CrossCode: Multi-level Visualization of Program Execution

Background

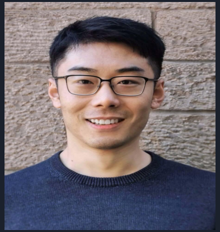
The paper was authored by these three individuals:

Devamardeep Hayatpur, Daniel Wigdor, and Haijun Xia.

The Authors

Devamardeep Hayatpur is a Ph.D. student at the University of California, San Diego (UCSD), working in the domain of human-computer interaction (HCI).

Daniel Wigdor is a professor at the University of Toronto, with notable contributions to the field of HCI, such as the development of natural user interfaces and advanced interaction techniques.



Haijun Xia is an Assistant Professor at the University of California, San Diego. His research in Human-Computer Interaction focuses on enhancing productivity and creativity by developing innovative representations and interaction techniques.

Publication

This research paper was presented at the 2023 CHI Conference on Human Factors in Computing Systems (CHI ’23), which is one of the most prestigious conferences in the field of HCI. The paper was published in April 2023 and represents a significant contribution to program visualization techniques, specifically targeting the challenge of navigating between different levels of abstraction in program execution.

Abstract

This paper introduces CrossCode, a web-based tool that helps programmers reduce cognitive challenges by visualizing program execution across different abstraction levels. CrossCode simplifies navigation by using aggregation, abbreviation repetitive steps and provides high level overview of program execution, improving JavaScript exploration. A study with expert programmers showed it enhances context management, state tracking, and debugging efficiency, making it a promising improvement for both programming education and professional workflows.

Methodology

Research Methods:

The research involved both formative studies and qualitative evaluations to design and validate CrossCode. The research focused on understanding how programmers reason about code at multiple levels of abstraction and how this process could be better supported with visual tools.

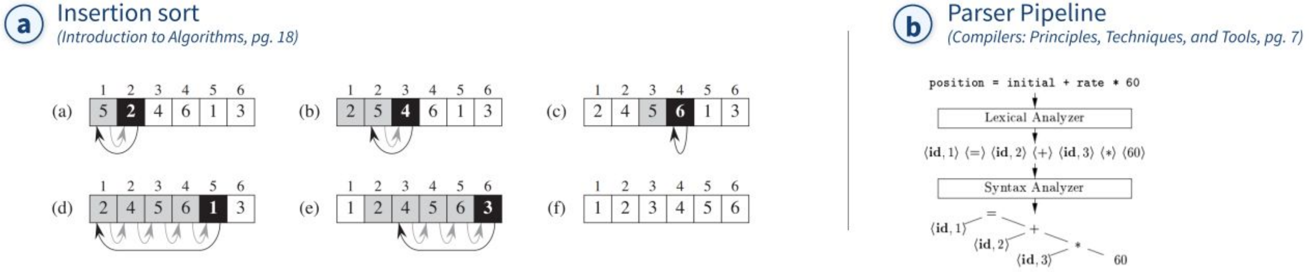
Formative Study

The researchers analyzed 92 program visualizations, diagrams, and animations from instructional materials such as textbooks and online tutorials to derive common design patterns used to communicate program behavior. This analysis led to the identification of three key design patterns (aggregation, abbreviation, and execution overview) that were used to influence the design of CrossCode.

Aggregation

Aggregation is the combining of multiple, low-level operations into larger, more meaningful units. Instead of viewing every single line of code, related operations are grouped into key steps.

D1. Aggregate *operations into key steps.*



a Insertion sort

(Introduction to Algorithms, pg. 18)

b Parser Pipeline

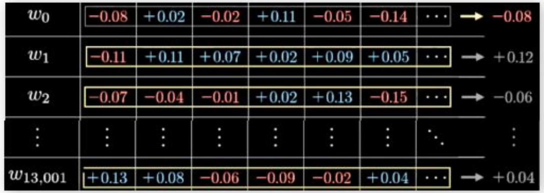
(Compilers: Principles, Techniques, and Tools, pg. 7)

Abbreviation

Abbreviation reduces the visual complexity of repetitive operations by summarizing them. In cases where loops or recursive functions are used, only the first few iterations and the final step are displayed, while the intermediate steps are abbreviated.

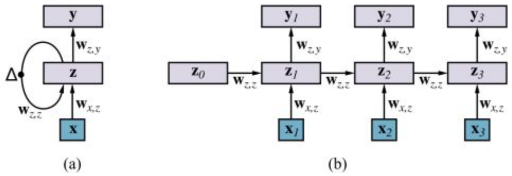
D2. Abbreviate
repetitive operations.

c Average of each row in a matrix
(3blue1brown, https://youtu.be/Ilg3gGewQ5U?t=515)



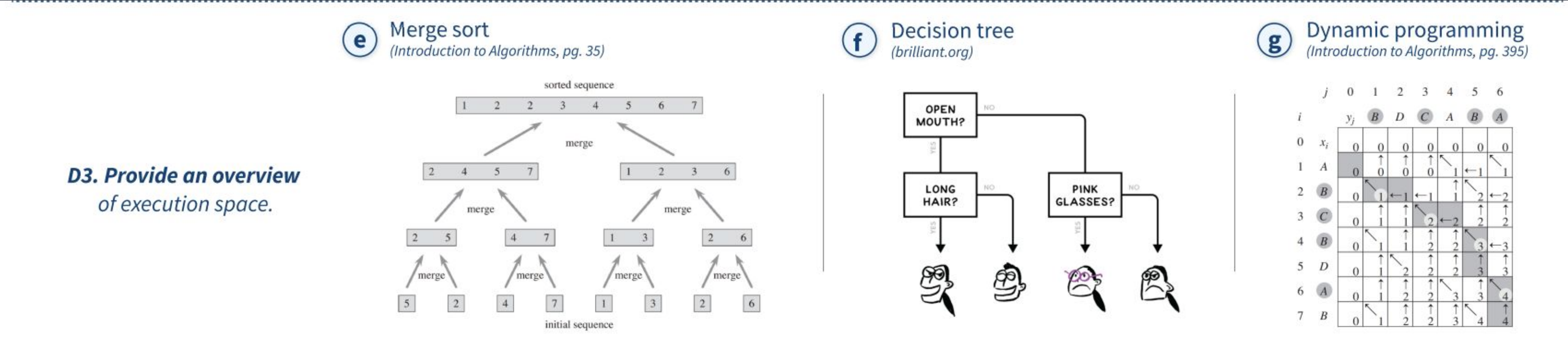
w_0	−0.08	+0.02	−0.02	+0.11	−0.05	−0.14	⋯	→	−0.08
w_1	−0.11	+0.11	+0.07	+0.02	+0.09	+0.05	⋯	→	+0.12
w_2	−0.07	−0.04	−0.01	+0.02	+0.13	−0.15	⋯	→	−0.06
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋱		⋮
$w_{13,001}$	+0.13	+0.08	−0.06	−0.09	−0.02	+0.04	⋯	→	+0.04

d Recurrent Neural Net
(Artificial Intelligence: A Modern Approach, https://ima.cs.berkeley.edu/figures.pdf#page=186)



Execution Overview

A high-level overview provides a broad visualization of the entire program's execution, allowing the user to see the overall control flow and structure.

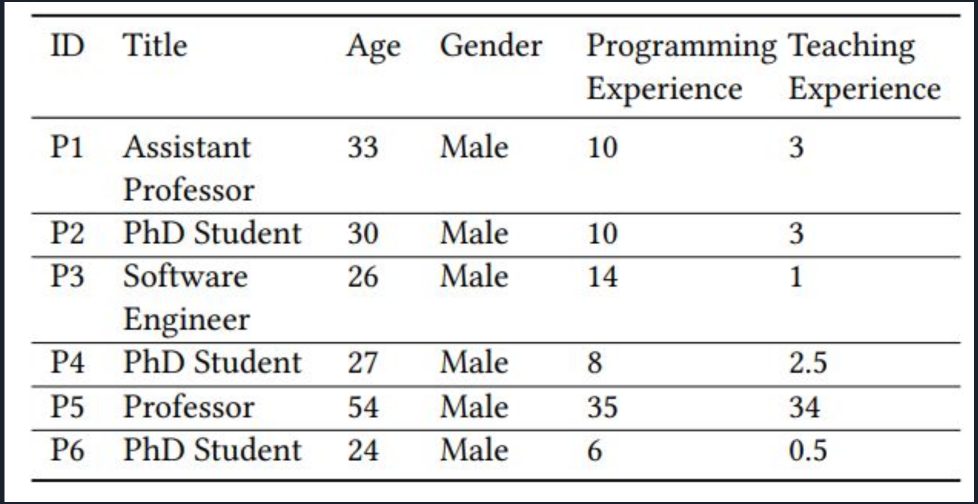


Experimental Design

A custom interpreter for JavaScript was developed in TypeScript to implement CrossCode, which allowed visualization of program execution at different levels of detail, using control flow and data flow to help users navigate between abstractions.

Usability Study

An exploratory qualitative study was conducted with six expert programmers. The participants were asked to debug programs using three different conditions: CrossCode, Python Tutor (a line-by-line program visualization tool), and manual drawing. They were then interviewed to reflect on the usefulness of each tool in terms of program understanding and debugging efficiency.



ID	Title	Age	Gender	Programming Experience	Teaching Experience
P1	Assistant Professor	33	Male	10	3
P2	PhD Student	30	Male	10	3
P3	Software Engineer	26	Male	14	1
P4	PhD Student	27	Male	8	2.5
P5	Professor	54	Male	35	34
P6	PhD Student	24	Male	6	0.5

Data Collection Techniques:

The researchers collected qualitative data through think-aloud protocols during the debugging tasks and semi-structured interviews. Participants shared insights on how each tool affected their ability to understand, debug, and explain code.

Results

The study found that CrossCode provided significant advantages over existing line-by-line debugging tools, particularly in navigating complex control flows and repetitive structures (e.g., loops and recursive functions).

Program Understanding: CrossCode helped participants maintain a clear sense of place during execution, especially in complex code, by providing intuitive overviews and aggregating operations into meaningful steps.

Debugging: Aggregations and abbreviations streamlined the debugging process, enabling focus on key steps while skipping repetitive operations. Color-coded animations and traces made tracking variable changes easier.

Efficiency: Compared to Python Tutor, CrossCode was more efficient for tasks involving loops or recursion. The ability to zoom between abstraction levels saved time compared to manually stepping through code.

Discussion

The research contributes to the field of HCI by addressing the cognitive load programmers face when navigating between different levels of program execution. CrossCode’s approach of allowing users to visualize multiple levels of abstraction aligns with cognitive models of how programmers reason about code, providing a more flexible and intuitive interface for debugging.

Limitations

The Learning Curve in using the program

Data Synchronization between Control Flow View and Data View

Limited to only JavaScript

Conclusion

CrossCode introduces a new approach to program visualization, allowing programmers to switch between abstraction levels, reducing cognitive load and improving debugging efficiency. By aggregating and simplifying operations, CrossCode aligns with how programmers naturally debug and understand code, making it useful for both debugging and teaching. This research is relevant to HCI, particularly in programming environments and educational tech. CrossCode’s flexible, multi-level view could inspire innovations that enhance understanding for novices and experts alike, with potential to scale to complex programs and other paradigms.

References

Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. 2009. Introduction to Algorithms, Third Edition (3rd ed.). The MIT Press, Cambridge, MA, USA.

Alfred V. Aho, Monica S. Lam, Ravi Sethi, and Jeffrey D. Ullman. 2006. Compilers: Principles, Techniques, and Tools (2nd Edition). Addison-Wesley Longman Publishing Co., Inc., USA.

Brilliant. 2011. Brilliant. Retrieved September 15, 2022 from https://brilliant.org/

Stuart Russell and Peter Norvig. 2009. Artificial Intelligence: A Modern Approach (3rd ed.). Prentice Hall Press, USA.

Grant Sanderson. 2017. 3Blue1Brown. Retrieved September 15, 2022 from https://www.youtube.com/c/3blue1brown

Linguistic Dead-Ends and Alphabet Soup: Finding Dark Patterns in Japanese Apps

CHI’ 2023

Background

Shun Hidaka: Department of Industrial Engineering and Economics Tokyo Institute of Technology, Japan

Sota Kobuki: Department of Industrial Engineering and Economics Tokyo Institute of Technology, Japan

Mizuki Watanabe: Department of Industrial Engineering and Economics Tokyo Institute of Technology, Japan

Katie Seaborn: Department of Industrial Engineering and Economics Tokyo Institute of Technology, Japan

Abstract

**Dark Patterns: type of malicious interface design pattern that tricks or forces the end-user into taking an action that is different from what they intend or expect*

- ## Main Objective

Investigate the prevalence of dark patterns in Japanese apps and compare the similarities and differences to those in Western apps.

- ## Contributions

A. Finding a new category of dark patterns
B. Finding evidence which aligns with previous taxonomy, further reinforcing and extending the previous model by Colin Gray

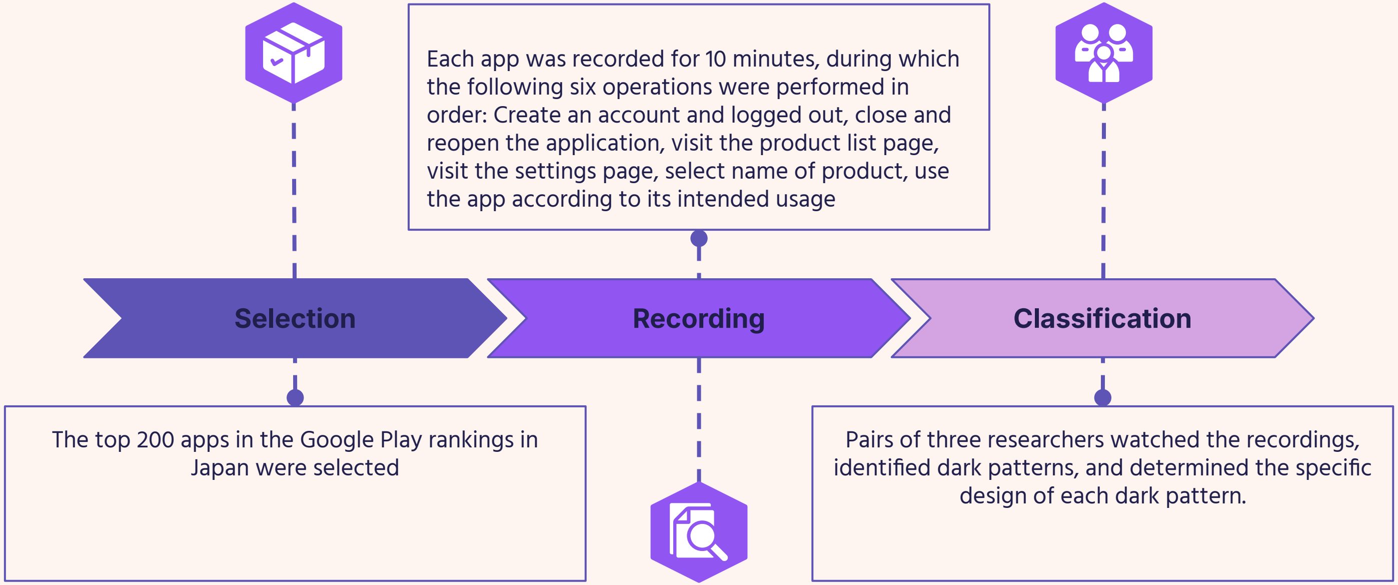
- ## Findings

significantly less representation of dark patterns in the Japanese context, specifically an average of 3.9 compared to 7.4.

Methodology

Observation

Qualitative Data Collection



Selection

The top 200 apps in the Google Play rankings in Japan were selected

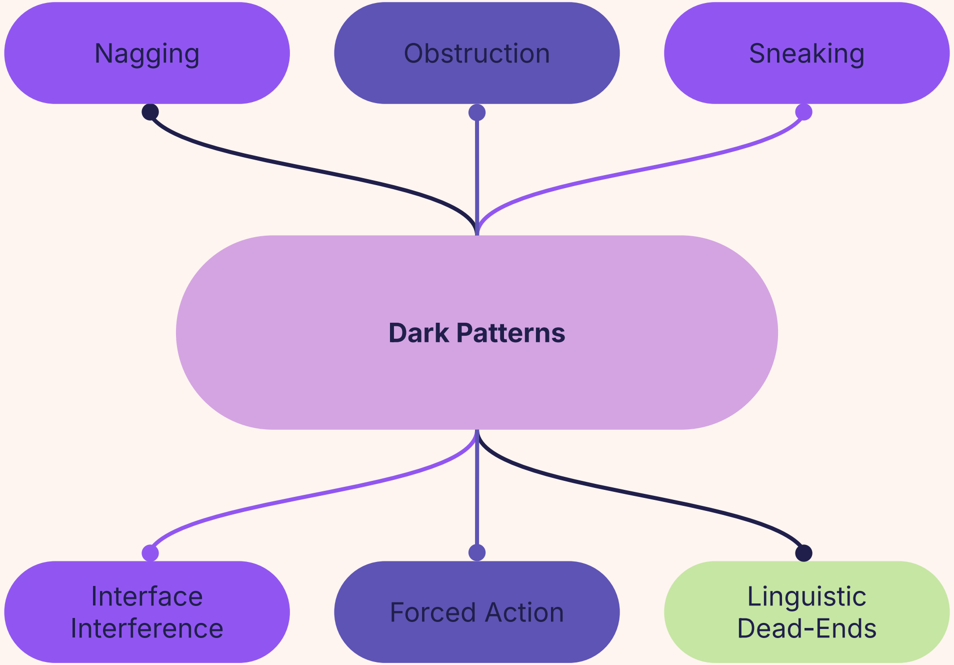
Recording

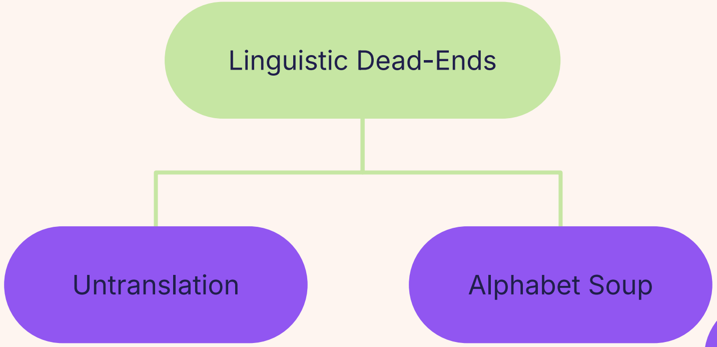
Each app was recorded for 10 minutes, during which the following six operations were performed in order: Create an account and logged out, close and reopen the application, visit the product list page, visit the settings page, select name of product, use the app according to its intended usage

Classification

Pairs of three researchers watched the recordings, identified dark patterns, and determined the specific design of each dark pattern.

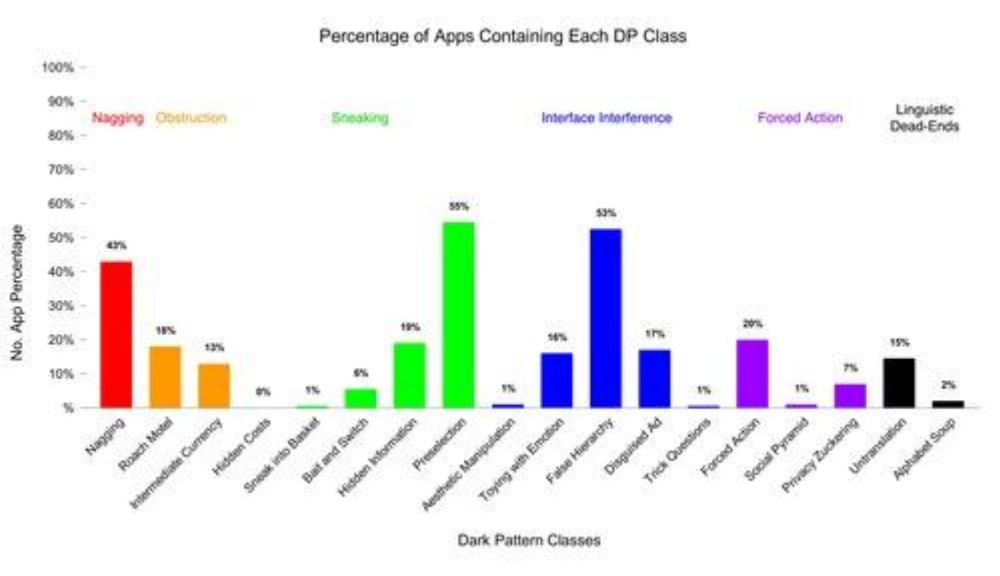
Classes of Dark Patterns





Results

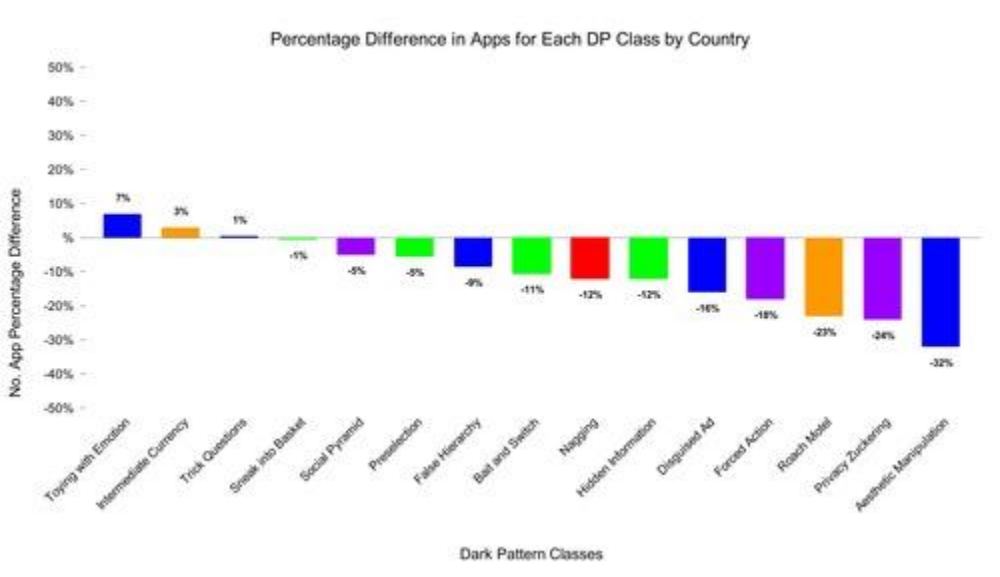
Frequency of each category of dark patterns



ℹ ANOVA and Shapiro-Wilk test showed no statistical difference of rate of appearance of each dark pattern

Results

Apps were further separated by their app category and their relative appearance were compared to previous work done by Di Geronimo



Discussion

- ## Implications

The results suggest that there are fewer dark patterns compared to Western apps, which may be due to dark patterns being relatively novel in Japan, but they could increase in the future.

Paper also show that more apps from different cultures need to be researched as different dark patterns can be found

- ## Limitations

This is not a controlled comparison to Di Geronimo's work, as Geronimo’s paper used Android apps downloaded from the Google Play Store, while this study used iOS apps from the App Store. Since the App Store and Google Play have different policies, some design patterns will vary to meet Apple's requirements.

Conclusion

This paper reveals that Western apps appear to have a higher frequency of dark patterns compared to Japanese apps. It also emphasizes the need for cross-cultural research on dark patterns, suggesting that other hidden categories may exist within different cultural and linguistic contexts. Additionally, the paper highlights the bias in current research, which is predominantly focused on Western countries, calling for more diverse studies as Western populations do not represent the global population.

References

- Di Geronimo, L., Braz, L., Fregnan, E., Palomba, F., & Bacchelli, A. (2020). UI Dark Patterns and Where to Find Them. *CHI' 20*. https://doi.org/10.1145/3313831.3376600
- Bhoot, A. M., Shinde, M. A., & Mishra, W. P. (2020). Towards the Identification of Dark Patterns: An Analysis Based on End-User Reactions. *IndiaHCL '20.* https://doi.org/10.1145/3429290.3429293

Exploring the Impact of Interconnected External Interfaces

in Autonomous Vehicles on Pedestrian Safety and Experience

CHI 2024

BACKGROUND

The authors of this paper are prominent researchers in the field of Human-Computer Interaction (HCI).

Tram Thi Minh Tran, **Callum Parker**, **Marius Hoggenmüller**, and **Yiyuan Wang** are affiliated with the **University of Sydney's Design Lab**,

Martin Tomitsch is part of the **Transdisciplinary School at the University of Technology Sydney**.

ABSTRACT

Main Objectives

The primary objective of the paper is to examine the influence of **interconnected external Human-Machine Interfaces (eHMIs) on pedestrian safety and user experience**.

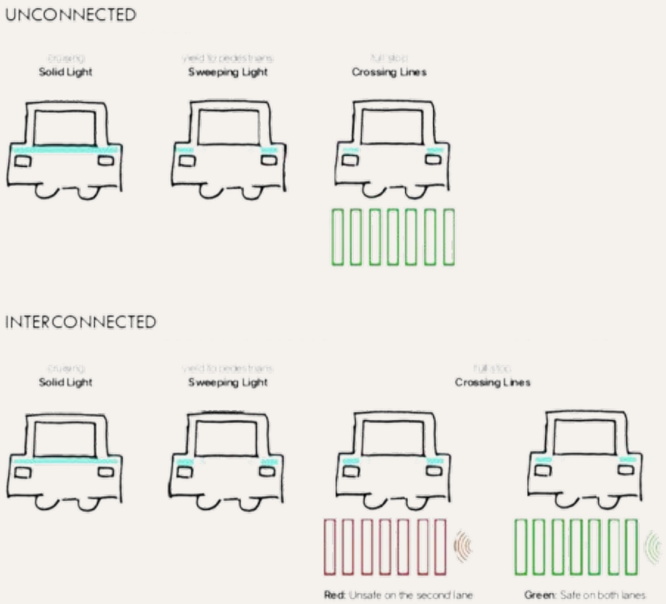
Contributions

The study provides insight into the effectiveness of eHMIs in enhancing safety in **complex traffic conditions** as well as offer design recommendations.

Findings

The study's findings reveal that **interconnected eHMIs** improved pedestrian safety by making interactions with AVs clearer and more predictable.

METHODOLOGY

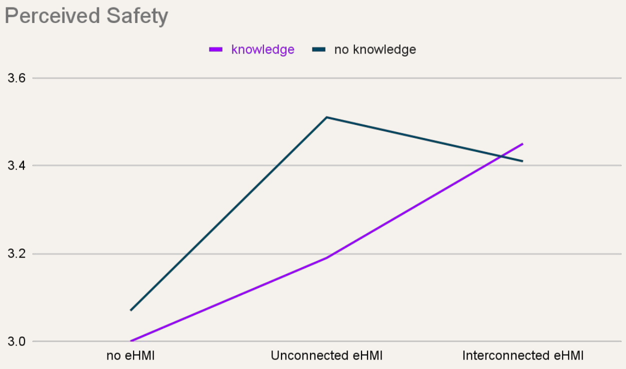


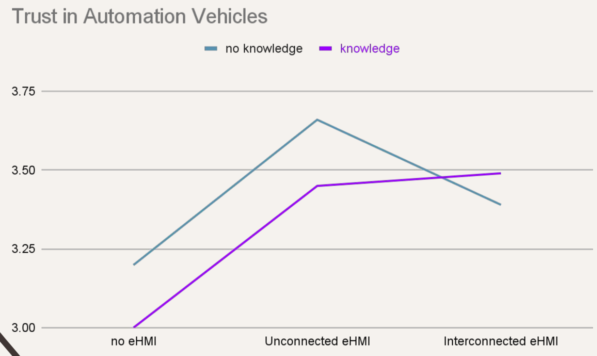
The research employed a **VR-based** experimental study involving 32 participants.

Participants were exposed to three conditions: **no** eHMI, **unconnected** eHMI, and **interconnected** eHMI, to assess pedestrian safety, and trust in AV systems.

Surveys and interviews were also conducted to gather **qualitative and quantitative data** on perceived safety and trust.

RESULTS





The study revealed that **interconnected eHMIs** led to **improved** pedestrian safety and cautious crossing behaviors.

However, **design issues**, such as the use of red for crosswalk signals, caused some **confusion** and discomfort among participants.

The research showed that interconnected eHMIs **reduced cognitive load** in some scenarios but did not always increase trust due to the complexity of the system.

Moreover, prior knowledge of eHMI concepts i**nfluenced** user experience positively.

DISCUSSION

IMPLICATIONS

The findings highlight that interconnected eHMIs **improve** pedestrian safety and interaction with AVs. They lead to **fewer collisions** and higher perceived safety, making interactions easier to understand and more reliable. This is crucial for user trust and safety in HCI, guiding better design and policy decisions for **autonomous driving** and pedestrian safety.

LIMITATIONS

Additionally, limitations include the use of **VR simulations**, which may not fully capture **real-world** pedestrian behaviors, and a relatively small sample size.

CONCLUSION

This research demonstrates the **importance** of designing effective eHMIs to improve pedestrian safety when interacting with AVs. Interconnected eHMIs show promise in reducing confusion in **multi-lane traffic environments**, but further refinements are needed.

REFERENCES

Jonas Andersson, Azra Habibovic, Maria Klingegård, Cristofer Englund, and Victor Malmsten-Lundgren. 2017. Hello human, can you read my mind? ERCIM News 109 (2017), 36–37.

[2] Driving Test Australia. 2017. What you need to know about red painted areas on some Australian roads? https://emudrivingschool.com.au/Blog/What-youneed-to-know-about-red-painted-areas-on-some-Australian-roads Accessed: 6 Feb 2024.

[3] Pavlo Bazilinskyy, Dimitra Dodou, and Joost de Winter. 2020. External HumanMachine Interfaces: Which of 729 Colors Is Best for Signaling 'Please (Do not) Cross'?. In 2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC). IEEE, Toronto, ON, Canada, 3721–3728. https://doi.org/10.1109/SMC42975. 2020.9282998