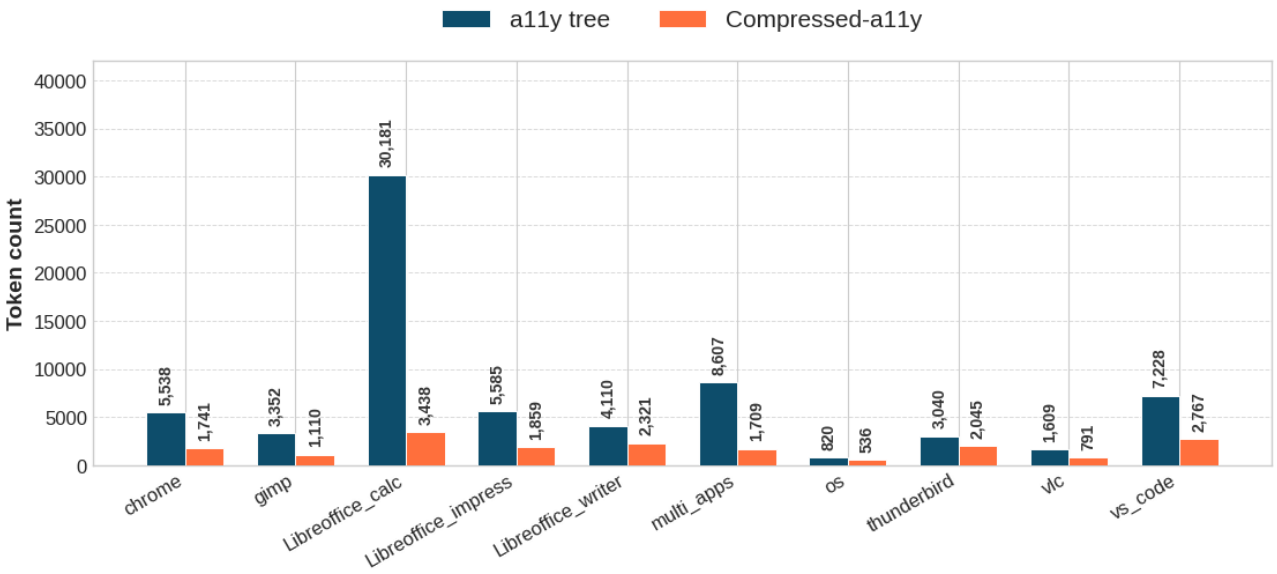

図 3: a11y tree（線形化）と Compressed-a11y のトークン数

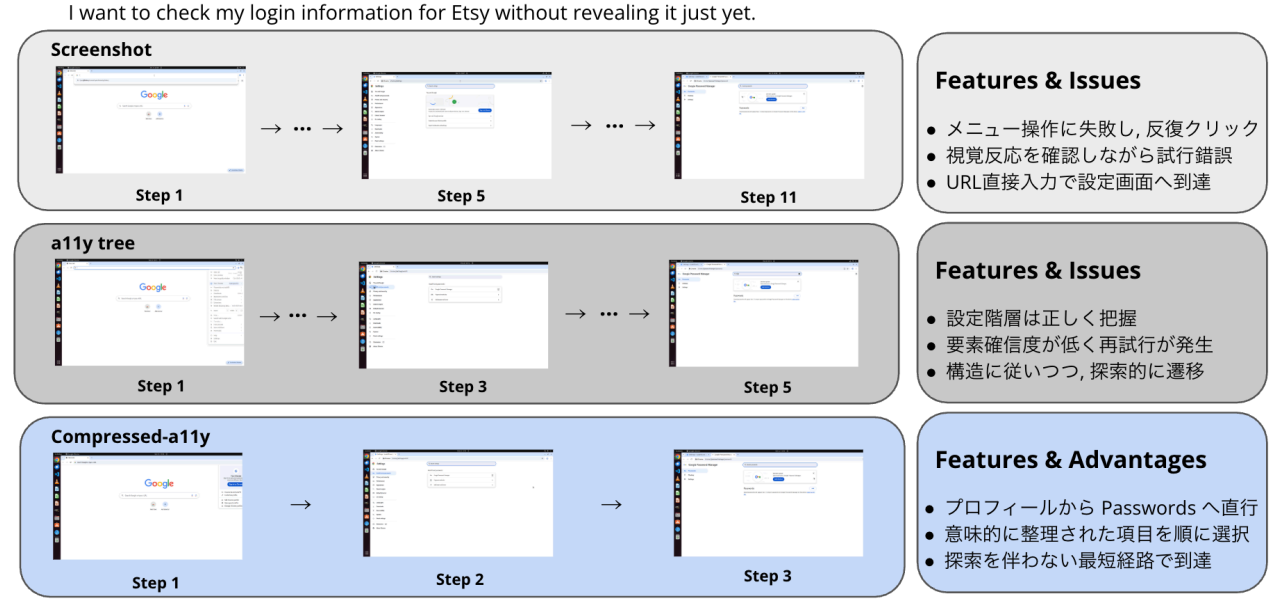

図 4: ブラウザにおける保存済みパスワード設定への到達プロセス比較（Screenshot / a11y / Compressed-a11y）

A　付録

図 3 は，線形化した a11y tree と Compressed-a11y における入力トークン数の比較を示す．Compressed-a11y はすべてのドメインにおいてトークン数を削減していることが確認できる．

図 4 は，ブラウザのパスワード管理画面へ到達するまでの行動過程を Compressed-a11y，線形・トリム化 a11y tree，およびスクリーンショットで比較した例である．Compressed-a11y では，意味的に整理された設定項目に基づき，探索を伴わずに最短経路で到達した．一方，a11y tree およびスクリーンショットでは，再試行や試行錯誤を含む探索的行動が観察された．