X-Ray: Screenshot Accessibility via Embedded Metadata

Sujeath Pareddy, Anhong Guo, Jeffrey P. Bigham
Human-Computer Interaction Institute, Carnegie Mellon University, Pittsburgh, PA, USA
sujeath@cmu.edu, {anhongg, jbigham}@cs.cmu.edu

ABSTRACT
Screenshots are frequently shared on social media, via personal communications, and in academic papers. Unfortunately, existing screenshot tools strip away semantics useful for making the content accessible, leaving only pixels. For example, a screenshot of a table removes the structural information useful for conveying it. We quantify the scale of the problem via a study of academic papers, showing that a large number of images included in academic papers are screenshots, and validate this via qualitative interviews with researchers about their figure generation process. We then introduce *X-Ray*, a system that captures and embeds the semantics of the underlying content into images. Using the X-Ray screenshot tool, semantic information is captured and stored in the Exif data of the resulting image, allowing it to "tag along" as the image is shared and reposted. We demonstrate that our approach retains accessibility for screen reader users via a study with five blind participants. More generally, our approach suggests a method for embedding accessibility metadata into otherwise inaccessible formats, enabling them to retain the more accessible representations that are present at capture time.

Author Keywords
Accessibility; alt text; screen readers; vision impairment; runtime modification; large-scale analysis; pdf; screenshot.

CCS Concepts
•Human-centered computing → Interactive systems and tools; Accessibility technologies;

INTRODUCTION
Screenshots are largely inaccessible to screen reader users. Currently, the only information available to screen reader users comes from alternative text (alt-text). While useful, this creates a disparity between sighted and visually impaired people. A screenshot is visually similar to the underlying interface and can stand-in as long as no interactivity is desired. Sighted users can inspect the contents of the image, read text and even determine the state of GUI controls. For screen reader users, it is little more than a black box.

Screenshots are convenient to take on modern operating systems. Windows, Mac OS, Android and iOS allow easy access to screenshot functionality. Recent iterations of Windows and iOS show a popup overlay for sharing and markup when a screenshot is taken. This ubiquity suggests that screenshots have significant utility to the end user. Indeed, research shows that a significant fraction of images on Twitter are screenshots [14, 18]. We contribute to this line of research by analyzing the incidence of screenshots in 2,272 papers taken from HCI conference proceedings and arXiv Computer Science (CS) papers published in the year of 2018. Overall, we found on average 2.70 tables and 2.49 plots per paper. Interestingly, ~15% of tables in our sample were images that contained tables.

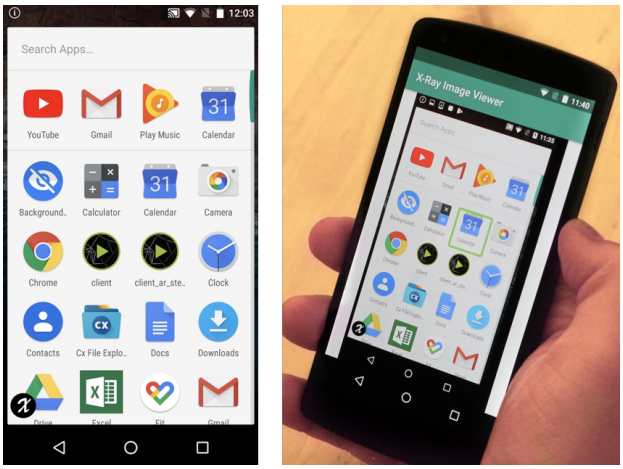

Figure 1. An "augmented" screenshot taken with the X-Ray capture tool (left) preserves the underlying semantics, allowing a screen reader user to interact with it as if they are using the underlying interface (right).

To better understand the figure generation process, we performed semi-structured interviews with 5 CS researchers in our institution. Researchers walked us through their process of making each figure in a recently published work. Our interviews validated our earlier findings about screenshots of tables. We found taking screenshots to be a significant part of the workflow. Many figures required moving between one or more GUI applications where they may be scaled, cropped, post-processed, composited or have other operations performed on them before placed in the paper. These steps strip away the semantics of the underlying content that is useful for accessibility. This motivated our idea to place metadata inside the image, so that the underlying semantics are preserved at every step along with the visual pixels.

In this paper, we introduce *X-Ray*, a system that captures and embeds the semantics of the underlying content into images. Using the X-Ray screenshot tool, semantic information is captured and stored in the Exif data of the resulting image, allowing it to "tag along" as the image is shared and reposted.

ASSETS '19, October 28-30, 2019, Pittsburgh, PA, USA

ACM ISBN 978-1-4503-6676-2/19/10 ...$15.00.
https://doi.org/10.1145/3308561.3353808

Exif is a standard metadata format that is normally used to store information such as camera model and geo-tags. Unlike photos taken with a physical camera, screenshots are images of a GUI which may have a machine readable representation. This is used by screen readers to navigate the interface, for automated UI testing and for web crawling. This representation can be captured when a screenshot is taken, and embedded inside the metadata fields of the image.

When the screen reader user encounters such an augmented image, the metadata can be extracted and the hierarchical structure can be mounted with its root in place of the image. This allows screen readers to navigate the virtual hierarchy. Users can use the same screen reader gestures and shortcuts they would normally use. Alternately, steganography could be used to embed information directly into the pixels themselves. Metadata could also be a separate file, similar to subtitle files for videos. Unlike reverse engineering approaches [8, 11, 12], our method has access to the ground truth of the underlying interface, allowing it to be robust. Alt-text that already exists in the interface is preserved, along with state information such as a particular element being selected.

We demonstrate that our approach retains accessibility for screen reader users via a user study with five blind participants, which is reasonable since the captured semantics are preserved and presented in the same way as in the original interface. More generally, our approach suggests a method for embedding accessibility metadata into otherwise inaccessible formats, enabling them to retain the more accessible representations that are present at capture time.

RELATED WORK

Our work is related to prior work on *(i)* alt-text, *(ii)* reverse engineering user interfaces, and *(iii)* analysis of accessibility.

Alternative Text

Many prior systems have attempted to obtain missing alt-text to increase their searchability and accessibility. Von Anh *et al.* [22] use an online game to provide alt-text for images. WebInsight [2] provides alt-text from a variety of sources, including Optical Character Recognition (OCR) and human labelling. While human provided alt-text can be very expressive and scalable, it can be expensive, potentially inaccurate and raise privacy concerns.

Salisbury *et al.* [21] explore the experiences of users with crowd generated captions. Their research explores how the alt-text generation process can be a conversation rather than a one off API call. This enables users to ask clarifying questions and request more detail. Our approach provides the raw interface itself and allows users to navigate and make sense of the interface without external support.

Wu *et al.* [23] automatically provide image captioning for images on the Facebook news feed. They report the results of a randomized control study with 9000 visually impaired users. Caption Crawler [16] provides alt-text by searching the web for instances of the same image that do have it. This highlights an important factor; alt-text may be available, but is not always copied along with the image. This motivates our approach of embedding the information inside the image itself.

AIMS [19] uses steganography to embed alt-text inside the pixels of images. Steganography can transmit information more robustly than Exif metadata, which may be stripped by services for privacy reasons. Our metadata can also use this method for transmission. However, embedding metadata in Exif does not degrade the quality of the image, and it can be efficiently and reliably extracted. It is also unclear if their method can survive image compression and post-processing, which is common on social media and in academic publications. Further, this metadata may be privacy sensitive, and the ability to easily strip it away is desirable.

Unlike alt-text, our system allows users to interactively navigate the hierarchy as if it were the original interface. This richer structured alternative to the image results naturally from our capture method, yet may more generally represent an alternative to alt-text useful for enabling access to complex content that is difficult to describe with text alone.

Reverse Engineering of Interfaces

There is a significant body of work on recovering GUIs from screenshots [8] and videos [17]. Dixon *et al.* [11, 12] reverse engineer the state and behavior of user interface widgets using only pixel level information. While promising, these approaches may be brittle and may require humans to refine the output. Furthermore, not all visually identical interfaces behave the same way when interpreted as a screen reader. Our approach preserves the original designer's intent. Elements are visited in the same order as in the original interface. Sometimes, developers create invisible elements that provide shortcuts and options, allowing screen reader users to use better strategies to navigate them [3], which our approach preserves. Of course, if the original interface was inaccessible, then X-Ray cannot help.

Interaction Proxies [24] is a strategy to improve mobile accessibility with accessible screen overlays. For example, an inaccessible image can be replaced with an accessible text overlay. Similarly, X-Ray replaces inaccessible screenshots with the original accessible virtual view hierarchy.

Engel *et al.* [13] explored a process and supporting tools to make static HTML versions of desktop GUIs. In their process, controls (e.g., buttons, checkboxes) are converted into HTML analogues. The primary goal of this work was to create screenshots for use in accessible tutorials. In contrast, our approach directly extracts the underlying UI structure when the screenshot is taken, and does not depend on the existence of an equivalent HTML control. As long as the control is accessible, its static behavior can be stored and reproduced.

Analysis of Accessibility

Rico [9] and Erica [10] are large scale repositories of Android UI screens, providing access to both screenshots and GUI metadata. Ross *et al.* [20] study the incidence of unlabelled buttons in Android apps. Brady *et al.* investigate the accessibility of HCI papers [4] and interview researchers on their experience of making papers accessible.

Bennet *et al.* [1] investigate social media practices among blind teens. They find that screenshots were used to overcome the ephemerality present on platforms such as Snapchat. Users

Selected	Count	Plot	Raw tbl	Img tbl	Bad Table	Photo	Illustration	GUI	Img text	Hybrid	Not sure
ASSETS	31	28.64%	29.61%	0.97%	3.17%	22.82%	9.71%	5.83%	0.0%	2.43%	0.00%
CHI	665	22.51%	21.11%	3.01%	12.49%	17.84%	19.24%	10.07%	1.01%	4.82%	0.40%
DIS	110	9.53%	10.94%	2.82%	20.51%	37.41%	24.71%	7.65%	0.82%	6.00%	0.12%
UIST	80	20.61%	8.52%	3.26%	27.68%	19.77%	24.40%	11.99%	0.53%	9.57%	1.37%
CSCW	184	24.19%	36.04%	3.08%	7.88%	6.82%	12.82%	14.04%	0.49%	2.19%	0.32%
IMWUT	202	42.76%	22.20%	1.78%	7.43%	7.48%	18.01%	2.49%	0.06%	4.93%	0.30%
arXiv	1,000	27.66%	30.88%	6.79%	18.03%	4.97%	23.96%	1.14%	0.45%	3.75%	0.40%
Summary	2,272	27.60%	25.47%	4.43%	14.81%	10.88%	21.06%	5.16%	0.54%	0.41%	4.46%

Table 1. Analysis of academic proceedings; the percentage of tables that are images are reported in column 'Bad Table.' Overall, 14.81% of tables in our sample were screenshots of tables, signifying the seriousness of the problem.

Category	Examples
Plot	Bar chart, histogram, etc.
Raw Tbl	Table whose text can be selected
Img Tbl	Screenshot of a table (text cannot be selected)
Photo	Photo of real world object
Illustration	Flow charts, system diagrams, clip art, etc.
GUI	Screenshot of software interfaces
Img text	Screenshot of text
Hybrid	More than one categories in a single image
Not sure	Others

Table 2. Coding manual for our academic proceeding analysis to understand the incidence of screenshots in academic papers.

would snapshot an image that is about to disappear and then view it later with assistive technology such as magnification. Gleason *et al.* [14] estimate that 9.7% of human uploaded images on Twitter are screenshots. Further, they report that users wished for descriptions longer than the 420 character limit imposed by Twitter at that time. Morris *et al.* [18] also analyse tweets and report interesting uses of screenshots, such as embedding text in images to circumvent the character limit.

ACADEMIC PROCEEDINGS ANALYSIS

To better understand the incidence of screenshots in academic papers, we analyzed 2,272 papers taken from HCI conference proceedings and ArXiv CS papers published in 2018 (Table 1). We manually downloaded proceedings from ACM Digital Library and from ArXiv using through their official bulk download mechanism. We randomly selected 1,000 papers that had Computer Science as their primary category. Using the coding manual in Table 2, an in-house annotator labelled the figures and tables in all the PDFs. Results are shown in Table 1. Specifically, the fraction of tables that are actually images is displayed in Bad Table column.

Overall, we found that there were 2.70 tables and 2.49 plots per paper on average. Interestingly, 14.81% of tables in our sample were actually screenshots of tables, signifying the seriousness of the problem. We found much variation in the fraction of such "bad tables" across conferences (only 3.17% in ASSETS, and 27.68% in UIST being the worst). Furthermore, even though bad tables are the most obvious problems, plots, illustrations, GUI screenshots, images containing just text, and hybrid figures – representing 63.24% of all figures and tables – are all constructed using mobile and desktop GUI applications. The semantics and structural information of the underlying content are lost, which is useful for accessibility.

INTERVIEW STUDY WITH RESEARCHERS

To better understand the figure and table design process beyond our quantitative analysis, we performed semi-structured interviews with 5 researchers (2F, 3M) from our institution. Researchers were from a diverse range of fields including Interaction Techniques, Fabrication, Natural Language Processing, User Experience and Machine Learning. We began by asking them to give an overview of their research and where they generally publish. We then asked them to describe their paper writing process with a focus on figures and tables from their recent work. We went over each figure / table and discussed their workflow. The interviews were audio recorded and results were analyzed using thematic analysis [15]. Two major themes emerged which we summarize below.

Screenshots are simply convenient

When asked which tool was used to generate a plot in his paper, R1 said, "*from Excel, we actually took a screenshot, from Excel there is a way to export it as a figure, but I just took a screenshot. We take screenshots a lot because it is easier but it depends on whether it is a high res screenshot or not.*" Some tools offered alternatives preferable to screenshots. When probed if a particular image from their paper was a screenshot, R1 said, "*No, that's from Keynote, we just exported as image, it's higher quality.*" When asked why they could not do the same in Excel, R1 said, "*Because it gave me weird spacings, you can't save the entire workbook as an image, just one chart at a time or the entire workbook.*" R5 were not aware that Keynote had this functionality. R2 reported, *"this image was done in a screenshot with Keynote and you can see it's a little blurred."* R4 highlighted how screenshots are used for editing images. R4 needed to increase the font size in a figure obtained from someone else. She said, "*that's taken directly from the documentation. I used a really really bad way ... created a cover over the original text and wrote the same text in a bigger font on the top (and then took a screenshot).*"

Researchers pick tools based on several factors

Researchers reported taking great pains to ensure that their tables were aesthetically pleasing, and making one figure often involved several tools. R1 reported taking screenshots of visualization tools built in Processing, arranging them in a grid in Keynote and then exporting to Photoshop for finishing touches before inserting into Word.

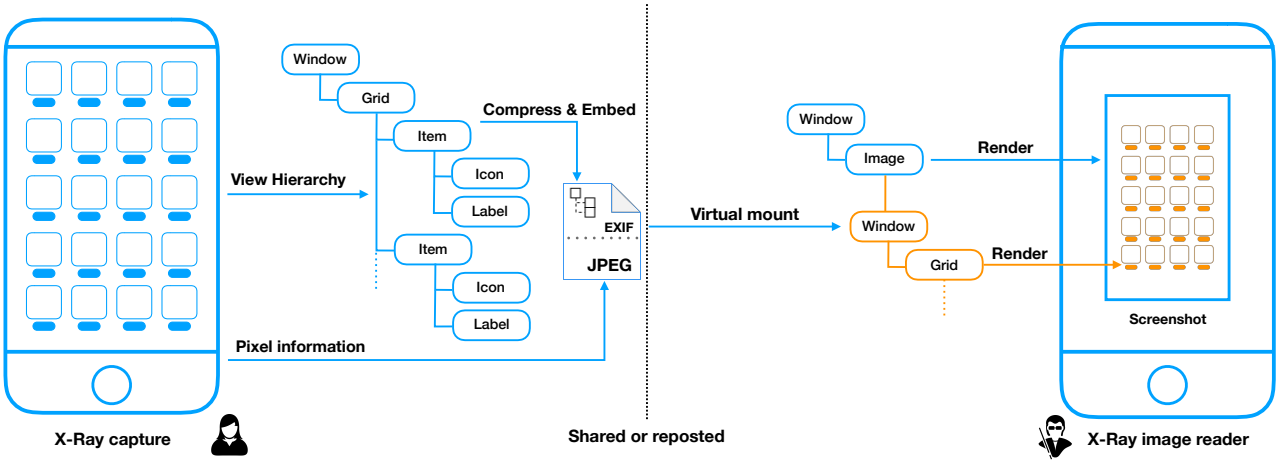

Figure 2. X-Ray is a system that embeds metadata into the image itself at capture time, allowing it to "tag along" as the image is shared. Using the X-Ray screenshot tool, semantic information is captured and stored in the Exif data of the screenshot. The X-Ray Image Viewer then allows screen reader users to access the screenshots as if they are using the underlying interface.

Researchers reported making trade offs between the quality of figures based on the time left before their deadlines. R1 said, "*I originally made the plot in matplotlib, we had to rerun some data at the last minute, and I just ended up making this one in Excel.*" R5 reported, "*the ideal way to do it (export) would be PDF, but if I am in a time crunch then I'll just take a screenshot of it.*" R2 reported how his desire for better quality figures sometimes conflicted with his collaborators' choice of tools. He said, "*If it was just me I would use (Adobe) Illustrator, but when I delegate tasks to someone, I cannot control what tools they use. I ended up using draw.io for this figure.*"

TECHNICAL IMPLEMENTATION

Our study highlights the importance of convenience to the researcher and the complexity of their workflows. We aim to provide a seamless experience identical to existing screenshot tools. To allow semantic information to be automatically propagated along with pixels, we choose to embed it inside the image itself in standard metadata fields (Exif). If an editing tool strips away this information, then it can also be transmitted externally as a separate file. Although our proof-of-concept implementation runs on Android, other platforms such as Windows and MacOS have equivalent APIs and can support similar implementations. The code is available here.[1]

Screenshot Capture

On Android, the UI hierarchy is available using the Accessibility Service API as a forest of nodes. This data structure allows screen readers such as TalkBack to access the GUI. Each node represents an interface element or a container used to organize child elements. We start at the root window of the active app and recursively proceed downwards. At each step, we store the node's text, description, bounds in the screen, class, state information and other attributes important for accessibility. This data is encoded as a JSON file. An example JSON can be accessed in the provided code repository.

Embedding Metadata Inside Images

The obtained JSON is compressed and embedded inside the Exif (exchangeable image file format) section of the screenshot JPEG. Exif is normally used to store image metadata such as the camera manufacturer and GPS coordinates. We use the User Comments field to store our metadata.

Virtual Rendering

Android Accessibility API allows custom views that can simulate virtual children.[2] This is used to facilitate accessibility of complex views such as Calendars, which are generally drawn onto a Bitmap instead of using the Android view system.

When the screenshot is opened, the metadata is extracted from the Exif and the view hierarchy contained within is virtually mounted below the image. This tricks the screen reader into thinking the original interface is actually present. The advantage of this approach is that Braille displays and interaction techniques such as explore by touch would still work, even for complex customized components. Furthermore, the alt-text field of the ImageView is still present, allowing the author to provide extra context to the screenshot.

USER EVALUATION

We performed a qualitative user study with 5 blind participants (Table 3). Our goal was to understand the ease of use of our prototype and its perceived fidelity to the source interface. We took screenshots of mock apps including GUI (Figure 1), images, text, and tables (Figure 3). Participants were asked to browse the screenshots and answer questions about them, e.g., is the "Stay awake" switch on or off (Figure 3c). Then, we showed them the raw interface and asked them about differences they could perceive and general questions about the user experience. They were asked to rate X-Ray along multiple scales in Table 4.

Participants were able to use both the augmented screenshots and the original interfaces. P4 found minor differences in the Android Home Screen use case. X-Ray was not aware of more advanced dynamic behavior of interfaces, such as announcing the available actions for an app (e.g., uninstall, move). P3 noticed that controls in the Settings screen were not grouped in the exact same way. This happened because our capturing

[1] https://gitlab.com/sujeathpareddy/xray

[2] https://developer.android.com/reference/android/view/View.AccessibilityDelegate.html

ID	Gender	Age	Occupation	Vision Level	Hearing	Smartphone Use	Desktop Use
P1	Female	64	Retired	Light perception, since 10 years old	Normal	iPhone, 9 years	JAWS, 24 years
P2	Female	71	Retired	Blind, since childhood	Slight loss	iPhone, 7.5 years	JAWS, 20 years
P3	Male	66	Retired	Blind, since birth	Slight loss	iPhone, 10 years	JAWS, NVDA 24 years
P4	Female	34	Unemployed	Blind, since birth	Normal	iPhone, 7 years	JAWS, 20 years
P5	Female	77	Retired	Light perception	Normal	iPhone, 2 years	JAWS, 24 years

Table 3. Demographics of five blind participants in the user evaluation of X-Ray.

ID	P1	P2	P3	P4	P5	Mean
Learnability	7	5	7	7	7	6.6
Comfort	7	4	7	6	7	6.2
Usefulness	7	5	7	5	6	6
Perceived Speed	7	3	7	4	7	5.6
Perceived Accuracy	6	4	6	7	7	6
Satisfaction	7	3	7	7	7	6.2

Table 4. Participants' ratings on X-Ray. Ratings were along a Likert scale of 1 to 7, 1 being extremely negative and 7 being extremely positive.

tool was not aware of the containing view group, which we will address in the future. Nevertheless, P3 commented, "*I could still tell that it (the Stay Awake switch) was a switch and that it was turned off.*" P1 said, "*I would definitely use it, because it's very frustrating you cannot access information that you know is there ... I would like it built right into the operating system, instead having to pull out a separate app.*"

DISCUSSION AND FUTURE WORK

Design Requirements for Screen Reader Users

In our implementation, we focused on making the experience of accessing screenshots as close to that of the original interface as possible and did not investigate if this is the most desirable thing to do. Design requirements for screen reader users is important for future work.

Technical Limitations

Our current implementation of X-Ray uses a dedicated reader to extract the embedded metadata and attach it to the Android View hierarchy. A more seamless integration would allow both desktop and mobile screen readers to access the embedded metadata without opening another app. This would either require modification to the source code of the screen reader itself or use a plugin system such as those in NVDA or JAWS.

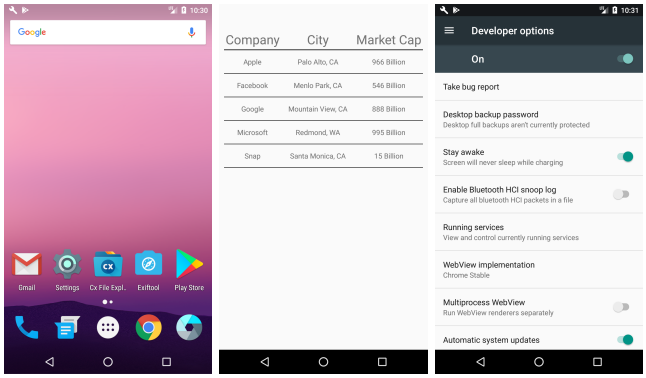

Figure 3. User study tasks for using X-Ray to access screenshots embedded with additional metadata, including: GUI, images, text, and tables.

X-Ray currently cannot handle operations such as zooming and rotation that may be performed by image editing tools. While these tools may remove nodes that are out of the frame, this may be addressed by storing region specific features in the node hierarchy. When loaded, the screen reader can then detect which nodes are now out of range and present the remaining.

Other limitations stem from the Android Accessibility API. For example, our crawler cannot access background content of the screen when there is modal content in the foreground (e.g., a dialog box). Our method is also unaware of dynamic behavior that may be added by developers. For example, an announcement can be made in response to an element becoming focused.

Accessibility of Graphical Plots

X-Ray currently does not support graphical charts and plots. However, it is possible to modify plotting tools to embed metadata into plots that can be accessed in a way similar to GUIs. This metadata could then be rendered using sonification techniques [6, 7], 3D printing [5], or other methods.

Privacy and Security

Embedding graphical metadata into screenshots can raise privacy concerns. Imagine a user takes a screenshot using X-Ray and then crops out personal information. If the cropping tool just copies the Exif data, private information may leak out. While implementing this technology as a standard can help solve the problem, backwards compatibility may remain an issue and pose a barrier for adoption.

CONCLUSION

We introduced X-Ray, a system that captures and embeds the underlying content semantics inside the Exif data of screenshots. Screen reader users were able to access the underlying GUI when viewing the image. Since this data is automatically stored when taking a screenshot, sighted users do not have to make special efforts to make their images accessible. Since most image processing tools support Exif, no special effort is needed to ensure that it is propagated along with the image.

ACKNOWLEDGMENTS

This work has been supported by the National Science Foundation (#IIS-1816012), Google, and the National Institute on Disability, Independent Living, and Rehabilitation Research (NIDILRR). We thank the participants who contributed to our studies, and the reviewers for their valuable feedback and suggestions. Special thanks to Patrick Carrington, Sven Mayer and Amy Pavel for their help and support.

REFERENCES

1. Cynthia L. Bennett, Jane E, Martez E. Mott, Edward Cutrell, and Meredith Ringel Morris. 2018. How Teens with Visual Impairments Take, Edit, and Share Photos on Social Media. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. ACM, New York, NY, USA, Article 76, 12 pages. DOI:http://dx.doi.org/10.1145/3173574.3173650
2. Jeffrey P. Bigham, Ryan S. Kaminsky, Richard E. Ladner, Oscar M. Danielsson, and Gordon L. Hempton. 2006. WebInSight:: Making Web Images Accessible. In *Proceedings of the 8th International ACM SIGACCESS Conference on Computers and Accessibility (Assets '06)*. ACM, New York, NY, USA, 181–188. DOI: http://dx.doi.org/10.1145/1168987.1169018
3. Yevgen Borodin, Jeffrey P. Bigham, Glenn Dausch, and I. V. Ramakrishnan. 2010. More Than Meets the Eye: A Survey of Screen-reader Browsing Strategies. In *Proceedings of the 2010 International Cross Disciplinary Conference on Web Accessibility (W4A) (W4A '10)*. ACM, New York, NY, USA, Article 13, 10 pages. DOI: http://dx.doi.org/10.1145/1805986.1806005
4. Erin Brady, Yu Zhong, and Jeffrey P. Bigham. 2015. Creating Accessible PDFs for Conference Proceedings. In *Proceedings of the 12th Web for All Conference (W4A '15)*. ACM, New York, NY, USA, Article 34, 4 pages. DOI:http://dx.doi.org/10.1145/2745555.2746665
5. Craig Brown and Amy Hurst. 2012. VizTouch: Automatically Generated Tactile Visualizations of Coordinate Spaces. In *Proceedings of the Sixth International Conference on Tangible, Embedded and Embodied Interaction (TEI '12)*. ACM, New York, NY, USA, 131–138. DOI: http://dx.doi.org/10.1145/2148131.2148160
6. L.M. Brown, S.A. Brewster, S.A. Ramloll, R. Burton, and B. Riedel. 2003a. Design guidelines for audio presentation of graphs and tables. (2003). http://eprints.gla.ac.uk/3196/
7. Lorna M Brown, Stephen A Brewster, SA Ramloll, R Burton, and Beate Riedel. 2003b. Design guidelines for audio presentation of graphs and tables. International Conference on Auditory Display.
8. Chunyang Chen, Ting Su, Guozhu Meng, Zhenchang Xing, and Yang Liu. 2018. From UI Design Image to GUI Skeleton: A Neural Machine Translator to Bootstrap Mobile GUI Implementation. In *Proceedings of the 40th International Conference on Software Engineering (ICSE '18)*. ACM, New York, NY, USA, 665–676. DOI: http://dx.doi.org/10.1145/3180155.3180240
9. Biplab Deka, Zifeng Huang, Chad Franzen, Joshua Hibschman, Daniel Afergan, Yang Li, Jeffrey Nichols, and Ranjitha Kumar. 2017. Rico: A Mobile App Dataset for Building Data-Driven Design Applications. In *Proceedings of the 30th Annual Symposium on User Interface Software and Technology (UIST '17)*.
10. Biplab Deka, Zifeng Huang, and Ranjitha Kumar. 2016. ERICA: Interaction Mining Mobile Apps. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology (UIST '16)*. ACM, New York, NY, USA, 767–776. DOI: http://dx.doi.org/10.1145/2984511.2984581
11. Morgan Dixon and James Fogarty. 2010. Prefab: Implementing Advanced Behaviors Using Pixel-based Reverse Engineering of Interface Structure. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '10)*. ACM, New York, NY, USA, 1525–1534. DOI: http://dx.doi.org/10.1145/1753326.1753554
12. Morgan Dixon, Gierad Laput, and James Fogarty. 2014. Pixel-based Methods for Widget State and Style in a Runtime Implementation of Sliding Widgets. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '14)*. ACM, New York, NY, USA, 2231–2240. DOI: http://dx.doi.org/10.1145/2556288.2556979
13. Christin Engel, Denise Bornschein, and Gerhard Weber. 2018. Accessible Screenshots for Blind and Visually Impaired People. In *Mensch und Computer 2018 - Tagungsband*, Raimund Dachselt and Gerhard Weber (Eds.). Gesellschaft für Informatik e.V., Bonn.
14. Cole Gleason, Patrick Carrington, Cameron Cassidy, Meredith Ringel Morris, Kris M. Kitani, and Jeffrey P. Bigham. 2019. "It's Almost Like They're Trying to Hide It": How User-Provided Image Descriptions Have Failed to Make Twitter Accessible. In *The World Wide Web Conference (WWW '19)*. ACM, New York, NY, USA, 549–559. DOI: http://dx.doi.org/10.1145/3308558.3313605
15. Greg Guest, Kathleen M MacQueen, and Emily E Namey. 2011. *Applied thematic analysis*. Sage Publications.
16. Darren Guinness, Edward Cutrell, and Meredith Ringel Morris. 2018. Caption Crawler: Enabling Reusable Alternative Text Descriptions Using Reverse Image Search. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. ACM, New York, NY, USA, Article 518, 11 pages. DOI: http://dx.doi.org/10.1145/3173574.3174092
17. Anhong Guo, Junhan Kong, Michael Rivera, Frank F. Xu, and Jeffrey P. Bigham. 2019. StateLens: A Reverse Engineering Solution for Making Existing Dynamic Touchscreens Accessible. In *Proceedings of the 32th Annual Symposium on User Interface Software and Technology (UIST '19)*. ACM, New York, NY, USA. DOI: http://dx.doi.org/10.1145/3332165.3347873
18. Meredith Ringel Morris, Annuska Zolyomi, Catherine Yao, Sina Bahram, Jeffrey P. Bigham, and Shaun K. Kane. 2016. "With Most of It Being Pictures Now, I Rarely Use It": Understanding Twitter's Evolving Accessibility to Blind Users. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 5506–5516. DOI: http://dx.doi.org/10.1145/2858036.2858116

19. Ab Shaqoor Nengroo and K. S. Kuppusamy. 2018. Accessible images (AIMS): a model to build self-describing images for assisting screen reader users. *Universal Access in the Information Society* 17, 3 (01 Aug 2018), 607–619. DOI: http://dx.doi.org/10.1007/s10209-017-0607-z

20. Anne Spencer Ross, Xiaoyi Zhang, James Fogarty, and Jacob O. Wobbrock. 2018. Examining Image-Based Button Labeling for Accessibility in Android Apps Through Large-Scale Analysis. In *Proceedings of the 20th International ACM SIGACCESS Conference on Computers and Accessibility (ASSETS '18)*. ACM, New York, NY, USA, 119–130. DOI: http://dx.doi.org/10.1145/3234695.3236364

21. Elliot Salisbury, Ece Kamar, and Meredith Ringel Morris. 2017. Toward scalable social alt text: Conversational crowdsourcing as a tool for refining vision-to-language technology for the blind. In *Fifth AAAI Conference on Human Computation and Crowdsourcing*.

22. Luis von Ahn, Shiry Ginosar, Mihir Kedia, Ruoran Liu, and Manuel Blum. 2006. Improving Accessibility of the Web with a Computer Game. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '06)*. ACM, New York, NY, USA, 79–82. DOI:http://dx.doi.org/10.1145/1124772.1124785

23. Shaomei Wu, Jeffrey Wieland, Omid Farivar, and Julie Schiller. 2017. Automatic Alt-text: Computer-generated Image Descriptions for Blind Users on a Social Network Service. In *Proceedings of the 2017 ACM Conference on Computer Supported Cooperative Work and Social Computing (CSCW '17)*. ACM, New York, NY, USA, 1180–1192. DOI: http://dx.doi.org/10.1145/2998181.2998364

24. Xiaoyi Zhang, Anne Spencer Ross, Anat Caspi, James Fogarty, and Jacob O. Wobbrock. 2017. Interaction Proxies for Runtime Repair and Enhancement of Mobile Application Accessibility. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. ACM, New York, NY, USA, 6024–6037. DOI: http://dx.doi.org/10.1145/3025453.3025846