Measuring Interface Similarity: Computing a more Perceptual Distance Between Graphical User Interfaces

Raphaël Perraud, Sylvain Malacria

Measuring Interface Similarity: Computing a more Perceptual Distance Between Graphical User Interfaces
Mesurer la Similarité des Interfaces : Calculer une Distance Plus Perceptuelle Entre Interfaces Utilisateur Graphiques

Raphaël Perraud
Univ. Lille, Inria, CNRS, Centrale Lille, UMR 9189 CRIStAL
Lille, France
raphael.perraud@inria.fr

Sylvain Malacria
Univ. Lille, Inria, CNRS, Centrale Lille, UMR 9189 CRIStAL
Lille, France
sylvain.malacria@inria.fr

Abstract

When facing a new Graphical User Interface (GUI), users compare it with familiar interface and try to transfer knowledge between interfaces. However, existing methods for computing interface similarity often rely on black-box models or structural heuristics, limiting their interpretability and generalizability. In this paper, we introduce a user-centered approach for computing perceptual interface distance, leveraging key dimensions such as layout, semantics, and spatial relationships. Our method achieves greater transparency while maintaining accuracy, as demonstrated through benchmarking against CNN-based and structure-driven similarity models. These findings contribute to HCI research by providing a systematic, explainable framework for assessing interface similarity, supporting knowledge transfer across interfaces and software and UI design consistency.

Résumé

Lorsqu'ils sont confrontés à une nouvelle interface utilisateur graphique, les utilisateurs la comparent à des interfaces familières afin d'estimer leur degré de similitude pour déterminer les connaissances qui peuvent être transférées. Le développement de méthodes qui reflètent cette similarité perçue pourrait aider à prédire dans quelle mesure des connaissances sont transférables d'une interface à l'autre. Cependant, les méthodes existantes de calcul de similarité des interfaces reposent souvent sur des modèles de boîte noire ou des heuristiques structurelles, ce qui limite à la fois leur interprétabilité et leur généralisation. Nous proposons une approche centrée utilisateur pour estimer la distance perceptuelle entre interfaces, en exploitant disposition, sémantique et relations spatiales des éléments. Notre méthode assure transparence et précision, et offre un framework explicable pour évaluer la similarité des interfaces et améliorer le transfert de connaissances et la cohérence de conception.

CCS Concepts

• **Human-centered computing** → **Heuristic evaluations**; *HCI theory, concepts and models.*

Keywords

user interface, interface similarity, interface distance

IHM'25, Toulouse, France
2025. ISBN XXX-X-XXXX-XXXX-X/XX/XX

Mots clés

interface utilisateur, similarité entre interfaces, distance entre interfaces

Reference Format:
Raphaël Perraud and Sylvain Malacria. 2025. Measuring Interface Similarity: Computing a more Perceptual Distance Between Graphical User Interfaces. In *IHM'25 : Actes étendus de la 36^{e} conférence Francophone sur l'Interaction Humain-Machine, Novembre 4–7, 2025, Toulouse, France.* 9 pages.

1 Introduction

Graphical User Interfaces (GUIs) vary significantly across software, differing in layout, structure, metaphor, terminology, and visual representations. As a result, upon launching an unfamiliar software, users must interpret its GUI to interact with it efficiently [20, 34] by infering the behavior of the interface, the purpose of avaible features, their interactions from her prior knowledge [26, 57]. To do so, users assess both similarities and differences between new and familiar interfaces to determine the knowledge they can effectively transfer [44, 54] and reason adequatly about the interface. The differences that may arise between the familiar interface and the unfamiliar one impact how easily users transfer knowledge from one interface to another.

This process of estimating the perceptual gap between two interfaces, which we refer to as *interface distance* in this work, poses a cognitive challenge for users as it requires drawing parallels between their mental models [3, 54], relocate feature across interfaces [6], translate vocabulary incoherences [39]. Indeed, spatial mismatches can hinder user's ability to transfer knowledge effectively [50], as well as vocabulary inconsistency across software which results in different terminologies for similar tools [51]. When facing different interfaces, users perform *technical reasoning* to infer the purpose of features from existing knowledge [55]. This process aim to assess the similarity of a new tool to familiar ones, applying conceptual knowledge gained from prior experience to infer how the new tool should be used [42]. For example, the "free space" principle of Figma is also shared with multiple others software such as Miro, which allow an experienced Figma user to infer how to navigate in the Miro workspace, drawing similarity between these two software while having different purpose.

Estimating how users perceive *interface distance* is crucial for designing consistent interfaces when migrating software or developing new tools with shared functionalities. Understanding this distance helps predict how users will navigate unfamiliar designs

based on prior knowledge and assess how usability updates in GUIs impact users' perception of change. Existing computational methods for interface similarity have key limitations. The main approach in HCI has been to leverage neural networks with fine-tuning to compute interface similarity [7, 8, 17, 18, 21], as they can capture complex features, including human perceptual similarity [65]. However, these models often function as "black boxes" [49], making their results difficult to replicate and interpret for decision-making. Other approaches focus on structural similarity [11, 15, 16, 23, 64], particularly in web interfaces, where structural and hierarchical data is more readily accessible rather than focusing on what users actually perceive, making them less effective in predicting knowledge transfer. To bridge this gap, we propose a refinement of existing metrics that incorporates user perception for computing *interface distance*, offering a more transparent, generalizable, and interpretable method for computing interface distance. This approach enables designers to identify and interpret key dimensions of interface similarity, guiding more informed design decisions.

In this paper, we introduce a method for computing the perception of *interface distance* using human assessments, ensuring transparency and reproducibility across software interfaces. We then benchmark it against existing methods, and discuss the results. Our findings aim to inform the design of GUIs that better support knowledge transfer across software by providing a computational method to estimate how users will perceive the *distance* between two interfaces.

2 Related work on computing interface similarity

When encountering a new GUI, users interpret on-screen elements by inferring tool functions based on their prior knowledge of familiar interfaces and the similarities they recognize between the new and known systems. This section reviews various approaches for estimating how users perceive interface similarity.

2.1 Quantifying the distance between images of interfaces

Computer vision has long explored techniques for recognizing similarity between images, particularly for object detection and scene matching. Early approaches relied on visual features to detect and compare visual patterns, such as Scale-Invariant Feature Transform (SIFT) [33], edge based orientation algorithms [47], low dimensional representation of a scene [41] and color histograms [56]. Later, deep learning became the standard approach in computer vision, enabling models to automatically learn high-level features that are more robust to variations in lighting, scale, and perspective [48]. Inspired by these advances, HCI research has applied similar methods to compare graphical user interfaces (GUIs), aiming to quantify interface similarity based on visual and structural attributes. Building on the established fact that deep features seems to be very accurate to assess the visual similarity between the images of two interfaces [65], standard state-of-the-art techniques in computer vision uses CNN (Convolutional Neural Network) [17] or GCN [36] (Graph Convolutional Network) to get objective measurements from GUIs [15, 21] usable to compute *interface distance*. However, these models suffer from a fundamental limitation: they operate as "*black boxes*" [49] which make it difficult to interpret the features they learn. In practice, they can provide a numerical similarity score, but without insight into the underlying dimensions of comparison. This lack of transparency makes it challenging to use these models for actionable UI improvements or to explain why two interfaces are considered similar or different. Moreover, reproducibility is a major issue: many deep learning models require large, proprietary training datasets, which are often unavailable or inconsistent across studies, making independent verification difficult. Finally, these methods rely solely on system-inferred similarities, without incorporating user perception, meaning they assess what an algorithm deems similar, not necessarily what users actually perceive as similar. On the other hand, some studies have attempted to correlate users' affective judgments with low-level image statistics extracted from website layout structures, demonstrating that these metrics can approximate participants' perception of visual complexity [68]. This approach was followed up by Reinecke et al. who addressed the lack of perceptual models by proposing a model that assess the visual complexity and colorfulness to predict the judgment of website aesthetics [52, 53].

2.2 Computing *interface distance* using models that support GUI matching

2.2.1 From app hierarchy. Prior research in this area have relied on app metadata or app hierarchies to assess interface similarityat the application or screen level. NEAR uses the DOM tree in the web context to detect near-duplicate pages on the web [64]. Zhang et al. [67] used a set of screen equivalency heuristics based on GUI identifiers and semantic structures found in Android screen hierarchies to determine screen equivalences for accessibility annotation. These heuristics leverage toolkit metadata, developer-provided indicators, and UI element properties to reliably identify screens that match predefined template screens. Multiple works have been using models to produce semantic representation of UI elements [13, 18, 31, 32, 36] enabling screen similarity computation using underlying view hierarchy information.

These methods are applied across various domains and serve different purposes; however, they lack the fine-grained precision needed to address the perception of similarity explored in this study, as they rely solely on structural descriptions of the interface rather than on how users actually perceive it.

2.2.2 From pixels. Malisa et al. [35] trained a model to compare a reference screen to the currently displayed screen to identify visual cues that are likely to be omitted by human viewer. In a similar fashion, Feiz et al. [18] proposed a screen similarity model able to recognize instances of a same screen by comparing a reference screen to a collection of screen. Screen Recognition [66] detects UI elements on iOS screens from on-screen pixels. While this work does not attempt to identify similar screens, it can provide data for screen similarity detection. Screen Parsing [62] and VINS [11] detect UI elements from pixels and infer their hierarchy for UI similarity search, primarily across different apps. These methods retrieve layout information from GUI screenshots but omit visual appearance cues that are essential for user's estimation of *interface distance*.

Computation method	Explainable	Reproductible	Estimate u.p.[1]	Use h.a.[2]
Deep networks	×	×	✓	×
Mixed (EMD + tree-edit distance)	✓	✓	×	×
Layout-based	✓	✓	×	×
User-centered *interface distance*	✓	✓	✓	✓

Table 1: Comparison between our user-centered approach for estimating user's perception of *interface distance* and related works.
[1] Estimate user perception. [2] Use human assessments for understanding of the underlying concepts of the interface (visual representation of features, expected behavior of the interface, vocabulary and semantic groupings).

Pixel-based approaches therefore analyse the same stimulus that a user perceives when confronted with a new interface for the first time and tries to find similarities with familiar interfaces, but do not attempt to model the way in which the user will identify these similarities and estimate a distance between two interfaces.

3 A user-centered approach on estimating perceived interface similarity

To compute *interface distance*, we chose to follow prior approaches which decomposed the studied concept into multiple underlying dimensions [37]. This section discusses the selected dimensions of *interface distance* based on the literature and the approach to compute a score of *distance* between two interfaces.

3.1 Perceiving interface similarity: a user-centered approach beyond deep visual representations

The effectiveness of deep features in assessing perceptual similarity between images [65] could be debatable for interfaces, as GUIs combine static visuals with reconfigurable elements and interactive potential [43], particularly in feature-rich software [28]. We argue that a more fine-grained approach, incorporating user-centered perceptual metrics, can better estimate the *distance* users perceive between two interfaces (Table 1 summarize the pros and cons of existing methods). To our knowledge, no research has explicitly examined users' perception of *interface distance* between GUIs, particularly at first exposure. Addressing this gap is crucial for estimating how users' will perceive a new interface as familiar and be able to transfer knowledge.

In parallel problems, such as visual complexity, researchers have often decomposed the concept into multiple underlying dimensions [4, 37]. Similarly, since interface comparison relies on the memorization of visuo-spatial information [10], prior research on visuo-spatial working memory (VSWM) can help identify key components of *interface distance*. Uddin and Gutwin [61] highlighted four crucial aspects that users rely on for memorizing and comparing interfaces: (1) *interface layout* (the arrangement of screen regions), (2) *command groups* (clusters of thematically related commands), (3) *icon visual appearance* (which convey tool functionality), and (4) the *relative positioning* of UI elements based on spatial landmarks such as corners and edges. These findings align with VSWM research, which suggests that applying Gestalt principles of grouping (connectedness, similarity, and proximity) [45] enhances interface memorization. Indeed, users tend to organize information into spatially structured chunks [25, 45] rather than processing GUIs feature by feature, enabling comparisons that are either holistic or focused on a chunk-by-chunk basis.

In the case of GUIs, image similarity computation alone is insufficient to capture how users perceive interface similarity. Instead, we should identify the key dimensions of *interface distance* that users rely on to recognize similarities and differences between interfaces, to better estimate user's capabilities to transfer knowledge from familiar to new interfaces.

3.2 Dimensions of *interface distance*

Our selection of dimensions as components of *interface distance* is grounded in prior research on visuo-spatial working memory (VSWM) and knowledge transfer across GUIs. Users primarily rely on structured spatial memory to compare interfaces, leveraging key visual and structural attributes to recognize similarities and differences [61]. The LAYOUT ARRANGEMENT and the CLUSTERING of commands are fundamental, as users organize information into spatial chunks rather than analyzing individual elements in isolation [25, 45]. SPATIAL POSITIONING of UI elements serves as a critical navigational aid, reinforcing recognition and recall of interface elements [61].

Beyond spatial organization, FEATURE RECOGNITION is facilitated by the visual appearance of icons and their text labels, which convey tool functionality through familiar cues. Similarity between icons can further support "technical reasoning", enabling users to infer a tool's purpose. [54]. Interface MODULARITY (i.e., the freedom to manually rearrange UI elements) has been shown to play a role [14, 61], as it can alter perceived complexity and hinder knowledge transfer by disrupting learned spatial relationships [59]. LANGUAGE influences perception by affecting users' ability to understand interface labels and efficiently transfer knowledge across interfaces [40, 63]. Finally, as our focus is on the first exposure to a new interface, we also included COLOR SCHEME as prior research has demonstrated its usefulness to assess first impressions similarity [53, 56].

By considering these seven dimensions, we move beyond purely visual similarity to a user-centered approach that accounts for how individuals perceive interface similarities and differences to trigger adequate knowledge.

4 Computing an *interface distance* metric (Δ)

We quantify the difference between two GUIs, $i1$ and $i2$ as a multi-dimensional vector over characteristics identified in prior work. As a running example throughout this section, we illustrate the computations on five desktop GUIs: Affinity Designer as reference,

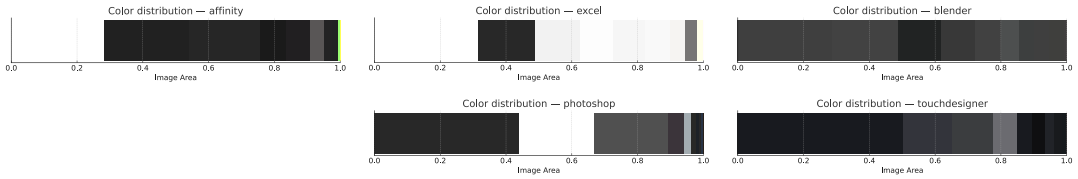

Figure 1: Plot of the color's distribution as fraction of the X-axis for each interface, inspired by [9]

and Photoshop, Excel, TouchDesigner, and Blender as targets (see Figure 4).

4.1 Computation methods per characteristic

Considering a pair of different GUIs i_1 and i_2, we model the difference between these two GUI in a multi-dimensional vector calculated from the different characteristics we identified from the literature. Building on existing methods used in the literature, we propose computational approaches for each dimension of *interface distance*: ColorScheme, Spatialization, Modularity, ReadingFlow, Clustering, FeatureRecognition, Context, OS, and Language. The approach combines automated and manual assessments, normalizing all scores to a consistent 0 (maximally similar) to 1 (maximally dissimilar) scale for comparability.

4.1.1 Color scheme. Differences in color schemes can be evaluated using the HSV histograms of a screen capture of each interface and the Earth Mover's Distance to quantify the difference between the two distributions [56]. The score is normalized using the following formula:

$$\Delta_{ColorScheme} = \frac{\text{EMD}}{max(\text{EMD})} \tag{1}$$

where $max(EMD) = \sqrt{(h_{\text{bins}} - 1)^2 + (s_{\text{bins}} - 1)^2}$ using both (h) hue and (s) saturation chanels.

4.1.2 Spatialization. The spatialization difference is calculated using the approximate mean size-weighted absolute Euclidean distance (d) [16] between the positions of all remappable components of an interface i_1 to a i_2 interface (components of similar semantic value [30] (e.g., toolbars, workspace, panels...) or similar features such as font size, color picker...), normalized by the maximum diagonal length of the biggest screen $max(Diagonal)$ (between i_1 and i_2):

$$\Delta_{Spatialization} = \frac{\frac{1}{N}\sum_{c=1}^{N}\left(\frac{\text{Area of component } c}{\text{Total area of all components in } i_1} \cdot d_c\right)}{\max(Diagonal)} \tag{2}$$

where N is the total number of remappable components and d_c the Euclidean distance between the component c in i_1 and i_2.

4.1.3 Modularity. The modularity score considers the difference of the degree of independence of interface components. A user's perception of an interface's modularity directly influences their reliance on spatial memory and layout landmarks when comparing interfaces [44]. As modularity increases, users are more likely to assess interfaces based on their customizability and adaptability to individual workflows [61]. The modularity score of each interface must thus first be calculated, so that they can be subtracted from each other to obtain a modularity difference score. Users assess the possibility of rearranging the interface by moving/resizing/hiding components independently. So for each interface set of components ($\mathbb{C}$), components are classified as independent (c_i) (can be freely moved or resized inside or outside the main window of the software), semi-independent (c_s) (can be rearranged with constraints (e.g., dockable panels)), or fixed (c_f) (cannot be moved or resized) based on their ability to be moved or resized. The modularity score is scaled from 0 (fixed interface) to 1 (fully modular interface):

$$\Delta_{Modularity} = \frac{(0 \cdot |c_f|) + (0.5 \cdot |c_s|) + (1 \cdot |c_i|)}{|\mathbb{C}|} \tag{3}$$

4.1.4 Layout arrangement. We approximate layout structure with a grid decomposition [2], partitioning each interface into a grid detailled enough to capture how users will interpret different regions of the interface. As example, we defined 12x8 grids for our interfaces sized of 2048x1168 px. Cells are indexed in scan order to reflect typical visual scanning patterns. This could be done using eye tracking systems as we did to plot Figure 3 using a sequence analysis or manually rating by informants, or, failing that, relying on expected order of viewing based on tailored eye scan patterns (e.g., F-pattern for text-heavy and content-dense interfaces [**?**], Z-pattern otherwise [5], Guttenberg diagram for asymmetrical layouts [22]). We compute sequence alignment using the Needleman-Wunsch algorithm [38] evaluates the similarity of the two patterns using a basic scoring system (+1 for match, -1 for mismatch, insertion or deletion) and normalize the output:

$$\Delta_{LayoutArrangement} = \frac{\delta}{16} \tag{4}$$

4.1.5 Clustering. Clusters of interface components are defined by spatial proximity and functional similarity, grouping together elements that serve related or complementary purposes (e.g., text-editing tools or color-selection controls). To assess whether such clusters are preserved across interfaces, we collect sets of matched features and groups between $i1$ and $i2$, evaluate both their overlap and the hierarchical organization in which these elements are embedded. While interface hierarchies can sometimes be extracted from software accessibility APIs [19], these descriptions are often incomplete or inconsistently populated. As a result, manual decomposition can be more reliable (see Figure 4). The outcome of this process is a named tree representation, in which groups and individual elements such as features, tools, parameters, and buttons are denoted, with each leaf explicitly labeled. Similarly to clustering analysis used for text datasets [60], we use the Jaccard community coefficient [24], which measures the overlap of clusters between two interfaces i_1 and i_2 is defined as the size of the intersection

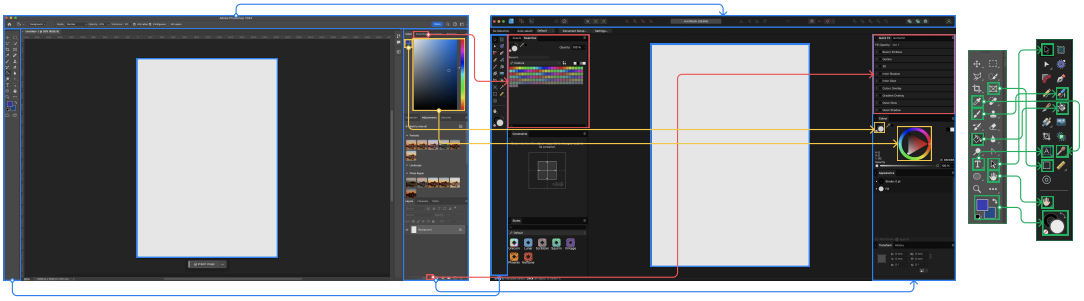

Figure 2: An example of Spatialization difference computation between the Adobe Photoshop (on the left) and the Affinity Designer (on the right) interfaces. Remappable elements include similar 1) panels and menu (in blue), 2) feature groups (in red), 3) parameters (in yellow) and 4) tools that produce similar outcomes (in green). For clarity, toolbars were visually extracted and highlighted separately.

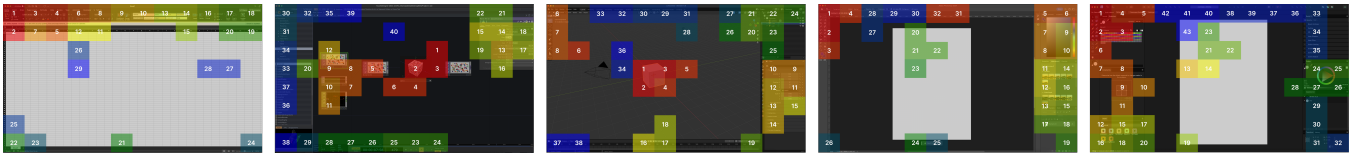

Figure 3: Left-to-right: mean gaze-sequence maps (average visit order per grid cell) for Excel, TouchDesigner, Blender, Photoshop, and Affinity Designer. Lower ranks indicate earlier fixations. This sequence maps are used to compute the layout arrangement delta.

divided by the size of the union of the sets:

$$\Delta_{Clustering} = \frac{|i_1 \cap i_2|}{|i_1 \cup i_2|} = \frac{|i_1 \cap i_2|}{|i_1| + |i_2| - |i_1 \cap i_2|} \tag{5}$$

4.1.6 Feature Recognition. This dimension requires to name every visual feature (icon, image or graphical item) and convert labels into vector representations using a word embeddings technique. We recommend to use a model that handles subword information and Out-of-Vocabulary situations such as FastText [1] trained on a GUI related dataset [27]. For each vector of the smaller set, find the closest corresponding vector in the other set using the cosine similarity $cos(\theta)$. The median cosine similarity will represent the FeatureRecognition score.

$$\Delta_{FeatureRecognition} = 1 - \frac{cos(\theta)_{Median} + 1}{2} \tag{6}$$

4.1.7 Language. Multilingual studies proposed several methods to measure a linguistic distance based on the proficiency to learn another language but we will be focusing on a method based on a Levenstein measure [29], which seems to be one of the most robust for general language classification without language family a priori [12]. We adopt the approach of Wichmann et al. [46] to compute normalized Levenshtein distances. First, retrieve the related labels and text content from the two interfaces and organize them as paired sequences of related terms (e.g., "Brush", "Eraser", "Circle" and "Cepillo", "Borrador", "Círculo"). The linguistic distance is then calculated using Levenshtein distance (LD), normalized by the length of the longer of the two sequences (LDN), ensuring comparability across sequences of varying lengths. To account for accidental similarities arising from shared random features (e.g., common labels such as "Close" in unrelated interface components), we compute the background similarity score, Γ, as the average LDN between randomly paired, unrelated text content across the two interfaces. The final linguistic distance between the interfaces, $\Delta_{Language}$, is then computed as the LDN between conceptually matched components divided by Γ:

$$\Delta_{Language} = \frac{\overline{LDN}}{\Gamma} = \frac{\frac{1}{n}\sum_{i=1}^{n} LD(s_{1,y}, s_{2,y})}{\max(len(s_{1,y}), len(s_{2,y}))} \Big/ \Gamma \tag{7}$$

Here, $s_{1,y}$ and $s_{2,y}$ represent the i-th paired sequences from the two interfaces, n is the total number of paired sequences, and $len(s)$ indicates the length of a sequence s.

4.2 Towards a unified scalar representation of *interface distance*

The perceptual *interface distance* between interfaces can be aggregated into a single scalar value by computing the mean across all dimensions. However, we recognize that a simple mean may not accurately capture the relative importance of each dimension in user perception. Determining appropriate weighting coefficients for each dimension remains an open challenge, which we leave to future work. A more refined approach could involve empirical studies to establish optimal weightings that better align with users' perception of interface similarity. The distance (Δ) between two interfaces can be represented as:

$$\Delta = \sum_{f}^{U} W_d \cdot S_d \tag{8}$$

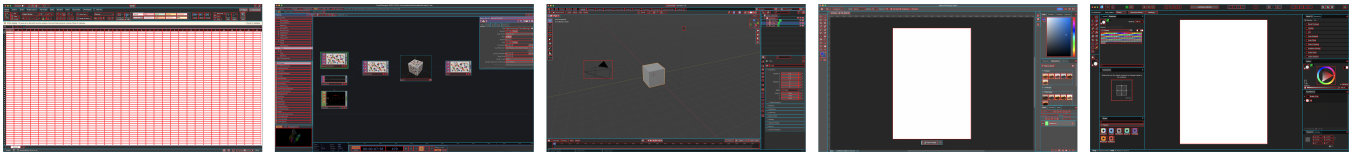

Figure 4: Left-to-right: decomposition trees used to define the clusters for Excel, TouchDesigner, Blender, Photoshop, and Affinity Designer. Groups are denoted in blue, individual elements in red.

where W indicates the weight of the d dimension (so $W_d = \frac{1}{|d|} = \frac{1}{7}$) and S the value obtained for this dimension according to the methodology declined in the following section.

Figure 5 summarizes per-dimension differences and the overall *interface distance* for the four targets relative to Affinity Designer. As expected, Spatialization and Modularity yield smaller distances for Photoshop than for the other interfaces, reflecting its shared domain as a graphics editor with the reference interface (Affinity Designer focuses on vector graphics, while Photoshop is primarily bitmap-based) and their reliance on comparable workspace configurations. In contrast, the proportion of features recognizable through visual cues remains relatively stable across all interfaces, as indicated by the similar values obtained on FeatureRecognition. All interfaces display large deltas on Clustering, underscoring substantial differences in how functions are hierarchized and grouped. These patterns illustrate why a multidimensional approach is needed: structural similarity alone would miss salient visual and lexical differences, and vice versa.

While we present comparisons to a single reference interface for conciseness, this computation does not capture how the target interfaces relate to one another. An avenue for future work would be to derive a similarity space from the difference scores, since each score reflects only relative distances to a reference. Such a representation could support cross-comparisons and provide a deeper understanding of the similarities within a set of interfaces.

5 Discussion and future works

This work proposes a set of perceptually grounded metrics for computing interface distance, derived from dimensions identified in prior literature. The method offers a transparent and interpretable framework for assessing perceived differences between interfaces, providing actionable insights for the design of GUIs. Building on the premise that interface comparison inherently relies on the memorization of one interface to facilitate comparison with another, we identified key interface characteristics from the visuo-spatial working memory (VSWM) literature that aid users in memorization and, consequently, in comparison. These characteristics are: ColorScheme, Layout Arrangement, Modularity, Spatial Positioning of UI elements, Language, and Icon Appearance. They form the core dimensions of perceptual interface distance from a user-centered perspective.

While these dimensions are theoretically motivated and implemented using interpretable computations, the method currently assumes equal importance across all dimensions. At this stage, we do not propose a validated weighting scheme, as determining the relative contribution of each dimension remains an open question. A next step is to empirically assess how users perceive these differences and whether certain dimensions should be given greater weight depending on context or task. Such work would require dedicated user studies to validate these metrics by comparing computed distance scores with ground truth derived from participants' explicit judgments of interface similarity.

Our approach offers a transparent alternative to existing deep learning-based models that are know to compute effectively similarity using a black-box representation, producing undifferentiated scalar scores that lack interpretability. A benchmark comparing our method to state-of-the-art models such as CNNs would help clarify the value of our multidimensional, discriminative representation. This comparison would be particularly valuable in the context of software interfaces, where variability across dimensions is likely more pronounced. Most existing CNN-based models and datasets focus on web interfaces, which may limit the relevance of such comparisons due to the growing homogenization of web design over time, as shown by Goree et al [21]. To our knowledge, no publicly available dataset currently exists for desktop software interfaces, making this step more challenging but also potentially more informative for evaluating perceptual interface distance in more diverse and heterogeneous environments. In parallel, future work could derive a similarity space from the difference scores, analogous to a model's latent space, allowing cross-comparisons between multiple interfaces and revealing the broader structure of their relationships.

Finally, a possible next step would be to extend this work beyond static interfaces to account for the inherently interactive nature of GUIs, which evolve in response to user input. This could involve computing perceptual distance between workflows, for instance by analyzing and comparing command sequences [58]. Alternatively, this line of work could be extended at a more abstract level by investigating knowledge transfer across software through system-wide tasks and functionalities, as a way to model users' evolving mental representations of software interfaces [26].

Beyond the technical evaluation, this work raises broader questions for the HCI community. Understanding how users perceive differences between interfaces could support adaptive interfaces by providing a quantifiable estimate of the cognitive effort required for users to adapt to changes. It may also contribute to the building of behavioral models of knowledge transfer across software tools and the study of learning across software tools. Furthermore, interface distance metrics could inform the design of interface updates by identifying elements likely to disrupt users' expectations or workflows. Similarly, designers working across a product suite (such as those from Adobe, Microsoft, Affinity, Autodesk, etc) may benefit

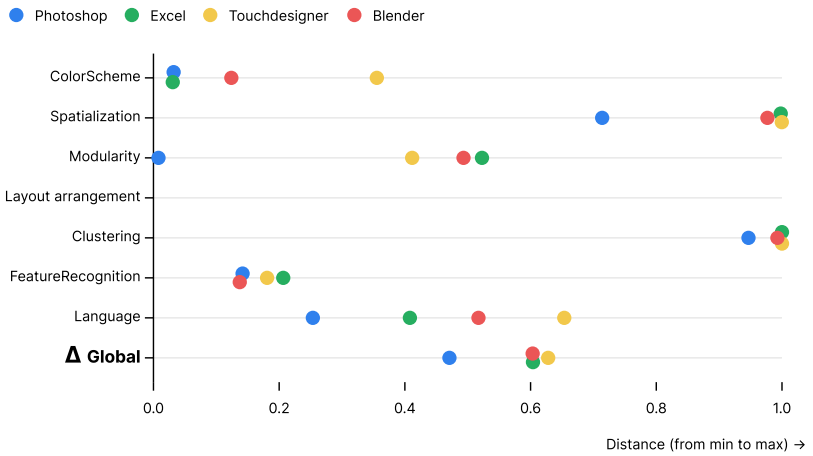

Figure 5: Results for all distance metrics and the overall *interface distance*score, comparing Photoshop, Excel, TouchDesigner, and Blender against Affinity Designer. Scores are normalized to [0,1]; larger scores denote larger deviations from the Affinity interface.

from tools that estimate perceptual consistency between newly designed and existing interfaces.

Taken together, this work outlines a methodological foundation for computing interface similarity in a perceptually transparent and explainable way. It opens avenues for empirical validation and raises conceptual questions on the role of similarity in adaptation, learning, and interface consistency. These directions would benefit from further discussion within the HCI community.

References

[1] [n. d.]. English Word Vectors · fastText. https://fasttext.cc/docs/en/english-vectors.html.

[2] [n. d.]. Using Grids in Interface Designs. https://www.nngroup.com/articles/using-grids-in-interface-designs/.

[3] Eytan Adar, Mira Dontcheva, and Gierad Laput. 2014. CommandSpace: Modeling the Relationships between Tasks, Descriptions and Features. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology*. ACM, Honolulu Hawaii USA, 167–176. doi:10.1145/2642918.2647395

[4] Eren Akça and Ömer Özgür Tanriöver. 2021. A Comprehensive Appraisal of Perceptual Visual Complexity Analysis Methods in GUI Design. *Displays* 69 (Sept. 2021), 102031. doi:10.1016/j.displa.2021.102031

[5] Julius Albiz, Olga Viberg, and Andrii Matviienko. 2023. Guiding Visual Attention on 2D Screens: Effects of Gaze Cues from Avatars and Humans. In *Proceedings of the 2023 ACM Symposium on Spatial User Interaction*. ACM, Sydney NSW Australia, 1–9. doi:10.1145/3607822.3614529

[6] Jessalyn Alvina, Andrea Bunt, Parmit K. Chilana, Sylvain Malacria, and Joanna McGrenere. 2020. Where Is That Feature?: Designing for Cross-Device Software Learnability. In *Proceedings of the 2020 ACM Designing Interactive Systems Conference*. ACM, Eindhoven Netherlands, 1103–1115. doi:10.1145/3357236.3395506

[7] Maxim Bakaev. 2018. Assessing Similarity for Case-Based Web User Interface Design. In *Digital Transformation and Global Society*, Daniel A. Alexandrov, Alexander V. Boukhanovsky, Andrei V. Chugunov, Yury Kabanov, and Olessia Koltsova (Eds.). Vol. 858. Springer International Publishing, Cham, 353–365. doi:10.1007/978-3-030-02843-5_28

[8] Maxim Bakaev, Vladimir Khvorostov, Sebastian Heil, and Martin Gaedke. 2017. Evaluation of User-Subjective Web Interface Similarity with Kansei Engineering-Based ANN. In *2017 IEEE 25th International Requirements Engineering Conference Workshops (REW)*. 125–131. doi:10.1109/REW.2017.13

[9] Anat Ben-David, Adam Amram, and Ron Bekkerman. 2018. The Colors of the National Web: Visual Data Analysis of the Historical Yugoslav Web Domain. *International Journal on Digital Libraries* 19, 1 (March 2018), 95–106. doi:10.1007/s00799-016-0202-6

[10] François Bérard and Amélie Rochet-Capellan. 2015. The Transfer of Learning as HCI Similarity: Towards an Objective Assessment of the Sensory-Motor Basis of Naturalness. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. ACM, Seoul Republic of Korea, 1315–1324. doi:10.1145/2702123.2702359

[11] Sara Bunian, Kai Li, Chaima Jemmali, Casper Harteveld, Yun Fu, and Magy Seif Seif El-Nasr. 2021. VINS: Visual Search for Mobile User Interface Design. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. ACM, Yokohama Japan, 1–14. doi:10.1145/3411764.3445762

[12] Elisa Cargnelutti, Barbara Tomasino, and Franco Fabbro. 2022. Effects of Linguistic Distance on Second Language Brain Activations in Bilinguals: An Exploratory Coordinate-Based Meta-Analysis. *Frontiers in Human Neuroscience* 15 (Jan. 2022). doi:10.3389/fnhum.2021.744489

[13] Sena Nur Cavsak, Aysu Deliahmetoglu, Berhan Turku Ay, and Senem Tanberk. 2023. GUI Component Detection Using YOLO and Faster-RCNN. In *2023 14th International Conference on Electrical and Electronics Engineering (ELECO)*. IEEE, Bursa, Turkiye, 1–5. doi:10.1109/ELECO60389.2023.10415929

[14] Andy Cockburn and Bruce McKenzie. 2002. Evaluating the Effectiveness of Spatial Memory in 2D and 3D Physical and Virtual Environments. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '02)*. Association for Computing Machinery, New York, NY, USA, 203–210. doi:10.1145/503376.503413

[15] Nathalia Da Cruz Alves, Leonardo Kreuch, and Christiane Gresse Von Wangenheim. 2022. Analyzing Structural Similarity of User Interface Layouts of Android Apps Using Deep Learning. In *Proceedings of the 21st Brazilian Symposium on Human Factors in Computing Systems*. ACM, Diamantina Brazil, 1–11. doi:10.1145/3554364.3559111

[16] Niraj Ramesh Dayama, Simo Santala, Lukas Brückner, Kashyap Todi, Jingzhou Du, and Antti Oulasvirta. 2021. Interactive Layout Transfer. In *26th International Conference on Intelligent User Interfaces*. ACM, College Station TX USA, 70–80. doi:10.1145/3397481.3450652

[17] Bardia Doosti, David J. Crandall, and Norman Makoto Su. 2017. A Deep Study into the History of Web Design. In *Proceedings of the 2017 ACM on Web Science Conference*. ACM, Troy New York USA, 329–338. doi:10.1145/3091478.3091503

[18] Shirin Feiz, Jason Wu, Xiaoyi Zhang, Amanda Swearngin, Titus Barik, and Jeffrey Nichols. 2022. Understanding Screen Relationships from Screenshots of Smartphone Applications. In *27th International Conference on Intelligent User Interfaces*. ACM, Helsinki Finland, 447–458. doi:10.1145/3490099.3511109

[19] Christian Frisson, Sylvain Malacria, Gilles Bailly, and Thierry Dutoit. 2016. InspectorWidget: A System to Analyze Users Behaviors in Their Applications. In *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems*. ACM, San Jose California USA, 1548–1554. doi:10.1145/2851581.2892388

[20] Wai-Tat Fu and Wayne D. Gray. 2004. Resolving the Paradox of the Active User: Stable Suboptimal Performance in Interactive Tasks. *Cognitive Science* 28, 6 (Nov. 2004), 901–935. doi:10.1207/s15516709cog2806_2

[21] Samuel Goree, Bardia Doosti, David Crandall, and Norman Makoto Su. 2021. Investigating the Homogenization of Web Design: A Mixed-Methods Approach. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. ACM, Yokohama Japan, 1–14. doi:10.1145/3411764.3445156

[22] Bruce Hillard, Jocelyn Amarego, and Tanya McGill. 2016. Optimising Visual Layout for Training and Learning Technologies. *27th Australasian Conference on Information Systems* (Dec. 2016).

[23] Yangyu Hu, Guosheng Xu, Bowen Zhang, Kun Lai, Guoai Xu, and Miao Zhang. 2020. Robust App Clone Detection Based on Similarity of UI Structure. *IEEE Access* 8 (2020), 77142–77155. doi:10.1109/ACCESS.2020.2988400

[24] Paul Jaccard. 1912. The Distribution of the Flora in the Alpine Zone. *New Phytologist* 11, 2 (Feb. 1912), 37–50. doi:10.1111/j.1469-8137.1912.tb05611.x

[25] Patrycja Kałamała, Aleksandra Sadowska, Wawrzyniec Ordziniak, and Adam Chuderski. 2017. Gestalt Effects in Visual Working Memory. *Experimental Psychology* (Feb. 2017).

[26] Neung Eun Kang and Wan Chul Yoon. 2005. A Cognitive Modeling of the User's Exploratory Behavior with Prior Knowledge. In *Proceedings of the 4th International Workshop on Task Models and Diagrams (TAMODIA '05)*. Association for Computing Machinery, New York, NY, USA, 35–42. doi:10.1145/1122935.1122943

[27] Farideh Khalili, Ali Mohebbi, Valerio Terragni, Mauro Pezzè, Leonardo Mariani, and Abbas Heydarnoori. 2022. The Ineffectiveness of Domain-Specific Word Embedding Models for GUI Test Reuse. In *Proceedings of the 30th IEEE/ACM International Conference on Program Comprehension*. ACM, Virtual Event, 560–564. doi:10.1145/3524610.3527873

[28] Benjamin Lafreniere, Andrea Bunt, and Michael Terry. 2014. Task-Centric Interfaces for Feature-Rich Software. In *Proceedings of the 26th Australian Computer-Human Interaction Conference on Designing Futures: The Future of Design*. ACM, Sydney New South Wales Australia, 49–58. doi:10.1145/2686612.2686620

[29] Vladimir I Levenshtein et al. 1966. Binary Codes Capable of Correcting Deletions, Insertions, and Reversals. In *Soviet Physics Doklady*, Vol. 10. Soviet Union, 707–710.

[30] Loet Leydesdorff and Kasper Welbers. 2011. The Semantic Mapping of Words and Co-Words in Contexts. *Journal of Informetrics* 5, 3 (July 2011), 469–475. doi:10.1016/j.joi.2011.01.008

[31] Toby Jia-Jun Li, Lindsay Popowski, Tom Mitchell, and Brad A Myers. 2021. Screen2Vec: Semantic Embedding of GUI Screens and GUI Components. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. ACM, Yokohama Japan, 1–15. doi:10.1145/3411764.3445049

[32] Thomas F. Liu, Mark Craft, Jason Situ, Ersin Yumer, Radomir Mech, and Ranjitha Kumar. 2018. Learning Design Semantics for Mobile Apps. In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology*. ACM, Berlin Germany, 569–579. doi:10.1145/3242587.3242650

[33] D.G. Lowe. 1999. Object Recognition from Local Scale-Invariant Features. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, Vol. 2. 1150–1157 vol.2. doi:10.1109/ICCV.1999.790410

[34] Eva Mackamul, Géry Casiez, and Sylvain Malacria. 2023. Exploring Visual Signifier Characteristics to Improve the Perception of Affordances of In-Place Touch Inputs. *Proceedings of the ACM on Human-Computer Interaction* 7, MHCI (Sept. 2023), 1–32. doi:10.1145/3604257

[35] Luka Malisa, Kari Kostiainen, and Srdjan Capkun. 2017. Detecting Mobile Application Spoofing Attacks by Leveraging User Visual Similarity Perception. In *Proceedings of the Seventh ACM on Conference on Data and Application Security and Privacy*. ACM, Scottsdale Arizona USA, 289–300. doi:10.1145/3029806.3029819

[36] Dipu Manandhar, Dan Ruta, and John Collomosse. 2020. Learning Structural Similarity of User Interface Layouts Using Graph Networks. In *Computer Vision – ECCV 2020*, Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm (Eds.). Springer International Publishing, Cham, 730–746. doi:10.1007/978-3-030-58542-6_44

[37] Aliaksei Miniukovich, Simone Sulpizio, and Antonella De Angeli. 2018. Visual Complexity of Graphical User Interfaces. 1–9. doi:10.1145/3206505.3206549

[38] Saul B. Needleman and Christian D. Wunsch. 1970. A General Method Applicable to the Search for Similarities in the Amino Acid Sequence of Two Proteins. *Journal of Molecular Biology* 48, 3 (March 1970), 443–453. doi:10.1016/0022-2836(70)90057-4

[39] David G. Novick, Oscar D. Andrade, and Nathaniel Bean. 2009. The Micro-Structure of Use of Help. In *Proceedings of the 27th ACM International Conference on Design of Communication - SIGDOC '09*. ACM Press, Bloomington, Indiana, USA, 97. doi:10.1145/1621995.1622014

[40] Cristina Olaverri-Monreal, Christoph Draxler, and Klaus-Josef Bengler. 2011. Variable Menus for the Local Adaptation of Graphical User Interfaces. In *6th Iberian Conference on Information Systems and Technologies (CISTI 2011)*. 1–6.

[41] Aude Oliva and Antonio Torralba. 2001. Modeling the Shape of the Scene: A Holistic Representation of the Spatial Envelope. *International Journal of Computer Vision* 42, 3 (May 2001), 145–175. doi:10.1023/A:1011139631724

[42] François Osiurak, Christophe Jarry, Philippe Allain, Ghislaine Aubin, Frédérique Etcharry-Bouyx, Isabelle Richard, Isabelle Bernard, and Didier Le Gall. 2009. Unusual Use of Objects after Unilateral Brain Damage. The Technical Reasoning Model. *Cortex* 45, 6 (June 2009), 769–783. doi:10.1016/j.cortex.2008.06.013

[43] Antti Oulasvirta, Per Ola Kristensson, Xiaojun Bi, and Andrew Howes. 2018. *Computational Interaction*. Oxford University Press.

[44] Raphaël Perraud, Aurélien Tabard, and Sylvain Malacria. 2024. Tutorial Mismatches: Investigating the Frictions Due to Interface Differences When Following Software Video Tutorials. In *ACM Conference on Designing Interactive Systems (DIS 2024)*. Copenhagen, Denmark. doi:10.1145/3643834.3661511

[45] Dwight J. Peterson and Marian E. Berryhill. 2013. The Gestalt Principle of Similarity Benefits Visual Working Memory. *Psychonomic Bulletin & Review* 20, 6 (Dec. 2013), 1282–1289. doi:10.3758/s13423-013-0460-x

[46] Filippo Petroni and Maurizio Serva. 2010. Measures of Lexical Distance between Languages. *Physica A: Statistical Mechanics and its Applications* 389, 11 (June 2010), 2280–2283. doi:10.1016/j.physa.2010.02.004

[47] António M.G. Pinheiro. 2009. Image Descriptors Based on the Edge Orientation. In *2009 Fourth International Workshop on Semantic Media Adaptation and Personalization*. 73–78. doi:10.1109/SMAP.2009.27

[48] Moacir Antonelli Ponti, Leonardo Sampaio Ferraz Ribeiro, Tiago Santana Nazare, Tu Bui, and John Collomosse. 2017. Everything You Wanted to Know about Deep Learning for Computer Vision but Were Afraid to Ask. In *2017 30th SIBGRAPI Conference on Graphics, Patterns and Images Tutorials (SIBGRAPI-T)*. 17–41. doi:10.1109/SIBGRAPI-T.2017.12

[49] Zhuwei Qin, Fuxun Yu, Chenchen Liu, and Xiang Chen. 2018. How Convolutional Neural Networks See the World — A Survey of Convolutional Neural Network Visualization Methods. *Mathematical Foundations of Computing* 1, 2 (May 2018), 149–180. doi:10.3934/mfc.2018008

[50] Reyhaneh Raissi, Evanthia Dimara, Jacquelyn H. Berry, Wayne D. Gray, and Gilles Bailly. 2020. Retroactive Transfer Phenomena in Alternating User Interfaces. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. ACM, Honolulu HI USA, 1–14. doi:10.1145/3313831.3376538

[51] Vidya Ramesh, Charlie Hsu, Maneesh Agrawala, and Björn Hartmann. 2011. ShowMeHow: Translating User Interface Instructions between Applications. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*. ACM, Santa Barbara California USA, 127–134. doi:10.1145/2047196.2047212

[52] Katharina Reinecke and Krzysztof Z. Gajos. 2014. Quantifying Visual Preferences around the World. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, Toronto Ontario Canada, 11–20. doi:10.1145/2556288.2557052

[53] Katharina Reinecke, Tom Yeh, Luke Miratrix, Rahmatri Mardiko, Yuechen Zhao, Jenny Liu, and Krzysztof Z. Gajos. 2013. Predicting Users' First Impressions of Website Aesthetics with a Quantification of Perceived Visual Complexity and Colorfulness. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '13)*. Association for Computing Machinery, New York, NY, USA, 2049–2058. doi:10.1145/2470654.2481281

[54] Miguel A. Renom, Baptiste Caramiaux, and Michel Beaudouin-Lafon. 2022. Exploring Technical Reasoning in Digital Tool Use. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems (CHI '22)*. Association for Computing Machinery, New York, NY, USA, 1–17. doi:10.1145/3491102.3501877

[55] Miguel A. Renom, Baptiste Caramiaux, and Michel Beaudouin-Lafon. 2023. Interaction Knowledge: Understanding the 'Mechanics' of Digital Tools. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*. ACM, Hamburg Germany, 1–14. doi:10.1145/3544548.3581246

[56] Yossi Rubner, Carlo Tomasi, and Leonidas J. Guibas. 2000. The Earth Mover's Distance as a Metric for Image Retrieval. *International Journal of Computer Vision* 40, 2 (Nov. 2000), 99–121. doi:10.1023/A:1026543900054

[57] John W. Satzinger and Lorne Olfman. 1998. User Interface Consistency across End-User Applications: The Effects on Mental Models. *Journal of Management Information Systems* 14, 4 (March 1998), 167–193. doi:10.1080/07421222.1998.11518190

[58] Joey Scarr, Andy Cockburn, Carl Gutwin, Andrea Bunt, and Jared E. Cechanowicz. 2014. The Usability of CommandMaps in Realistic Tasks. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '14)*. Association for Computing Machinery, New York, NY, USA, 2241–2250. doi:10.1145/2556288.2556976

[59] Joey Scarr, Andy Cockburn, Carl Gutwin, and Sylvain Malacria. 2013. Testing the Robustness and Performance of Spatially Consistent Interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '13)*. Association for Computing Machinery, New York, NY, USA, 3139–3148. doi:10.1145/2470654.2466430

[60] G. SureshReddy, T. V. Rajinikanth, and A. Ananda Rao. 2014. Design and Analysis of Novel Similarity Measure for Clustering and Classification of High Dimensional Text Documents. In *Proceedings of the 15th International Conference on Computer Systems and Technologies*. ACM, Ruse Bulgaria, 194–201. doi:10.1145/2659532.2659615

[61] Sami Uddin and Carl Gutwin. 2021. The Image of the Interface: How People Use Landmarks to Develop Spatial Memory of Commands in Graphical Interfaces. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. ACM, Yokohama Japan, 1–17. doi:10.1145/3411764.3445050

[62] Jason Wu, Xiaoyi Zhang, Jeff Nichols, and Jeffrey P Bigham. 2021. Screen Parsing: Towards Reverse Engineering of UI Models from Screenshots. In *The 34th Annual ACM Symposium on User Interface Software and Technology*. ACM, Virtual Event USA, 470–483. doi:10.1145/3472749.3474763

[63] Xin Xia, David Lo, Feng Zhu, Xinyu Wang, and Bo Zhou. 2013. Software Internationalization and Localization: An Industrial Experience. In *2013 18th International Conference on Engineering of Complex Computer Systems*. 222–231.

doi:10.1109/ICECCS.2013.40

[64] Rahulkrishna Yandrapally, Andrea Stocco, and Ali Mesbah. 2020. Near-Duplicate Detection in Web App Model Inference. In *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ACM, Seoul South Korea, 186–197. doi:10.1145/3377811.3380416

[65] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. 2018. The Unreasonable Effectiveness of Deep Features as a Perceptual Metric. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. IEEE, Salt Lake City, UT, 586–595. doi:10.1109/CVPR.2018.00068

[66] Xiaoyi Zhang, Lilian de Greef, Amanda Swearngin, Samuel White, Kyle Murray, Lisa Yu, Qi Shan, Jeffrey Nichols, Jason Wu, Chris Fleizach, Aaron Everitt, and Jeffrey P Bigham. 2021. Screen Recognition: Creating Accessibility Metadata for Mobile Applications from Pixels. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. ACM, Yokohama Japan, 1–15. doi:10.1145/3411764.3445186

[67] Xiaoyi Zhang, Anne Spencer Ross, and James Fogarty. 2018. Robust Annotation of Mobile Application Interfaces in Methods for Accessibility Repair and Enhancement. In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology*. ACM, Berlin Germany, 609–621. doi:10.1145/3242587.3242616

[68] Xianjun Sam Zheng, Ishani Chakraborty, James Jeng-Weei Lin, and Robert Rauschenberger. 2009. Correlating Low-Level Image Statistics with Users - Rapid Aesthetic and Affective Judgments of Web Pages. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, Boston MA USA, 1–10. doi:10.1145/1518701.1518703

Received 20 February 2007; revised 12 March 2009; accepted 5 June 2009