A cumulative dissertation submitted in fulfillment
for the academic grade Doktor-Ingenieur (Dr.-Ing.)
with the Bauhaus-Universität Weimar

# Interactive Surface Environments: Design and Implementation

Everett M. Mthunzi
Faculty of Media
Bauhaus-Universität Weimar
everett.mthunzi@zeiss.com

Internal Reviewer
Prof. Dr. rer. nat. Florian Echtler
Department of Computer Science
Aalborg University
floech@cs.aau.dk

External Reviewer
Prof. Dr. Ing. Dietrich Kammer
Technische Visualistik
Hochschule für Technik und Wirtschaft Dresden
dietrich.kammer@htw-dresden.de

Date of Defence: 30 Jun. 2023

## Declaration

This dissertation cumulates research conducted by myself in candidature for the grade Doktor-Ingenieur with the Bauhaus-Universität Weimar. The research comprises original ideas that have not been submitted for other qualifications. Contributions from collaborators have been acknowledged, and ideas from other scholars have been duly referenced, both literally and in context. This dissertation, in its entirety, has been authored and edited by myself.

July 2, 2023

Everett M. Mthunzi
(Faculty of Media, Bauhaus-Universität Weimar)

Date

## Acknowledgements

I want to thank Florian for his constant support throughout my doctoral candidature. I also want to thank Chris and Sujay, for much–appreciated support and comradery, in and outside the DBL. To John, thank you for roping me into discussions that encouraged me to always aim upward. To my late mother, Ann, your love, kindness, patience, strength, perseverance, and wisdom are with me always.

## Funding

This research was made possible through the support of the Bundesministerium für Bildung und Forschung (grant number 16SV8288). I am also grateful to the Bauhaus-Universität Weimar and VIGITIA (Vernetzte Intelligente Gegenstände durch, auf und um interaktiveTische im Alltag) for the research opportunity.



What we have once enjoyed, we can never lose.
All that we love deeply becomes a part of us.
—Helen Keller

## Dedication

In loving memory of Ann,
1965 — 2020

# Abstract

This dissertation presents three studies on the design and implementation of interactive surface environments. It puts forward approaches to engineering interactive surface prototypes using prevailing methodologies and technologies. The scholarly findings from each study have been condensed into academic manuscripts, which are conferred herewith.

The first study identifies a communication gap between engineers of interactive surface systems (i.e., originators of concepts) and future developers. To bridge the gap, it explores a UML-based framework to establish a formal syntax for modeling hardware, middleware, and software of interactive surface prototypes. The proposed framework targets models-as-end-products, towards enabling a shared view of research prototypes thereby facilitating dialogue between concept originators and future developers.

The second study positions itself to support developers with an open-source solution for exploiting 3D point clouds for interactive tabletop applications using CPU architectures. Given dense 3D point-cloud representations of tabletop environments, the study aims toward mitigating high computational effort by segmenting candidate interaction regions as a preprocessing step. The study contributes a robust open-source solution for reducing computational costs when leveraging 3D point clouds for interactive tabletop applications. The solution itself is flexible and adaptable to variable interactive surface applications.

The third study contributes an archetypal concept for integrating mobile devices as active components in augmented tabletop surfaces. With emphasis on transparent development trails, the study demonstrates the utility of the open-source tool developed in the second study. In addition to leveraging 3D point clouds for real-time interaction, the research considers recent advances in computer vision and wireless communication to realize a modern, interactive tabletop application. A robust strategy that combines spatial augmented reality, point-cloud-based depth perception, CNN-based object detection, and Bluetooth communication is put forward. In addition to seamless communication between adhoc mobile devices and interactive tabletop systems, the archetypal concept demonstrates the benefits of preprocessing point clouds by segmenting candidate interaction regions, as suggested in the second study.

Collectively, the studies presented in this dissertation contribute; 1—bridging the gap between originators of interactive surface concepts and future developers, 2—promoting the exploration of 3D point clouds for interactive surface applications using CPU-based architectures, and 3—leveraging 3D point clouds together with emerging CNN-based object detection, and Bluetooth communication technologies to advance existing surface interaction concepts.

# Zusammenfassung

In dieser Dissertation werden drei Studien zur Gestaltung und Umsetzung interaktiver Oberflächenumgebungen vorgestellt. Sie liefert Methoden für die Entwicklung von Prototypen mit modernen Ansätzen. Die wissenschaftlichen Ergebnisse der einzelnen Studien wurden in akademischen Manuskripten zusammengefasst, die hiermit überreicht werden.

In der ersten Studie wird eine Kommunikationslücke zwischen den Ingenieuren interaktiver Oberflächensysteme (d. h. den Urhebern der Konzepte) und den künftigen Entwicklern festgestellt. Um diese Lücke zu schließen, wird ein UML-basierter Rahmen erforscht und eine formale Syntax für die Modellierung von Hardware, Middleware und Software interaktiver Oberflächenprototypen in Richtung Modelle als Endprodukte entwickelt. Der beigetragene modellgetriebene Ansatz fördert konsistente Beschreibungsprototypen, die direkt eine gemeinsame Sichtweise für Forschungsartefakte ermöglichen und damit den Dialog zwischen Ingenieuren und zukünftigen Entwicklern erleichtern.

Die zweite Studie will Entwickler mit einer Open-Source-Lösung für die Nutzung von 3D-Punktwolken für interaktive Tabletop-Anwendungen mit CPU-Architekturen unterstützen. Bei dichten 3D-Punktwolken-Darstellungen von Tabletop-Umgebungen zielt die Studie darauf ab, den hohen Rechenaufwand durch die Segmentierung von möglichen Interaktionsregionen als Vorverarbeitungsschritt zu verringern. Die Studie liefert eine robuste Open-Source-Lösung zur Reduzierung der Rechenkosten bei der Nutzung von 3D-Punktwolken für interaktive Tabletop-Anwendungen. Die Lösung selbst ist flexibel und kann an unterschiedliche anwendungsspezifische Szenarien angepasst werden. In der dritten Studie wird ein Konzept für die Integration von mobilen Geräten als aktive Komponenten in erweiterte Tischoberflächen vorgestellt. Die Studie legt Wert auf transparente Entwicklungspfade und zeigt den Nutzen des in der zweiten Studie entwickelten Open-Source-Tools auf. Neben der Nutzung von 3D-Punktwolken für die Interaktion in Echtzeit berücksichtigt die Studie die jüngsten Fortschritte in der Computer Vision und der drahtlosen Kommunikation, um eine praktische, interaktive Tabletop-Anwendung zu realisieren. Es wird eine Strategie vorgeschlagen, die räumliche Augmented Reality, punktwolkenbasierte Tiefenwahrnehmung, CNN-basierte Objekterkennung und Bluetooth-Kommunikation kombiniert. Neben einer robusten Strategie für die nahtlose Kommunikation zwischen mobilen Ad-hoc-Geräten und interaktiven Tabletop-Systemen demonstriert der Prototyp die Vorteile der Vorverarbeitung von Punktwolken durch Segmentierung von möglichen Interaktionsbereichen, wie in der zweiten Studie vorgeschlagen. Insgesamt tragen die in dieser Dissertation vorgestellten Studien dazu bei 1—den Graben zwischen den Urhebern interaktiver Oberflächenkonzepte und zukünftigen Entwicklern zu überbrücken, 2–die Erforschung von 3D-Punktwolken für interaktive Oberflächenanwendungen mit CPU-basierten Architekturen zu fördern, und 3—die Nutzung von

3D-Punktwolken zusammen mit neuen Technologien, um bestehende Interaktionskonzepte zu verbessern.

# List of publications

## SELECTED PUBLICATIONS


[1] Everett Mondliwethu Mthunzi, Christopher Getschmann, and Florian Echtler. Fast 3d point-cloud segmentation for interactive surfaces. In *Interactive Surfaces and Spaces*, ISS '21 Companion, New York, NY, USA, 2021. Association for Computing Machinery. DOI: https://doi.org/10.1145/3447932.3491141.

[2] Everett Mondliwethu Mthunzi and Florian Echtler. Artefact: A uml-based framework for model-driven development of interactive surface prototypes. In *Interactive Surfaces and Spaces*, ISS '21 Companion, New York, NY, USA, 2021. Association for Computing Machinery. DOI: https://doi.org/10.1145/3447932.3490523.

[3] Everett Mondliwethu Mthunzi, Mathias Mueller, Dietrich Kammer, and Florian Echtler. Model-driven development of interactive surface prototypes. 2022. DOI: 10.31219/osf.io/2ux4p.

[4] Everett Mondliwethu Mthunzi and Florian Echtler. Traceless: Integrating mobile devices into augmented tabletop environments. 2022. DOI: 10.31219/osf.io/x3dg6.

[5] Everett Mondliwethu Mthunzi and Florian Echtler. Segmenting candidate-interaction regions: Preprocessing 3d point clouds for interactive tabletop applications. 2022. DOI: 10.31219/osf.io/ubgr3.


## OTHER PUBLICATIONS


[1] Everett Mondliwethu Mthunzi. Performance analysis of a protection scheme based on p-class synchrophasor measurements. Master's thesis, Cape Peninsula University of Technology, 2016. http://etd.cput.ac.za/handle/20.500.11838/2378.

[2] Everett Mondliwethu Mthunzi and Raynitchka Tzoneva. Algorithm for mitigating adverse synchrophasor measurement latency in packet losses in wide-area power systems. In *25th Southern African Universities Power Engineering Conference*, SAUPEC 2017, Vereeniging, South Africa, 2017. SAIEE.

[3] Kay Smarsly, Everett Mondliwethu Mthunzi, Oliver. Hahn, and Josephine. Planer. Validation of an ultra-low-cost wireless structural health monitoring system for civil infrastructure. In *Proceedings of the 12th international workshop on structural health monitoring*, Stanford, CA, USA, 2019. DOI: 10.12783/shm2019/32445.

[4] Everett Mondliwethu Mthunzi and Hayder Alsaad. Monitoring strategies for personalized hvac: A work in progress report. In *Proceedings of the 31st Forum Bauinformatik*, Berlin, Germany, 2019. Universitätsverlag der TU Berlin. DOI: http://dx.doi.org/10.14279/depositonce-8763.

[5] Patricia Peralta Abadia, Everett Mondliwethu Mthunzi, Sebastian Heine, Horst-michael Ludwig, and Kay Smarsly. A metamodel for 3d concrete printing. In *Proceedings of the 16th international conference on civil, structural & environmental engineering computing*, Riva del Garda , TN, Italy, 2019. Elsevier.

[6] Patricia Peralta Abadia, Everett Mondliwethu Mthunzi, and Sebastian Heine. A semantic model for additive manufacturing of concrete structures. In *Proceedings of the 31st Forum Bauinformatik*, Berlin, Germany, 2019. Universitätsverlag der TU Berlin. DOI: http://dx.doi.org/10.14279/depositonce-8763.

[7] Christopher Getschmann, Everett Mondliwethu Mthunzi, and Florian Echtler. Mirrorforge: Rapid prototyping of complex mirrors for camera and projector systems. In *Sixteenth International Conference on Tangible, Embedded, and Embodied Interaction*, TEI '22, New York, NY, USA, 2022. Association for Computing Machinery. DOI: 10.1145/3490149.3501329.

# Table of contents

# List of figures

# List of algorithms



# List of tables

# List of equations

# Contributed open-source repositories

*"A problem well-stated is half-solved." — Charles Kettering*

# 1
# Research outline

## 1.1 INTRODUCTION

HCI (Human-computer interaction) continues to play a central role in advancing computing systems in everyday environments [7, 85]. While the discipline comprises multiple paradigms [192], the Computer Science perspective narrows in on the relationship between computers and users, focusing on the argument—the functionality of any computer cannot be realized without usability. In this context, functionality relates to the set of actions and services provided by a computer, and usability is a measure of how perceived functionality enables users to employ a computer for its intended purpose. The far-reaching impact of HCI research can be seen in ubiquitous applications such as the World Wide Web and computing systems such as the Macintosh [125, 169].

A niche area in HCI that has gained traction over the last two decades is the space of interactive surface environments.

The motivation behind the increase in academic interest is self-evident: be it at home or work, day-to-day activities center around walls and tabletops. The study of interactive surface environments explores the potential of everyday surfaces toward re-imagining how we perceive and interact with surfaces around us. In 1991 Wellner introduced the digital desk calculator, which elaborated on the concept of merging digital and non-digital interactions on tabletop surfaces [215]. To achieve this, Wellner integrated physical objects into a virtual environment using spatial augmented reality. [1] Among other early works, the digital desk calculator laid groundwork for augmenting day-to-day surfaces with tangible manipulable mixed reality. Since then, the space of interactive surface environments has grown immeasurably. Concepts such as the cave [39], the meta-desk [201], continuous workspaces [175], and many more have pervaded the research community.

[1] Spatial augmented reality merges the physical and digital worlds by superimposing computer-generated graphics onto real-world surfaces using projectors [56, 195, 221].

Modern interactive-surface concepts now consider a broad range of aspects that envelop ubiquitous computing [135, 214], frameworks [109], and tools [225]. The emergence of low-cost commodity sensors has also played a significant role in propelling the niche area. This observation is consistent with the rapid progression of technological considerations in prevailing research artifacts, viz, the rapid advancement of concepts, particularly over the last decade, e.g., from enhanced workspaces in 2001 [118], to world-kit in 2013 [228], and magic paper in 2019 [227].

The foremost goal in the space of interactive surface environments is to re-imagine how we perceive and interact with surfaces in our everyday environments. Tabletops [118], walls [194], floors [84], surface tangibles [207], and even limbs [226] all have untapped display potential. This potential can be realized, and arbitrary surfaces

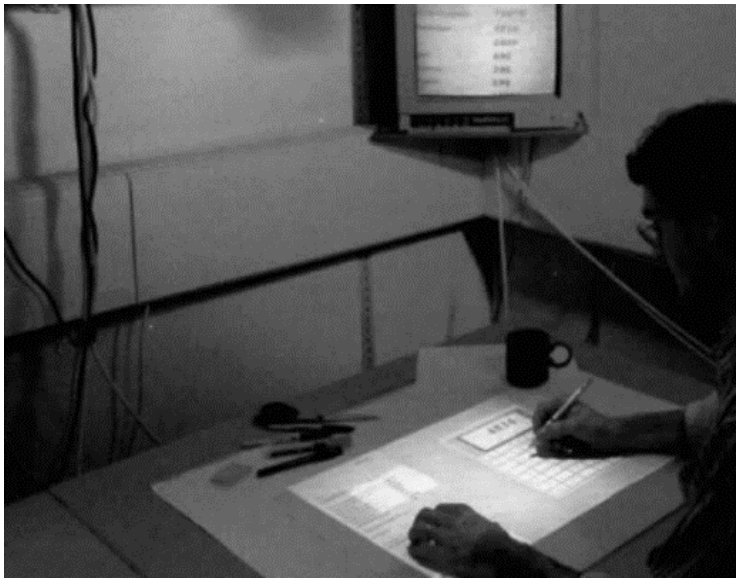


Fig. 1. The digital-desk calculator [215]. Interactive surface environments facilitate tangible manipulation on everyday surfaces, enabling physical interaction with digital objects. They also integrate physical objects as active components in interactive computing systems, enhancing the interaction context of physical objects.

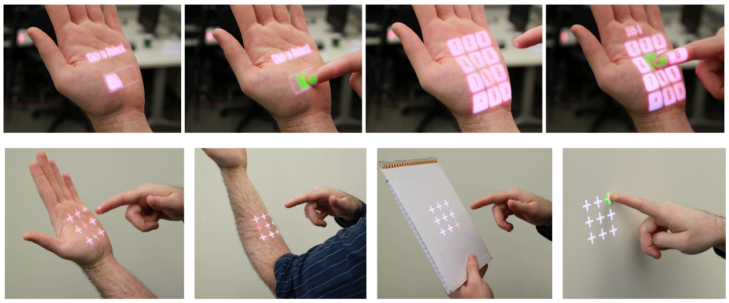


Fig. 2. Omni-touch [86]. Re-imagining how we perceive and interact with surfaces in our everyday environments.

in day-to-day environments can be enriched with interaction, e.g., ubiquitous collaborative workspaces [105], the physical-virtual workstation [229], augmented surface tangibles [33], information hubs [228], and assistive living [43].

## 1.2 MOTIVATION

A body of literature in the field of HCI extensively discusses the replication crisis. The replication crisis refers to the difficulty researchers face when reproducing the findings of previous studies. This challenge raises critical concerns about the credibility and reliability of scientific research. Moreover, the replication crisis hinders the progression and advancement of existing contributions. One factor contributing to the replication crisis is the absence of methodology to support replication. Keeping view of the replication crisis, the research presented in this dissertation aims to address technical challenges associated with the design and implementation of interactive surface environments. The following subsections detail the motivations behind three research studies focusing on methodological reform and promoting open source and open science to converge the studies towards contributing to the mitigation of the replication crisis.

### 1.2.1 Study 1: Methodological Reform

> With emphasis on transparent development trails, i.e., from design to implementation, what methods be realized to support engineering interactive surface environments?

Different approaches have been established for realizing interactive surface systems [37, 178]. While some propose integrating digital displays directly into surfaces [69, 70], others explore spatial augmented reality [106, 227]. Irrespective of the approach, there are common underlying design considerations. These considerations are self-evident in comparable research artifacts that have been put forward over the last two decades, e.g., the Meta-desk (1998) [202], Enhanced-desk (2001) [118], Play-anywhere [217], G-nome surface (2010) [189], Flexi-wall (2014) [152], Haptable (2019) [57], and Magic paper (2019) [227]. However, despite recurring design considerations, there is no support for concept re-use. This shortcoming has directly contributed to the accumulation of near-identical research prototypes with recurring limitations; a drawback that motivates the question: Can concepts from existing artifacts be abstracted into generic structures, architectural views, and technical descriptions to inform the design of future interactive surfaces?

### 1.2.2 Study 2: Promoting Open Source

> How can we promote open source to support developers with freely accessible building blocks for leveraging commodity depth cameras and spatial augmented reality for interactive surface applications?

In line with accumulating prototypes, increased accessibility to vision sensors [121, 142] has propelled adopting spatial augmented reality for interactive applications [50, 137]. A body of literature suggests that multiple software solutions have been realized. However, a critical review of research prototypes indicates a significant drawback—besides highly application-specific solutions, software contributions are typically not open-sourced. This drawback motivates the question: Given pervading depth cameras, can an open-source solution be realized to support researchers and developers exploit non-trivial point-cloud operations for spatial augmented reality?

### 1.2.3 Study 3: Exhibit Contributions Tackling the Replication Crisis

> How can we demonstrate the significance of transparent technical development trails using the contributions from the first two studies?

Narrowing in on application, one approach that has gained popularity over the last decade is extending ubiquity from mobile devices to interactive surfaces. Among others, prevailing studies in this direction include Pocket-transfers [136], Huddle-lamp [172], Bluetable [224], Flash-light [94], Tracko [104], and Headphones [77]. The central idea behind these concepts is how ubiquitous information can be leveraged to provide intuitive interaction with surface and surface tangibles in everyday environments. In addition to potential as interaction hubs, surfaces, particularly tabletops, are inanimate assistants that support day-to-day activities. An innate tendency is placing mobile devices on tabletop surfaces. Drawing parallels, mobile devices are in and of themselves assistants inseparable from day-to-day activities [77, 204]. The continuous physical interaction between mobile devices and tabletop surfaces motivates exploring latent interaction space between the two assistants. The motivating question that follows: Can both assistants be integrated as active components in an interactive surface's computing environment to realize the latent symbiotic interaction space between both assistants?

While studies address technical challenges associated with the design and implementation of interactive surface environments, the contributions converge towards tackling replication crisis.

### 1.2.4 Research Methodology

Each study follows a) structured research, b) peer review, and c) open science.

*A.* Structured research:
The general template for publication was adopted. Accordingly, each study

a. reviewed relevant literature,
b. critically studied existing methodology,
c. identified knowledge gaps,
d. developed contributions bridge identified knowledge gaps, and
e. validated proposed contributions.

*B.* Peer review and publication:
Findings from each study were synthesized into academic manuscripts. Then, each manuscript was entered into a publication track, subjecting scholarly findings to the scrutiny of field experts: a critical step in assessing the validity and quality of established contributions. Peer review was also used to safeguard against premature dissemination of insignificant findings as well as preemptively identify possible misinterpretations.

*C.* Open science:
While on track for publication, following peer-review, manuscripts were made freely accessible to researchers and developers using the open science framework [31].

## 1.3 DOCUMENT ORGANIZATION

The remainder of this dissertation is organized as follows: Chapters 2–4 present the original manuscripts. Each manuscript is prefaced with research context and concluded with critical remarks. Chapter 5 discusses the significance of the research contributions, and chapter 6 concludes with an outlook of potential future work.

*"Science is the systematic classification of experience." — George Henry Lewes*

# 2

# Model-driven development for interactive surface prototypes

## 2.1 CONTRIBUTORS


Co-author: Mathias Müller
email: mathias.mueller@htw-dresden.de
affiliation: HTW Dresden, Dresden, Germany
Contribution: Development of software meta model

Co-author: Dietrich Kammer
email: dietrich.kammer@htw-dresden.de
affiliation: HTW Dresden, Dresden, Germany
Contribution: Framework evaluation and extension

Co-author: Florian Echtler
email: floech@cs.aau.dk
affiliation: Aalborg University, Aalborg, Denmark
Contribution: Critical feedback and manuscript analysis


## 2.2 RESEARCH CONTEXT

Although the space of HCI continues to collect numerous research artifacts, replicating these archetypal implementations is complex and arguably impractical [51, 208]. This drawback is partly a consequence of the template for publication, which emphasizes academic manuscripts over diffuse of study assets. For research artifacts, details integral to replicating prototypes are commonly omitted and often left undisclosed [208]. As a result, assessing the validity of prototypes through replication is not possible, which diminishes the credibility of research artifacts to a great extent. In [208], Wacharamanotham et al. highlighted how sharing research assets is generally uncommon. Similarly,

Echtler and Haussler underlined how developers in HCI do not follow the open-source model, which can promote the replicability of scientific findings [51].

Wacharamanotham et al. identified a common underlying drawback faced by researchers: no strategies exist for documenting, communicating, and diffusing research assets. Given the first subproblem, the work presented in this chapter contributes towards mitigating the "replication crisis" [51, 208]. The study aims to bridge the communication gap between originators of concepts and future developers, thereby promoting the replicability of research artifacts. The concept of "models-as-end-products" [184] is explored together with a model-driven approach.

Existing studies show that model-driven approaches enhance communication between stakeholders at all levels [1, 116]. A model-driven methodology is thus a promising approach to bridging the communication gap amongst HCI developers. Moreover, model-driven approaches provide significant benefits in early development stages [4, 66, 188]. These benefits include (a) increased understanding of a problem space, (b) early identification limitations, (c) preemptive-risk evaluation, and (d) early assessment of possible design transformations. By adopting a model-driven approach, developers in HCI can leverage the outlined advantages. However, a setback that needs to be addressed first is the absence of shared logical-model representations [131].

The study presented in this chapter critically reviews research prototypes that have been contributed over the last two decades. [2] The review is used to establish a knowledge base and logical-model representations: a framework targeting unambiguous and consistent representation for prototypes. In line with the aim of the study, the objectives of the framework are as follows:

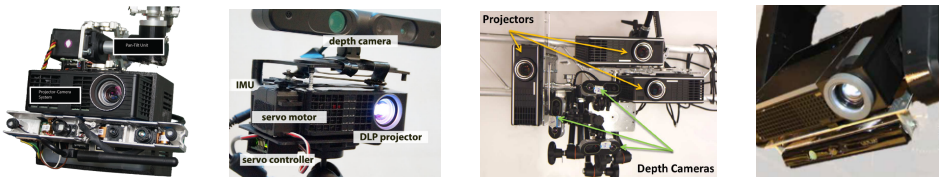
- Provide a strategy for documenting, communicating, and diffusing research assets.
- Bridge the communication gap between originators of concepts and future developers.
- And last, lend the benefits of model-driven development to the space of HCI.

# Manuscript 1

### 2.3 ABSTRACT

Interactive surface prototypes in the research community are typically application-specific. However, common features in existing artifacts suggest recurring design considerations. Given the rapid accumulation of near-identical prototypes, there is a need to promote design reuse. Existing research prototypes motivate abstracting generic components and architectural views to inform future designs. In this paper, we propose a UML-based framework for designing, documenting, and sharing interactive

[2] The research presented in this chapter extends [148]: a study put forward by myself and F.Echtler and published with the ACM International Conference on Interactive Surfaces and Spaces. The research extension remains on track for publication and can be freely accessed on the Open Science Framework [3].




(a) Ubeam [81]. (b) AMP-D [226]. (c) LightSpace [220]. (d) Beamatron [223].

Fig. 3. Interactive surface prototypes.

surface prototypes. The framework composes hardware, middleware, and software meta-models to support developers with technology-agnostic guidelines for expressing prototypes unambiguously. The meta-models are based on the study of existing prototypes and are evidence-based. For validation, we use the proposed framework to express interactive surface artifacts from prevailing studies. We then present the resulting model representations to experts to learn their perceptions towards the model-driven approach. As a proof of concept, and to show tacit benefits, we conducted a case study using ReFlex: a software framework for implementing elastic displays. Our findings highlight three significant benefits: (i) an accessible graphical syntax with unambiguous model representation, (ii) a system for capturing arbitrary technical specifications, and (iii) flexible model representation with consistent notation.

Additional Key Words and Phrases: Interactive surface environments, interactive surface prototypes, UML-based framework

## 2.4 INTRODUCTION

In the research space of interactive surface environments, prototyping enables the creative exploration of emerging hardware, middleware, and software. In principle, once a prototype is shared with the research community, its underlying concepts can be abstracted and reused as building blocks for future prototypes. However, in practice, replicating concepts in studies is non-trivial because research prototypes generally evaluate heuristics and usage while overlooking concept reuse. Few studies have addressed the challenge of concept reuse directly. Moreover, there is a body of literature that indicates the accumulation of near-identical prototypes with similar limitations [11, 81, 185, 186, 220, 223, 226], suggesting a connection to the "replication crisis" in HCI research [51, 208].

Our approach to mitigating this issue considers the varying perspectives of unique developers. Different actors, e.g., system architects, embedded-systems developers, network developers, and software developers, address domain-specific challenges. While system architects are concerned with meta aspects such as sub-system infrastructure and overarching system design, embedded-system developers focus on data acquisition and performance. Also, while network developers focus on adhering to communication

protocols and managing data transmission, software engineers center on data processing and data-driven functionality. Besides the focus on separate challenges, developers still need to work cohesively across the separate domains to realize a coherent interactive surface prototype. From a research perspective, developers must first review existing prototypes to underpin known limitations and knowledge gaps, irrespective of competency. While the goal may be straightforward, studying existing interactive surface prototypes is complex as there is no shared view, widespread representation, or unified syntax for documenting, sharing, and contrasting prototypes.

In this paper, we therefore propose Artefact: a UML-based framework for model-driven development of interactive surface prototypes. We first condense existing research prototypes into a knowledge base. Then, we leverage the knowledge base to formulate a flexible framework that promotes designing and expressing interactive surface prototypes unambiguously. With an emphasis on "models as end products" [184], we establish logical and consistent model representations for hardware, middleware, and software towards streamlining concept reuse. For validation, we employ the proposed framework to model prototypes from recent studies. We then subject the captured models to the scrutiny of experts and learn their perceptions towards the proposed model-driven approach. Finally, we present a proof of concept by applying the model-driven approach to our new ReFlex framework, developed to create applications for elastic displays.

## 2.5 RELATED WORK

Several studies have surveyed employing model-driven approaches for reusing artifact concepts [12, 40, 72, 98, 235]. Genero et al. [72] conducted a literature review on the role of models in development cycles and highlighted the significance of models in the iterative development of prototypes. Ho-Quang et al. [98] conducted a study across 458 projects to identify the motivation for growing interests in model-driven methods. The study suggested a causal relationship between the surge in open-source solutions and a need to communicate implementations unambiguously. Ho-Quang et al. also identified that developers utilize models to communicate structurally complex concepts to broader audiences, effectively bridging the gap between designers (i.e., originators of concepts) and future developers. In [40], Da Silva and Oliveira suggested stereotype-based modeling for concept reuse and elaborated by formulating generic artifact abstractions based on existing prototypes. The study highlighted the benefits of models as end products, underlining iterative advancement for existing artifacts. In [197], Torre et al. pointed to the widespread adoption of UML-based models and conducted a case study to investigate developer perceptions toward UML. Besides learning perceptions similar to those identified by Ho-Quang et al. in [98], the study also highlighted open and fast-growing UML-model repositories [67, 92].

Yoshino and Matsuura [235] discussed utility in employing models to reverse engineer prototypes. In a similar research context, Bowen contributed lightweight tools that enable reverse-engineering models from existing artifacts [25]. Akin to [98, 197],

Yoshino and Matsuura also emphasized the role of UML models for bridging the communication gap between designers and developers. Dittmar et al. discussed how high-abstraction models can contribute to end-user programming by enabling users to manipulate the models [47]. The authors suggested that UML-based model-driven development, in specific cases, could promote formal verification [235].

Turner et al. focused on using models for generating test cases that cover the intersection between frontend and backend components, targeting structured test coverage [200]. Similarly, Bowen and Reeves also investigated how models-specific components could be grouped to create consistent test cases across separate domains [26]. Hinze et al. presented an emulator to enable testing of user-facing components prior to implementation [97]. In the same research direction, Besnard et al. [16] discussed abstraction of physical environments to promote verification processes. Here, the authors pointed out two major challenges: First, capturing associations of components across separate domains. And second, capturing arbitrary technical specifications in abstract modules. To address these challenges, Besnard et al. suggested adopting UML mechanisms to generalize system description and proposed a technology-agnostic model-driven approach.

In the context of user interface development, several works [3, 27, 58, 71] have discussed using formal models to provide a structure for "user interface patterns", with focus on auto-generating user interface variants for multiple platforms. Luyten et al. discussed the replication crisis with respect to interactive surfaces. This state seems to have remained unchanged in the decade that has passed since this workshop [134]. Since then, a related metamodel has been proposed for pen-and-paper input. However, it has not been adapted for the more general context of interactive surfaces [93].

The related work presented in this section highlights the benefits of focusing on models as end products. There is a need to lend these benefits to the space of interactive surface environments. Facilitating a method that enables contributing models as end products would (1) promote expressing and communicating prototypes unambiguously, (2) enable systematic assessment of prototypes, and (3) facilitate concept reuse, i.e., address the replication crisis for interactive surface prototypes. We, therefore, propose the Artefact framework.

## 2.6 THE ARTEFACT FRAMEWORK

The Artefact framework enables viewing an interactive surface prototype as a system comprising of the hardware, middleware, and software subsystems. Each subsystem promotes identifying, factoring, and reusing domain-specific concepts, properties, and associations. Here, the term concept is used to refer to a component or a set of components as applied to realize domain-specific functionality. The framework employs UML [159] to represent each subdomain as an unambiguous model. UML provides a formal syntax which promotes logical and consistent model representation of interactive surface prototypes. In addition to generalization hierarchies that facilitate flexibility, another significant benefit to UML is its extensive documentation [159].

In the following, we introduce the Artefact framework and how it leverages stereotype-based modeling. We also discuss our approach to establishing evidence-based modeling constructs for the proposed hardware, middleware, and software models.

### 2.6.1 Stereotype-based modeling

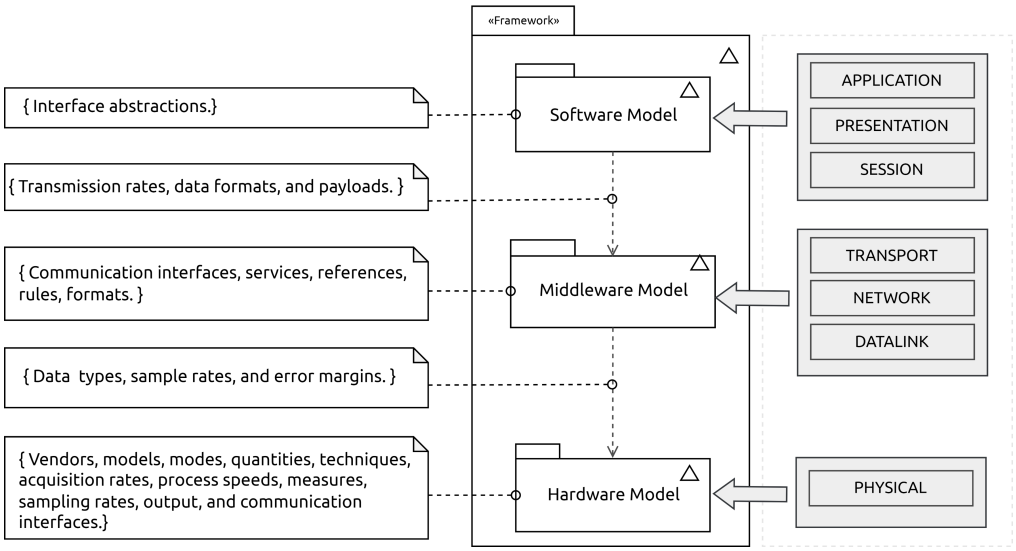



Fig. 4. Overview of the Artefact framework. For an interactive surface prototype, we describe the hardware, middleware, and software as complementary models where, a *Model* is an abstract description of a domain's environment, i.e., the organization of its concepts and associations.

The Artefact framework employs metaclasses, stereotypes, profiles, and tag values; UML mechanisms that extend the normative UML [159]. In contrast to the conventional *Class*, i.e., from normative UML, Metaclasses are abstract *Class* representations. Profiles are package modules required to define stereotypes: concrete *Metaclass* extensions, that can only be defined by extending existing metaclasses. Profiles can extend other profiles and can also be factored for reuse [159]. Akin to a normative UML *Class*, a *Stereotype* also uses structural compartments for describing attributes, members, and tag values [159].

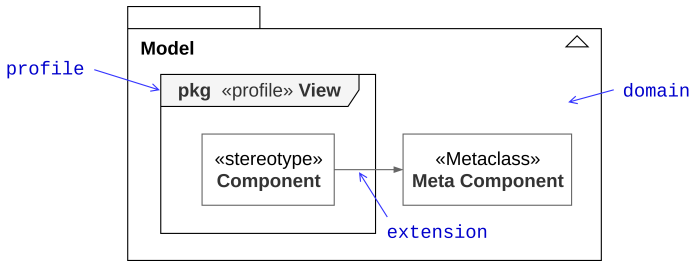



Fig. 5. Stereotype-based modeling. Metaclasses, profiles, and stereotypes are used to express a domain's environment.

### 2.6.2 Evidence-based modeling constructs

We conducted a targeted literature review to identify hardware, middleware, and software requirements essential for developing interactive surface prototypes. The literature study prompted learning evidence-based engineering requirements, i.e., requirements based on existing research prototypes. As part of the review process, we formulated verification questions (see Tab. 1) to promote determining relevant studies. The methodology of the targeted literature review as well as an overview of the selected studies are presented in Appendix A.

Table 1. Verification questions formulated to target studies with well-rounded artifacts.

| | **Verification questions** | **Points** |
|---|---|---|
| Q1 | Does the study present a clear research goal? | 2 |
| Q2 | Is the presented research prototype adequate for the stated research goal? | 2 |
| Q3 | Does the study outline hardware, middleware, and software utilized? | 2 |
| Q4 | Does the study justify the choice of hardware employed? | 2 |
| Q5 | Does the study utilize commodity hardware? | 2 |
| Q6 | Does the study motivate the software implementation? | 2 |
| Q7 | Does the study employ a repeatable validation procedure? | 2 |
| Q8 | Are validation metrics outlined? | 2 |
| Q9 | Are limitations of the research prototypes discussed? | 2 |
| Q10 | Is the efficacy of the utilized concepts outlined? | 2 |
| Q11 | Are the design and implementation strategies of the software discussed? | 2 |
| Q12 | Is the research goal realized? | 2 |

Interactive surface prototypes were examined to identify hardware, middleware, and software components. Subsequently, common underlying components were abstracted into modeling constructs (see Tab. 3). As evinced by the literature study: software-design considerations for interactive surfaces are typically omitted in publications. During study verification, no studies answered questions **Q**6 and **Q**11 (see Tab. 1), which were formulated to identify studies with software development strategies. This observation is again consistent with the "replication crisis" [51, 208]. Therefore, we studied the ReFlex framework to learn evidence-based software engineering requirements.

ReFlex provides technical guidelines and design considerations for developing interactive surface software. Tab. 2 presents studies that have employed the ReFlex framework to implement research prototypes [68, 112, 113, 152, 168]. In section 5 we present a case study that elaborates further on ReFlex framework. In addition to studying the ReFlex for evidence-based modeling constructs (Tab. 2), modeling constructs were also inferred from prototypical applications (see Tab. 6 in Appendix A).

Table 2. Studies that have employed the ReFlex framework to implement research prototypes.

| Study | Focus | Prototype | Validation |
|---|---|---|---|
| DepthTouch: An elastic surface for tangible computing [168] | Emulating physical surface interactions | Interactive surface system for back-projection on tabletop surfaces using a mirror | Demonstration |
| FlexiWall: Interaction in-between 2D and 3D interfaces [68] | Toolkit for exploring layered data | Interactive surface system for back-projection on walls using a mirror | Categorization of data types and interaction techniques |
| Data exploration on elastic displays using physical metaphors [150] | Simulation of physical interaction metaphors on deformable interactive surfaces | Interactive surface system for tabletop surfaces | Demonstration |
| Exploring big data landscapes [112] | Visualization and analysis of clustering algorithms for interactive surfaces | Interactive surface system for wall surfaces | Demonstration |
| A tangible concept for layered map visualizations [153] | Exploring layered data combined with semantic zoom visualizations | Client-server middleware | Demonstration |
| Designing interfaces for elastic displays using workshop and prototyping methods [83] | Shared description for design Process | Exploring input modalities | Demonstration |

Table 3. Hardware, middleware, and software modeling constructs. Common underlying components were identified generalized to stereotypes. The stereotypes were then abstracted into high-level metaclasses. For the software modeling constructs, in addition to studying the Reflex framework, flexible stereotypes were inferred based on the reverse engineering of existing software solutions.

| Metaclass | Stereotype | Component | Body of literature |
|---|---|---|---|
| **Hardware modeling constructs** | | | |
| Device | System Processor | Workstation, Server | [50, 55, 57, 81, 86, 114, 118, 155, 165, 175, 202, 217, 219, 220, 223, 225, 226] |
| | | Microprocessor | [86, 165] |
| Device | Sensor | Camera Sensor | [50, 55, 81, 86, 114, 155, 165, 175, 202, 217, 223, 225, 226, 229] |
| | | Depth Sensor | [50, 55, 86, 155, 165, 219, 220, 223, 226, 229] |
| | | Infrared Sensor | [118, 202] |
| Device | Display Device | Projector, Display | [50, 57, 81, 86, 118, 155, 165, 175, 202, 217, 220, 223, 225, 226, 229] |
| Device | Actuator | Actuator | [81, 118, 175, 202, 223, 226] |
| Device | Human Interface Device | Mobile Device | [86, 114, 155, 165, 175, 217, 223, 226] |
| Device | Peripheral Device | Phicon | [202] |
| | | Marker | [114, 118, 175] |
| | | Mirror | [57, 118, 202] |
| Device | Audio Device | Transducer | [57] |
| **Middleware modeling constructs** | | | |
| Architecture | Transmission Service | Framework | [50, 109] |
| | | Specification, Protocol | [107, 108, 110] |
| **Software modeling constructs** | | | |
| Manager | Display manager | | [55, 86, 114, 118, 155, 165, 217, 220, 226, 229] |
| Manager | Tracking manager | | [55, 118, 155, 165, 217, 223, 229] |
| Manager | Service manager | | [55, 57, 81, 86, 114, 118, 155, 165, 217, 220, 225, 229] |
| Manager | Calibration manager | | [68, 78, 83, 112, 113, 150–154, 168] |

### 2.6.3 The hardware, middleware, and software metamodels

Metamodels for the hardware, middleware, and software (see Fig. 6) were developed using the constructs presented in Tab. 3. [1] Metamodel formulation was based on outlines specified in [159], which follow: For a stereotype to be well-formed, it must exist within a *Profile* and be linked to a *Metaclass* and there are no restrictions on the number of profiles, metaclasses, and stereotypes that can be introduced [159]. Note that the profiles do not add any functionality and only describe a unified view for the underlying system. Akin to normative UML, concrete stereotypes in the hardware, middleware, and software models utilize structural-class compartments. This feature facilitates documenting arbitrary technical specifications using descriptions of attributes, members, and values. Metaclasses reflect a domain's environment and, by design, are high-level abstractions that remain adaptable for different application scenarios. The role of meta-classes is to enable modeling specifications using meta attributes, viz, tagged values. Leveraging UML's extension association, stereotypes afford the description of arbitrary information to meta-classes, i.e., using tagged values. SoaML and DICOM are good examples of how meta-classes are extended to model specifications using meta attributes [44, 239]. For engineering software solutions, the software meta-model offers a generic way to design, document, and share application programming interfaces.

## 2.7 EMPIRICAL VALIDATION

Given that there is no standard method for determining the absolute correctness of a model [159], for validation, we adopted a two-fold approach. First, the Artefact framework was employed to describe ten research artifacts, i.e., those presented in [68–70, 80, 106, 150, 153, 168, 227, 230].

The artifacts selected were subject to the selection criteria outlined for identifying relevant studies and prototypes in Appendix A. Services, formats, transmission rates, vendors, models, modes, quantities, process speeds, sampling rates, and communication interfaces were captured and expressed using tag values, directed relations, cardinalities, stereotype compartments. Once captured (graphical-model representation presented in Appendix B), we used the resulting model descriptions for validation. In [184], Sargent suggested evaluating the correctness of a model using "face validation", i.e., subjecting a model to scrutiny of experts. Following this guideline, we conducted an interview study to determine whether experts considered the proposed modeling framework reasonable and acceptable. The study was also used to gain insights about perceptions towards adopting a model-driven approach to support developing interactive surface research prototypes.

A call for participation was sent out to research centers at KTH Royal Institute of Technology, the University of Regensburg, and Bauhaus-Universität Weimar. We also

[1] A meta-model is an abstract model that provides ontology mapping [144]. It describes modeling constructs that can be utilized to define concrete models.

invited industry experts from Extend3D GmbH, Munich. Of twelve experts, five met the requirement of at least two years working with interactive surface environments or components associated with interactive surface environments (e.g., projector-camera

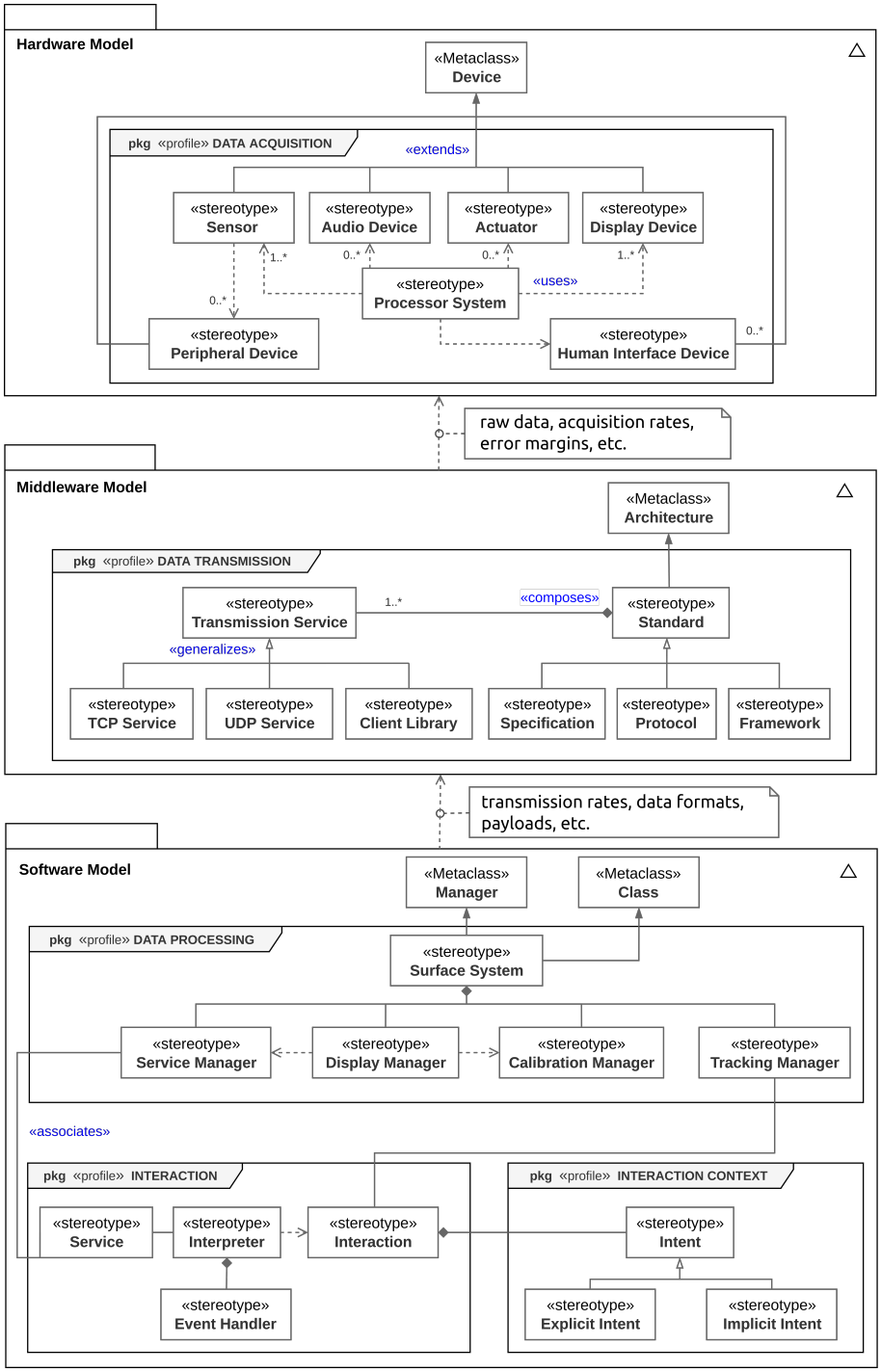



Fig. 6. The hardware, middleware, and software meta-models. Profiles and metaclasses are used to describe a domain's environment. Properties and tags are used to convey arbitrary technical specifications. Directed relationships and cardinality are utilized to describe associations. Each model is flexible and extensible to variable application scenarios.

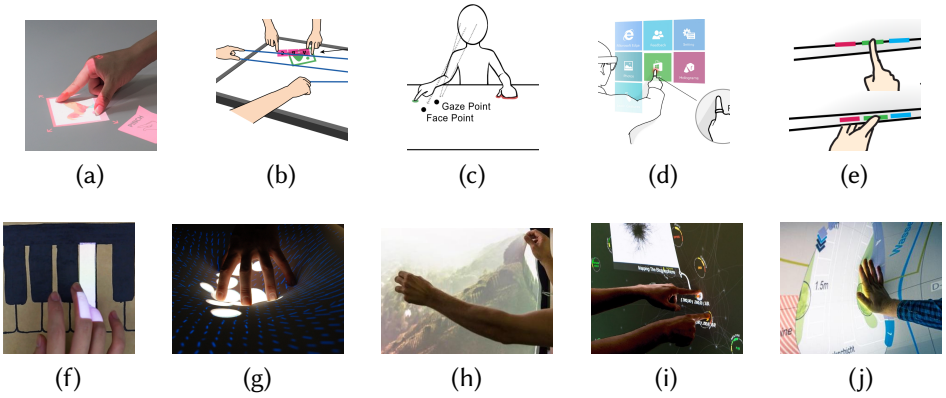



Fig. 7. Prototypes selected for validation. In (a), an overhead projector-camera system for tabletop interaction [69]. In (b), a digital tabletop (Panasonic TH-50PH12) and motion-track system for gesture detection [70]. In (c), a digital tabletop with head-mount gaze and eye-tracking for determining target interaction points [230]. In (d), wearable devices for surface interaction on surfaces [80]. In (e), a projector-camera system with a 10-camera Vicon system for tracking touch input on tabletop edges [106]. In (f), an overhead projector-camera system for tabletop and tabletop-object interaction [227]. In (g), *DepthTouch*, an elastic tabletop display [168]. In (h), *FlexiWall*, an elastic display wall for image blending [68]. In (i), Zoomable UI combined with physical interaction metaphors, called *DeeP* [150]. In (j), an elastic display using semantic layers and magic lenses [153].

systems). The interviews were conducted over Skype, and an online form was used to collect expert perceptions. Each interview was structured as follows: First, the research aim was discussed. Then, we asked the experts to confirm the suitability of their experience given the context of the research. Afterward, experts were introduced to the proposed modeling framework. Thereafter, we introduced the hardware, middleware, and software metamodels. After discussing the rationale and objectives of each model, we presented captured model representations of the prototypes. We asked the experts to scrutinize and remark on using the framework to model prototypes. Lastly, we discussed whether a model-driven approach would benefit the research community for developing prototypes in the early design stages. Discussion with all experts was structured using questionnaires. All questions were open-ended towards deeper discussions, as led by the experts. This approach promoted collecting experts' perceptions and learning their opinions on setbacks, benefits, and implications of a model-driven approach using the proposed framework.

Expert #1 remarked on the rationality of the models, *"The models are rational and clear."* Expert #2 gave merit to unambiguous technical outlines, *"…specifications simplify prototyping."* Expert #3 pointed to capitalizing existing prototypes, *"…developers can obtain structured information about existing prototypes…"* Expert #4 remarked on learnability, *"…it took me ~5 minutes to grasp the framework."* Expert #5 underlined inherent fostering of artifact replication, *"…full specifications simplify replicating prototypes."*

Data collected during the interviews was coded. Our findings suggest that experts found the proposed framework to be beneficial with perceptions converging toward three main benefits: (i) a generic and simple syntax for prototyping, (ii) an approach to systematically compare different prototypes, and (iii) a convenient starting point for developing interactive surface prototypes.

## 2.8 CASE STUDY: THE REFLEX FRAMEWORK

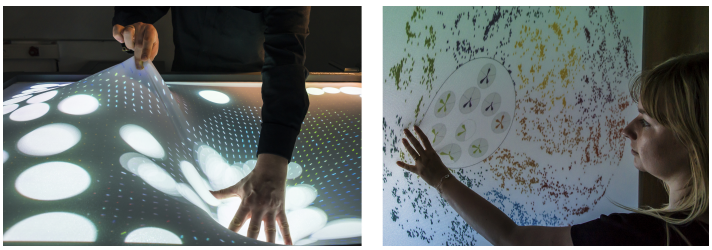
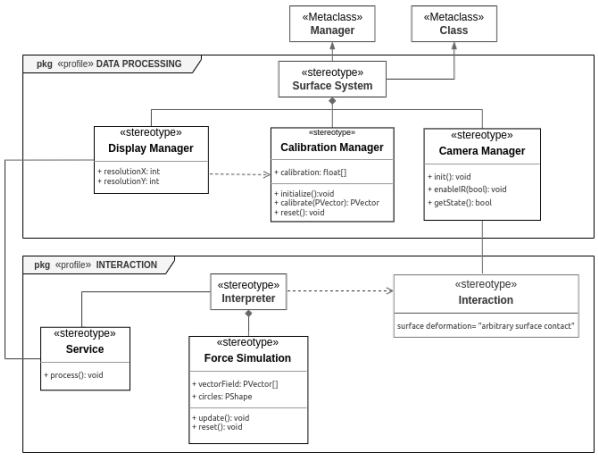


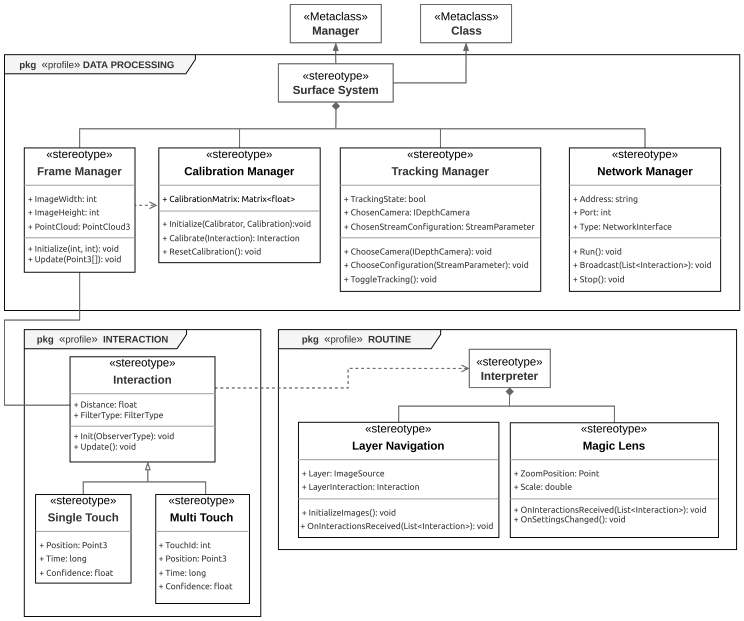
Fig. 8. ReFlex is a modular software framework for elastic displays that has been developed over the last ten years. It started as a proof-of-concept [168] and evolved into a platform-independent framework that utilizes a modular architecture with abstraction layers for different application scenarios. The framework has been used for prototypes in several publications, e.g., surface deformation interaction (left) and Zoomable user interfaces (right). It has served as a foundation for a task taxonomy for elastic displays [113]. ReFlex, in its current state, employs web-technologies for visualization, provides interface to different client technologies (e.g., Angular/React, .NET Core/WPF or Unreal Engine 4), and provides several tools for developing applications for elastic displays.

For proof-of-concept, we conducted a case study with software developers who have been actively developing the ReFlex framework over the last ten years. As already mentioned, ReFlex is a software framework that provides technical guidelines and design considerations for developing interactive surface software. ReFlex builds on an initial concept proposed by Peschke et al. in [168] and targets enabling the development of application programming interfaces (APIs) for interactive elastic displays. It also supports interactive visualizations using different interaction techniques [113]. The ReFlex framework has been progressively developed and used to implement several prototypes over the last ten years [68, 112, 113, 152, 168]. ReFlex's development has been driven by the demand to adapt APIs to rapidly evolving technology.

Given transparent software-development trails, using ReFlex for the case study enabled assessing how the Artefact framework can support developers to design and communicate interactive surface software. With an emphasis on models as end products [184], the hypothesis for the case study was as follows. "The Artefact framework provides a flexible approach to modeling different software specifications. Its software metamodel promotes clear design and unambiguous documentation of concrete APIs."




(a) Case study: software model from [168].




(b) Case study: software model from [153].

Fig. 9. While the API presented in (a) centered on processing depth data into a vector field for physical force simulation based on surface deformation, the API presented in (b) explored identifying and differentiating between single and multi-touch interactions. The API in (b) also targeted interpreting interactions to navigate layered images while utilizing a magic lens for visualization.

### 2.8.1 Employing the Artefact framework to model research artifacts developed using ReFlex

The software developers selected four prototypes developed using the ReFlex framework. Akin to the validation presented in section four, APIs were captured and expressed as models. Once captured, together with the software developers, we discussed the benefits of the model-driven approach. To evaluate the hypothesis, we learned whether software developers considered the resulting software models beneficial. The

case study also aimed to learn general perceptions models as artifacts. As a starting point, the software developers identified the software interfaces of the first elastic display prototype [168]. Then, they modeled the components using Artefact's software metamodel (see Fig. 9). For comparative assessment, a similar approach was employed for remaining prototypes. Together with the software developers, we discussed the resulting model representations. We also discussed the benefit of using metamodels as flexible guidelines to support capturing and expressing APIs unambiguously.

### 2.8.2 Findings

Artefact's software metamodel provided a flexible guideline for instantiating concrete stereotypes, thereby supporting abstract modeling of different APIs unambiguously. Moreover, the resulting model representations facilitated direct comparison between different APIs. This finding supported the first supposition: "The Artefact framework provides a flexible approach to modeling different software specifications." The software developers emphasized the utility in knowledge transfer, pointing out the potential role of diffusing API models to bridge the gap between originators of concepts and future developers. Direct comparison between different APIs also promoted identifying advances in API development. This finding also supported the second supposition: "The Artefact's software metamodel promotes clear design as well as unambiguous documentation of concrete API designs." Additionally, the software developers pointed out that the resulting model representations were technology agnostic. Despite employing different technologies in the different software solutions, i.e., a OpenNI-based implementation for one prototype and a .NET and WPF-based implementation for the other, the designs of the APIs remained comparable. Furthermore, the software developers pointed out that focusing on modeling high-level abstractions promoted understanding of structurally complex software concepts. Another benefit identified by the software developers was that unambiguous API models simplified identifying entry points for possible API extensions, even without comprehension of the entire API. This benefit was underlined as essential for advancing existing interactive surface concept.

Going one step further, the software developers discussed how the resulting model artifacts could, in principle, be leveraged to generate software documentation using tools such as doxygen[2] and graphvis[3]. Doxygen grouping mechanism could be utilized to declare API models, generate an overview of classes, and specify the stereotypes as nested elements. For structurally complex APIs, custom doxygen tags could be extracted for creating specific API-model components using the graphviz dot tool.

## 2.9 DISCUSSION

Given the verification questions (see Tab. 1), body of literature suggests there are no shared approaches to analyzing and designing interactive surface prototypes. The

[2]https://www.doxygen.nl/index.html
[3]https://graphviz.org/

absence of such approaches is made apparent by question **Q**11, for which no study accounts. Also, questions **Q**3, **Q**4, **Q**6, and **Q**10, indicate there is no shared view, widespread representation, or unified syntax for designing, communicating, or documenting prototypes. The literature review also highlighted the limited extent to which middleware has been explored for interactive surface applications.

*Limitations.* Our validation approach suffers from the absence of a standard methodology for evaluating the *"absolute"* validity of models [159, 184]. Although an argument can be made for the intuitive representation of structurally complex systems, our proposed modeling framework is not comprehensive by design. Absent an extensive set of all possible abstractions, the minimal set presented in this paper is insufficient for non-experts who may also seek to employ the modeling framework as a knowledge base. Another limitation is that our validation was limited to six experts.

*Future.* Our empirical validation demonstrates the role of experts in the iterative development of the framework. In this regard, there is promise in conducting validation on a larger scale and learning expert perceptions towards *"models as end products"* [4], i.e., in the space of interactive surface environments. Moreover, experts have pointed out the benefit of open discourse about design considerations for such a modeling framework. Conducting workshops with experts and developers would be one possible approach to promote such a discussion.

## 2.10 CONCLUSION

This paper identifies the accelerated accumulation of comparable prototypes for interactive surface environments and proposes a model-driven approach to promote design reuse. Existing prototypes have been leveraged to define a generic UML-based framework for modeling hardware, middleware, and software layers of interactive surface prototypes. The proposed framework has been applied to capture existing prototypes, and a study has been conducted to learn experts' perceptions towards the captured model representations. By modeling our new ReFlex framework for elastic displays, we further show the validity and tacit benefits to the proposed model-driven approach. Our initial findings highlight three significant benefits: (i) an accessible graphical syntax with unambiguous model representation, (ii) a system for capturing arbitrary technical specifications, and (iii) flexible model representation with consistent notation. While no absolute conclusions can be drawn, initial results suggest significant benefits in the Artefact framework.

## ACKNOWLEDGMENTS

This work was supported by the German Federal Ministry for Education and Research (BMBF) through grant 16SV8288 as part of the VIGITIA project.

## 2.11 APPENDIX

## A LITERATURE REVIEW

Fig. 10 provides an overview of the methodology employed for developing the Artefact framework. A targeted literature review was used to answer the question: What hardware, middleware, and software engineering requirements are necessary to develop an interactive surface prototype? This central question promoted identifying interactive surface prototypes and learning engineering requirements that have been considered for existing implementations.

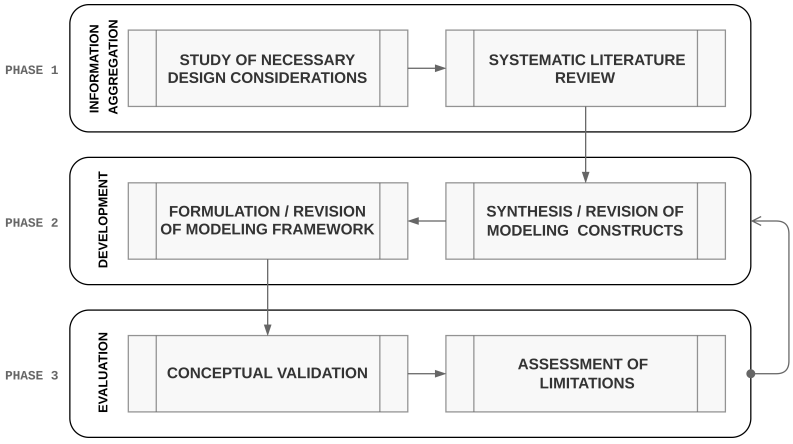



Fig. 10. Methodology for developing the Artefact framework: First, a relative knowledge base was systematically aggregated. Then, common underlying concepts were identified and modeling constructs were derived systematically to formulate flexible hardware, middleware, and software models. In the final phase, which is perpetual by definition, the proposed framework is evaluated and developed iteratively.

In the following, we elaborate on how the targeted literature review was conducted, discussing the search strategy, selection criteria, and verification of selected literature. We also present an overview of the final studies used to develop the proposed framework.

### A.1 Search strategy

The ACM Digital Library, CiteSeerX, and the IEEE Xplore Digital Library were databases of choice. Where possible, advanced searches were used to exclude non-refereed publications and books.
We inferred a list of search strings from keywords such as interactive surfaces, tabletop interaction, projector-camera systems, and spatial augmented reality. We then used the search strings to find publications in aforementioned databases.

### A.2 Initial selection of related work

Besides appraising titles, we reviewed each paper's abstract, introduction, discussion, and conclusion subject to non-strict criteria that considered;

- publication—the study must be published,

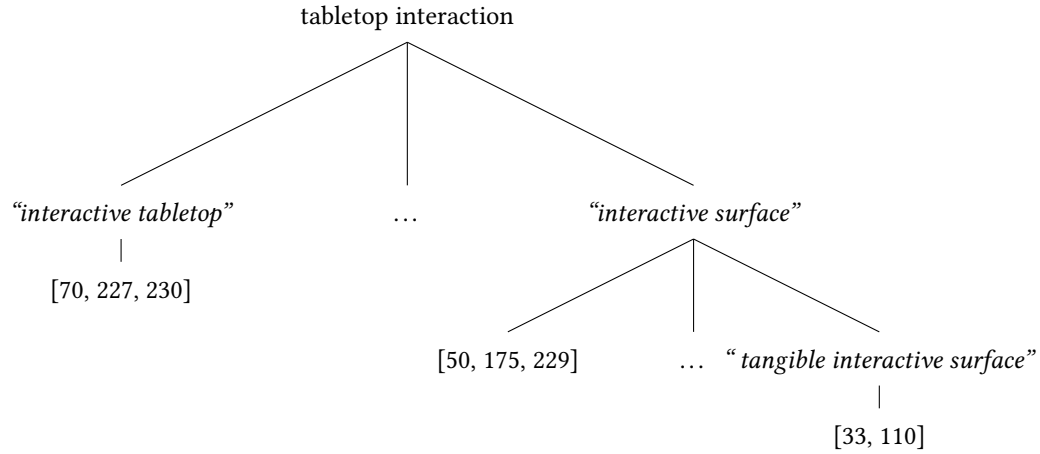



Fig. 11. Targeted search. Example of how the keyword ‘tabletop interaction’ was used to infer a list of database search strings (e.g., *“interactive tabletop”*, *“interactive surface”*, and *“tangible interactive surface”*) to identify publications.

- an artifact—the study must contribute a research prototype,
- peer review—the study must be subjected to the scrutiny of field experts,
- proof of concept—the study’s prototype must demonstrate validity, and
- prototype description—the study must outline development in a manner that enables artifact reproducibility.

While rudimentary, this approach simplified excluding non-relevant studies and building up an initial body of literature to be verified.

### A.3 Verification

In addition to the verification questions (see Table 1), an analysis scheme was used to assess each study. For study assessment, mark allocation was as follows: If a verification question was not answered, no points were allocated. If an answer to a question was inferred but not stated explicitly in the study, 1 point was allocated, and if a question was answered directly by the study, 2 points were allocated. Given a total number of 12 questions, the maximum possible score a study could achieve was 24. Each study’s score was then expressed as a percentage, and the corresponding percentage equivalency (Table 4) assigned.

A scale of $1 - 5$ was introduced, 1 being the highest possible score and 5 being the lowest possible score. Studies were selected in a three-stage process. In the first stage, initial scores were allocated. In the second stage, excluded studies were double-checked. Last, accepted studies were also double-checked. We note, no scores were changed during the double-checking.

### A.4 Overview of selected studies

Our literature search identified 136 studies. In evaluating the identified studies against the selection criteria, 47 studies were excluded. During verification, 63 studies were

Table 4. Percentage equivalency and grade descriptors. A grade more than 3 was interpreted as an indication that the study in question would provide insufficient information towards answering the outlined research question.

| Percentage | Grade | Descriptor |
|---|---|---|
| 91 - 100% | 1 | Highly relevant |
| 81 - 90% | 2 | Relevant |
| 66 - 80% | 3 | Moderately relevant |
| 50 - 65% | 4 | Marginally relevant |
| 0 - 49% | 5 | Irrelevant |

Accept studies $\leq 3$ Exclude studies $> 3$

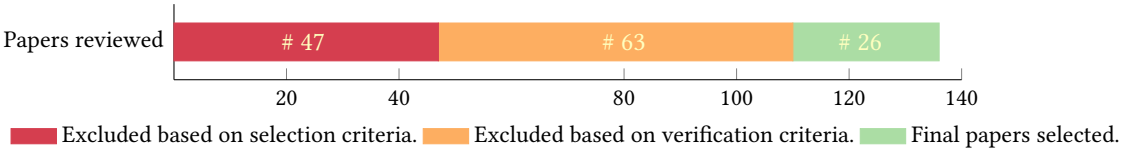



Fig. 12. Overview of excluded studies.

excluded. In summary, 26 published and peer-reviewed papers were selected. We organized the selected studies using Mendeley, which simplified ordered caching of bibliographies. Table 5 lists the publication sources and Tables 6 provides a summary of studies selected during the targeted literature review.

Table 5. Literature sources.

| Publication | Type | Year |
|---|---|---|
| Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies | Journal | 2020 |
| CHI Conference on Human Factors in Computing Systems | Conference | 2019 |
| Proceedings of the 31st Australian Conference on HCI | Conference | 2019 |
| Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology | Conference | 2019 |
| Proceedings of the CHI Conference on Human Factors in Computing Systems | Conference | 2019 |
| Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology | Conference | 2018 |
| Proceedings of the ACM International Conference on Interactive Surfaces and Spaces | Conference | 2018 |
| Proceedings of the ACM on Human-Computer Interaction | Journal | 2018 |
| Proceedings of the ACM International Conference on Interactive Surfaces and Spaces | Conference | 2017 |
| Conference on Human Factors in Computing Systems | Conference | 2014 |
| Proceedings of the 9th ACM International Conference on Interactive Tabletops and Surfaces | Conference | 2014 |
| Proceedings of the SIGCHI Conference on Human Factors in Computing Systems | Conference | 2013 |
| Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology | Conference | 2012 |
| Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology | Conference | 2011 |
| Proceedings of the 23rd ACM Symposium on User Interface Software and Technology | Conference | 2010 |
| Proceedings of the SIGCHI Conference on Human Factors in Computing Systems | Conference | 2012 |
| Proceedings of the 5th Nordic Conference on Human-Computer Interaction | Conference | 2008 |
| Proceedings of the 15th IEEE International Workshops on Enabling Technologies | Conference | 2006 |
| Proceedings of the Annual ACM Symposium on User Interface Software and Technology | Conference | 2005 |
| Proceedings of the 6th International Workshop on Gesture in HCI and Simulation | Conference | 2005 |
| ACM Transactions on Computer-Human Interaction | Conference | 2001 |
| IEEE and ACM International Symposium on Augmented Reality | Conference | 2000 |
| Conference on Human Factors in Computing Systems | Conference | 1999 |
| Proceedings of the 10th annual ACM Symposium on User Interface Software and Technology | Conference | 1998 |

Table 6. Reviewed literature.

| Study | Score | Focus | Approach | Validation |
|---|---|---|---|---|
| Off-surface tangibles: Exploring the design space of midair tangible interaction Cherek et al. | 2 | Midair interaction above interactive surfaces | Exploring input modalities | Demonstration |
| HapTable: An interactive tabletop providing online haptic feedback for touch gestures Emgin et al. | 2 | Haptic feedback for touch interaction on display surfaces | Design and implementation of a multi–modal digital interactive display | Usability study |
| Occlusion-aware hand posture based interaction on tabletop projector Fujinawa et al. | 3 | Finger occlusion on interacting with projected images | Depth sensor-based evaluation of hand posture | Demonstration |
| Ray-casting based interaction using an extended pull-out gesture for interactive tabletops Fujita et al. | 2 | Exploration for a pull-out gesture | Ray-casting based interaction | Demonstration |
| Accurate and low-latency sensing of touch contact on any surface with finger-worn IMU sensor Gu et al. | 1 | Optimization of wearable finger touch sensing on surfaces | Evaluation of response rates based on different finger modalities | Evaluation of recall rates of contact sensors |
| An evaluation of touch input at the edge of a table Joshi and Vogel | 1 | Surface-edge interaction for occluded tabletops | Exploration of edge tailored gestures | Usability study |
| MagicPAPER: Tabletop interactive projection device based on tangible interaction Wu et al. | 2 | Interactive tabletop projection | Development of a multipurpose sensor | Demonstration |
| Recognizing unintentional touch on interactive tabletop Xu et al. | 2 | Robust touch detection for surface interaction | Study of user behavioral patterns on tabletop surfaces | Comparative |
| Surfacestreams: A content-agnostic streaming toolkit for interactive surfaces Echtler | 1 | Communicating interactive surface information | Open source development of a information transmission pipeline | Demonstration |
| The TUIO 2.0 protocol: An abstraction framework for tangible interactive surfaces Kaltenbrunner and Echtler | 1 | Formal description for tabletop semantics | Definition of properties for common controller objects | — |

Continued on next page

**Table 6 — continued from previous page**

| Study | Score | Focus | Approach | Validation |
|---|---|---|---|---|
| Prodesk: An interactive ubiquitous desktop surface Wingert et al. | 1 | Tabletop surface interaction | Extending WIMP functionality and surface user interface concepts | Usability study |
| The interactive dining table, or pass the weather widget, please Echtler and Wimmer | 2 | Augmenting tabletops with projection images | Integrating low-cost projector-camera systems into everyday environments | Living lab report |
| UbiBeam: An interactive projector-camera system for domestic deployment Gugenheimer et al. | 1 | Qualitative study on the practicality of projector-camera systems | Study of interaction modalities | Usability study |
| Pervasive information through constant personal projection: The ambient mobile pervasive display (AMP-D) Winkler et al. | 1 | Evaluation of wearable projector system | Study of continuous interaction space controlled using hand gestures | Demonstration of interaction metaphors and use cases |
| WorldKit: Rapid and easy creation of Ad-hoc interactive applications on everyday surfaces Xiao et al. | 3 | On-the-fly instantiation of interactive surfaces in everyday environments | Abstracting and extending existing projector-camera system implementations | Demonstration of use case scenarios |
| Extended multitouch: Recovering touch posture and differentiating users using a depth camera Murugappan et al. | 1 | Multi-user tabletop touch interactions | Study of multi-user input modalities | Usability study |
| Steerable augmented reality with the Beamatron Wilson et al. | 1 | Steerable surface projections | Depth camera-based interactions | Demonstration of multi-view projection |
| OmniTouch: Wearable multi-touch interaction everywhere Harrison et al. | 1 | Pervasive interaction on surfaces in everyday environments | Wearable depth sensing | Usability study |

Continued on next page

**Table 6 — continued from previous page**

| Study | Score | Focus | Approach | Validation |
|---|---|---|---|---|
| Combining multiple depth cameras and projectors for interactions on, above, and between surfaces Wilson and Benko | 1 | Omni-directional interaction with surfaces | Processing multiple depth sensor view perspectives | In the wild usability study |
| A multitouch software architecture Echtler and Klinker | 2 | Layered software architecture for surface interaction interfaces | Systematic review of existing interaction interfaces | Case study |
| The reacTable*: A Collaborative Musical Instrument Kaltenbrunner et al. | 2 | Tabletop tangible user interface | Fiducial marker vision engine | Case study |
| TUIO: A protocol for table-top tangible user interfaces Kaltenbrunner et al. | 1 | Formal description for tabletop semantics | Definition of properties for common controller objects | — |
| PlayAnywhere: A Compact Interactive Tabletop Projection-Vision System Wilson | 1 | Mobile front-projection for surfaces in everyday environments | study Surface projection and user interaction sensing in small form factor, mobile devices | Demonstration of interactive surface applications in different environments |
| Virtual object manipulation on a table-top AR environment Kato et al. | 3 | Detecting user input for virtual object manipulation | Augmenting interactive surface tangibles with imagery | Usability study |
| Integrating paper and digital Information on EnhancedDesk: A method for realtime finger tracking on an augmented desk system Koike et al. | 2 | Design and implementation for tabletop surface | Exploring design strategies for interactive surface implementations | Case study |
| Augmented surfaces: A spatially continuous work space for hybrid computing environments Rekimoto and Saitoh | 1 | Design and implementation for spatially continuous interactive surface environments | Exploring continuous input modalities | Case study |

Continued on next page

**Table 6 — continued from previous page**

| Study | Score | Focus | Approach | Validation |
|---|---|---|---|---|
| The metaDESK: Models and prototypes for tangible user interfaces Ullmer and Ishii | 2 | Platform for developing interaction techniques | Describing interface abstractions for implementing interactive tabletop surfaces | Case study |

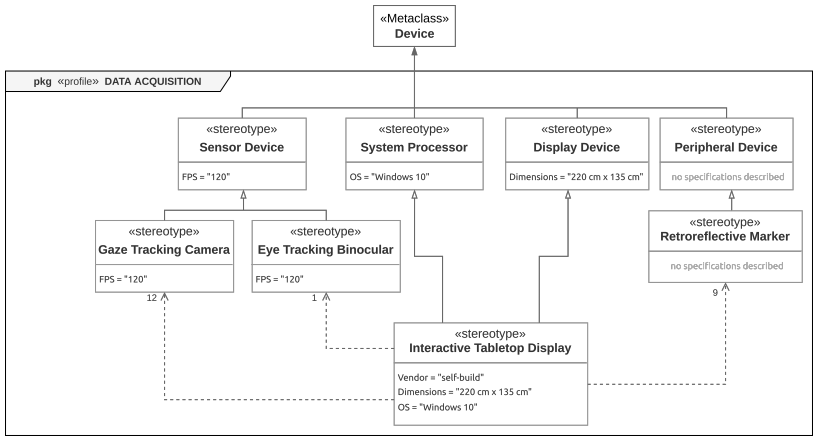
## B FRAMEWORK EVALUATION: MODELING EXISTING RESEARCH PROTOTYPES

As there is no standard methodology for evaluating model correctness [159, 184], for validation, we employed the proposed framework to describe existing research artifacts. For each modeled prototype, we assessed completeness, i.e., whether or not the framework

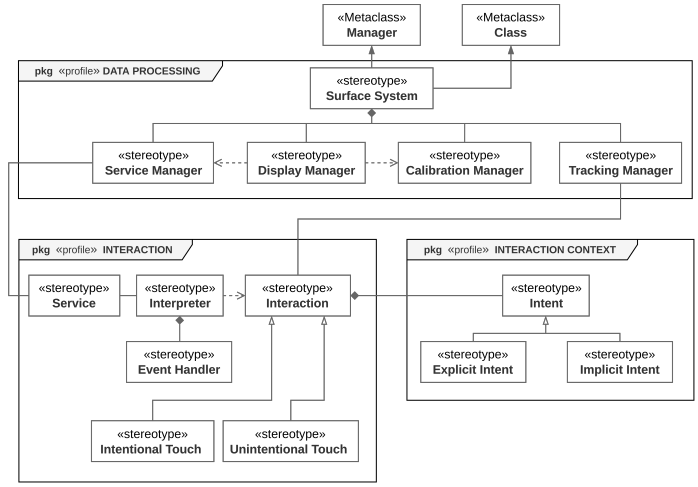
- captured all concepts in the hardware, middleware, and software layers unambiguously,
- captured all associations between identified concepts across the separate layers, and
- conveyed all arbitrary specifications.

We formulated model representations for ten prototypes [68–70, 80, 106, 150, 153, 168, 227, 230]. For each prototype, possible logical views for the hardware, middleware, and software layers were modeled based on technical descriptions provided in corresponding publications. As discussed in section 3, the Artefact framework facilitates modeling specifications, i.e., meta-modeling. The framework also enables instantiating concrete software interfaces for the software layer, i.e., description of APIs. For [69, 70, 80, 106, 227, 230], all high-level meta-models are captured adequately. However, as APIs are not disclosed, the captured software models only describe possible logical views derived from application descriptions in publications. The Artefact framework also demonstrates utility in underlining non-disclosed hardware specifications, i.e., technical descriptions omitted in corresponding publications. Four of the ten models were cross-verified by the original developers of the prototypes [68, 150, 153, 168].

## B.1 Digital tabletop with head-mount gaze and eye-tracking



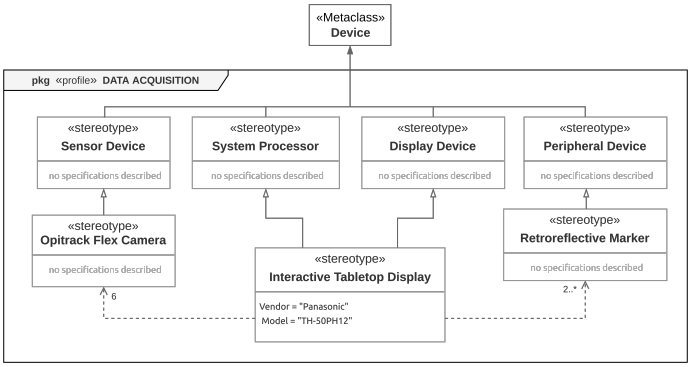

(a) Hardware model.



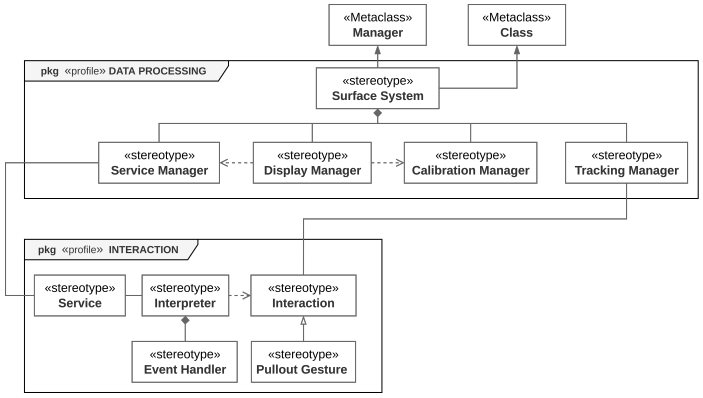

(b) Software model.

Fig. 13. Modeling the digital tabletop prototype. The software model shows a possible logical view derived from the application described by Xu et al. in [230].

## B.2 Digital tabletop (Panasonic TH-50PH12) and motion-track system



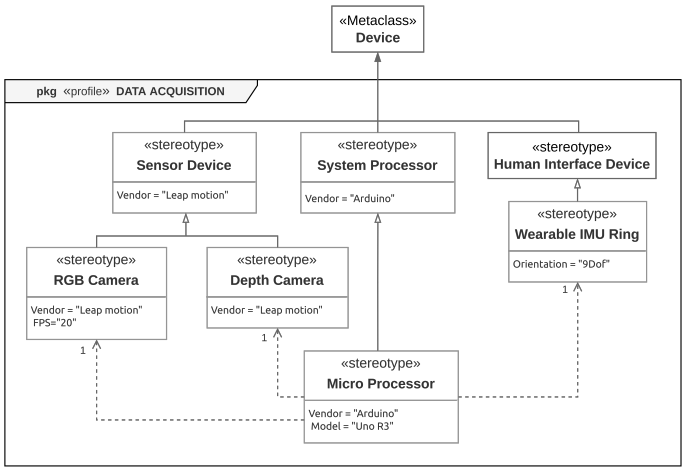

(a) Hardware model.



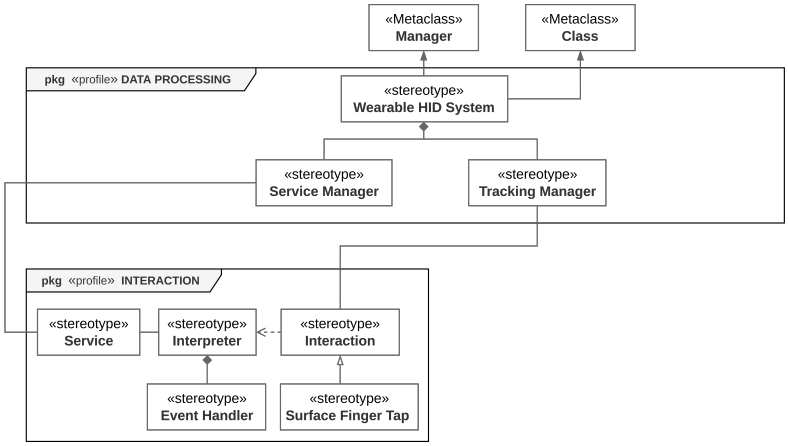

(b) Software model.

Fig. 14. Modeling the digital tabletop prototype. The software model shows a possible logical view derived from the application described by Fujita et al. in [70].

## B.3 Surface interaction wearables



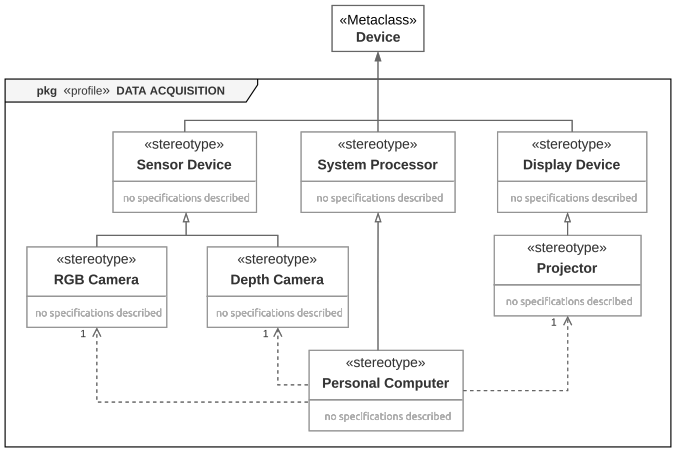

(a) Hardware model.



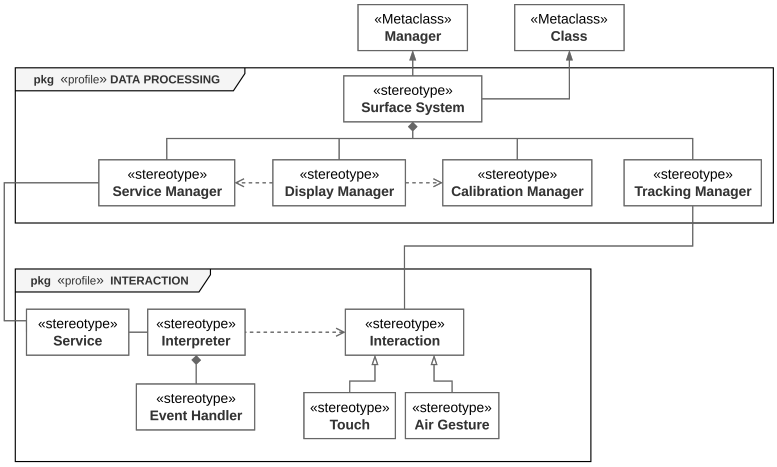

(b) Software model.

Fig. 15. Modeling the surface interaction prototype. The software model shows a possible logical view derived from the application described by Gu et al. in [80].

## B.4 Overhead projector-camera system for tabletops



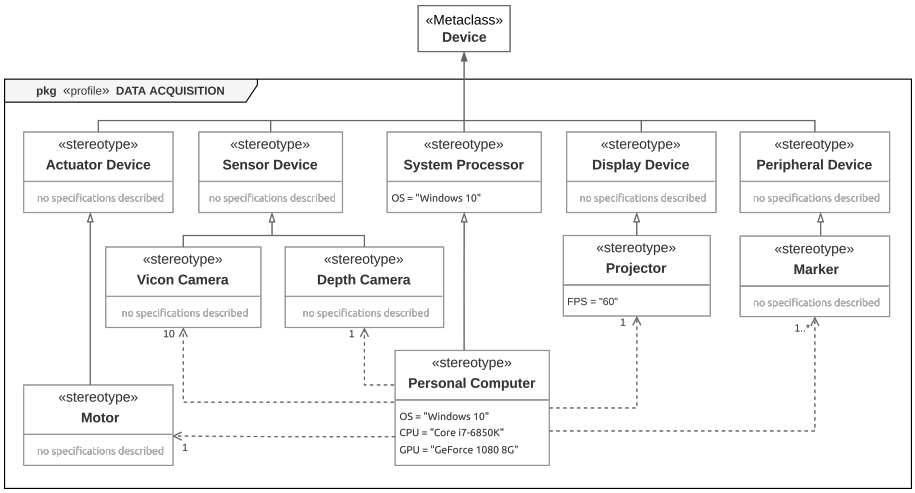

(a) Hardware model.



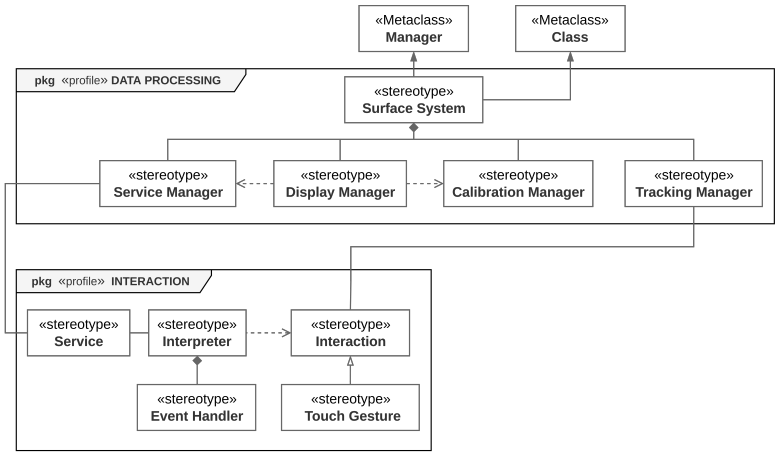

(b) Software model.

Fig. 16. Modeling the projector-camera system prototype. The software model shows a possible logical view derived from the application described by Fujinawa et al. in [69].

## B.5 Projector-camera system with a 10-camera Vicon system for tabletop edges



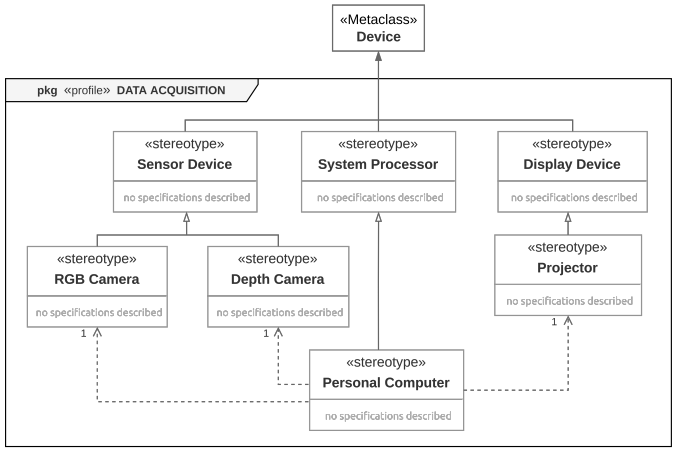

(a) Hardware model.



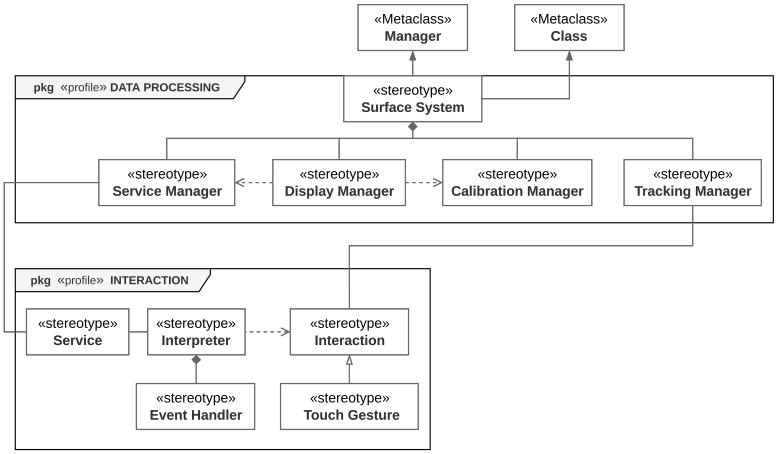

(b) Software model.

Fig. 17. Modeling the projector-camera system prototype. The software model shows a possible logical view derived from the application described by Joshi and Vogel in [106].

## B.6 Overhead projector-camera system for tabletops



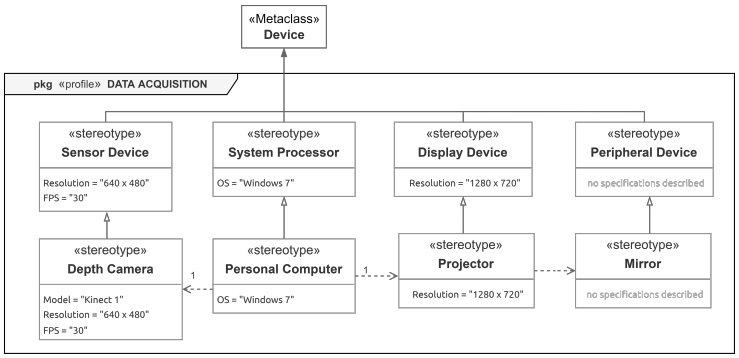

(a) Hardware model.



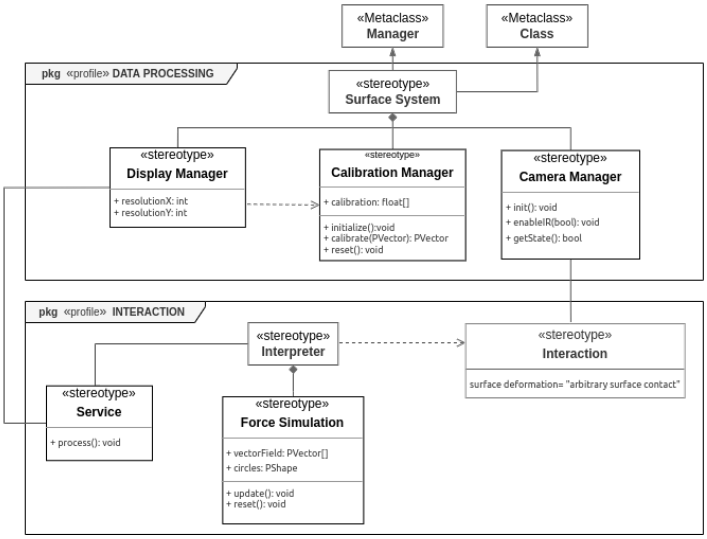

(b) Software model.

Fig. 18. Modeling the projector-camera system prototype. The software model shows a possible logical view derived from the application described by Wu et al. in [227].

## B.7 Back-projected interactive tabletop surface



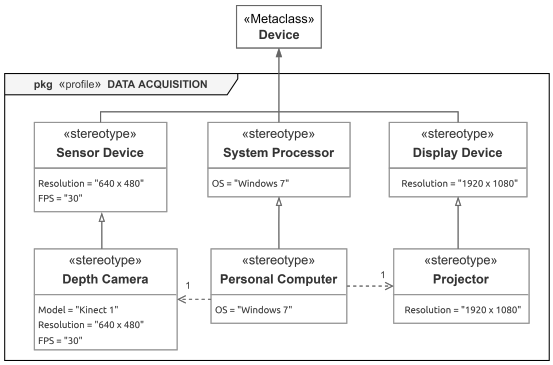

(a) Hardware model.



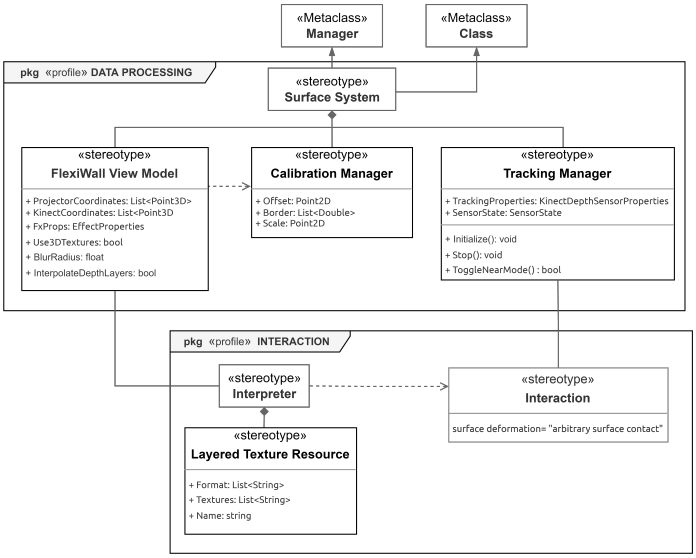

(b) Software model.

Fig. 19. Modeling the projector-camera system prototype [168].

## B.8 Back-projected interactive wall



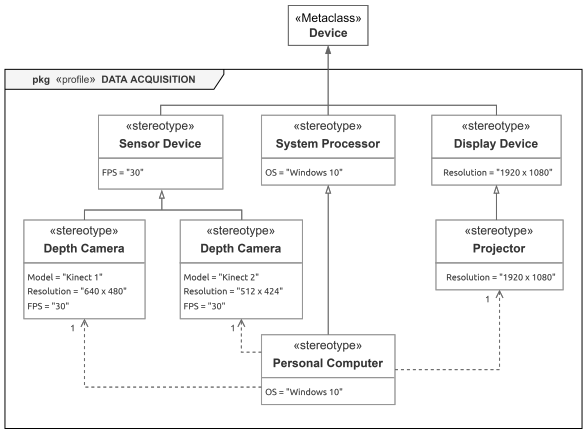

(a) Hardware model.



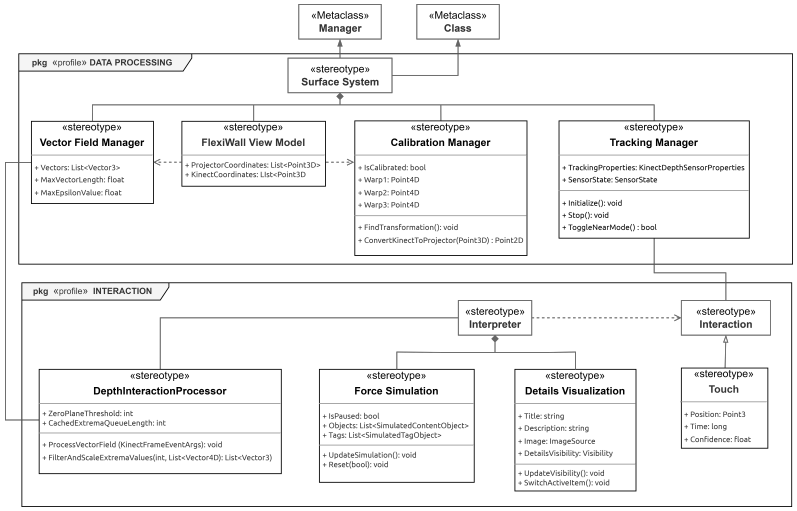

(b) Software model.

Fig. 20. Modeling the projector-camera system prototype [68].

## B.9 Back-projected interactive wall



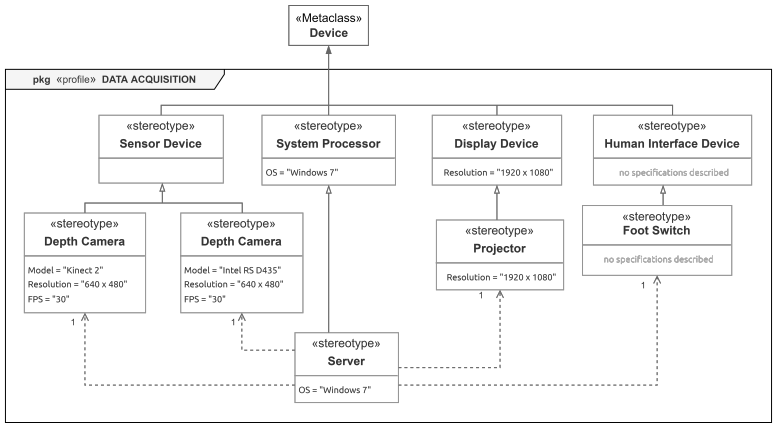

(a) Hardware model.



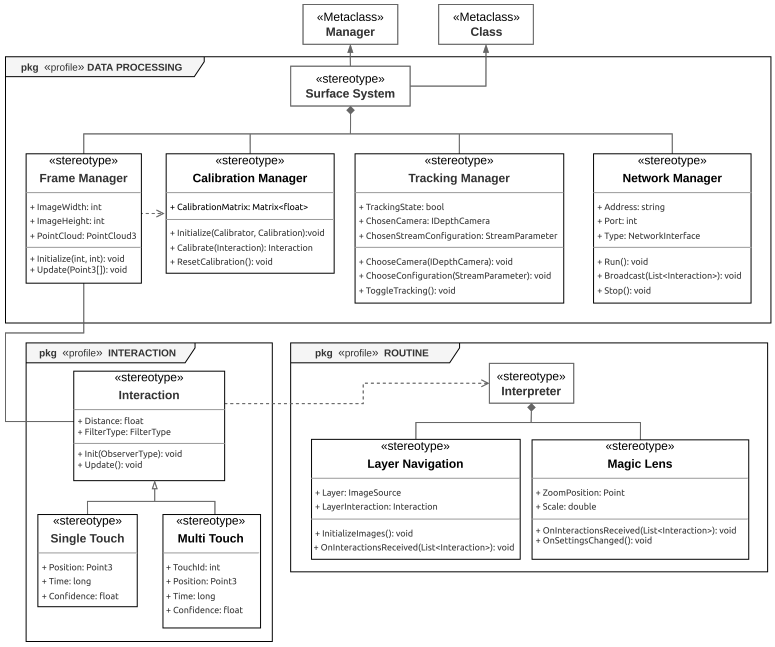

(b) Software model.

Fig. 21. Modeling the projector-camera system prototype [150].

## B.10 Back-projected interactive wall



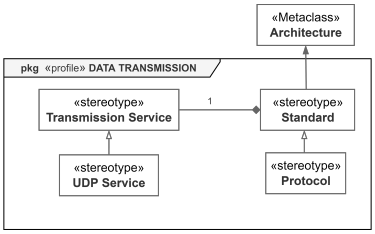

(a) Hardware model.




(b) Software model.

Fig. 22. Modeling the projector-camera system prototype [153].

### B.11 Back-projected interactive wall




(a) Middleware model.

Fig. 23. Modeling the projector-camera system prototype [153].

## 2.3 CRITICAL REMARKS

A general observation based on discussions with researchers in academia is that model-driven methodologies are typically considered needless for research prototypes. One speculation for this perspective is short-sighted vision on the life-cycle of research artifacts in a broader context, i.e., outside the scope of academic contribution. Discussions with experts have also indicated an awareness of the far-reaching impact of research artifacts, this in contrast to the replication crisis. Historically, when model-driven methods initially emerged in the 1980s, the research space of HCI was still in its infancy. As such, model-driven methodologies may not have been considered significant at that point. However, we are witnessing an unprecedented accumulation of embedded solutions [121]. Outside the research community, the open-source model has been adopted on a global scale to keep up with fast-emerging hardware. Besides the novel contributions, without strategies for diffusing emerging prototypical assets, we cannot bridge the gap between originators of concepts and future developers.

In the larger context of addressing the replication crisis, the adoption of UML for model-driven development has already gained traction in neighboring research spaces. In [182], Salber et al. introduced the context toolkit: a UML-based modeling framework that

- encapsulates of appearance and low-level implementations of physical devices,
- promotes technology-agnostic modeling,
- abstracts domain-specific components for flexibility, and
- enables reusable modeling constructs.

Similarly, in [10], Bardram introduced a UML-based API with a minimal set of core interfaces and classes to support modeling implementations. The impact of [182] and [10] forecast the role of model-driven approaches and their potential in enabling developers to diffuse research concepts with a high level of detail, thus promoting replicability.

## 2.4 SUMMARY

A study exploring model-driven methodology for replicating research prototypes in the space of interactive surface environments, has been presented. The concept of "models as end products" [184], has been applied for the development of interactive surface prototypes. As a contribution, the study has put forward a framework that targets a shared approach to documenting, communicating, and diffusing prototypical concepts. The contribution also provides for an approach to lending the benefits of model-driven approaches to developers in the early design stages.

*"The best way to predict the future is to invent it." — Alan Kay*

# 3

# Fast 3D point-cloud segmentation for interactive surfaces

## 3.1 CONTRIBUTORS


Co-author: Christopher Getschmann
email: cget@cs.aau.dk
affiliation: Aalborg University, Aalborg, Denmark
Contribution: Manuscript review and poster design

Co-author: Florian Echtler
email: floech@cs.aau.dk
affiliation: Aalborg University, Aalborg, Denmark
Contribution: Critical feedback


## 3.2 RESEARCH CONTEXT

Emerging depth cameras have propelled the implementation of numerous interactive surface prototypes [124, 140]. However, advances in depth-camera technology often imply changes in sampling rates, process speeds, capture modes, image formats, and communication interfaces. Consequently, despite the numerable benefits from emergent depth cameras, constantly changing specifications antiquate software solutions in the research community. To mitigate this setback, developers need to adopt strategies that support efficiently bridging gaps introduced by changes in depth-camera specifications. One tried-and-true approach is the open-source model [38, 117, 145]. Adopting the open-source model for software solutions in the space of interactive surface environments would promote

- community-based development and consolidation,
- community-based triage of outdated software,

- community-based scaling,
- open support.

While these benefits are well-known [6], studies indicate that the open-source model has not been adopted in the research community [51, 208].

The study presented in this chapter positions itself to promote the open-source model by addressing the second subproblem, i.e., what open-source solution can support developers to exploit non-trivial 3D point-cloud operations for spatial augmented reality? [4] In line with first study, which has underlined how the replication crisis as a direct consequence of researchers not disclosing implementation details, the second study has focused on two aspects: First, given the pervasion of commodity depth cameras in the research community, the study aims to equip developers with point-cloud-based scene understanding, i.e., how actors and objects characterize dynamic surface environments using 3D point clouds. Second, it targets open-sourcing the software solution on GitHub, a common and freely accessible open-source platform. [5] While the study places emphasis on realizing a flexible 3D point-cloud processing solution for variable depth cameras, it also focuses on lending the benefits of the open-source model to developers in the space of interactive surface prototypes.

# Manuscript 2

## 3.3 ABSTRACT

Fig. 24. Segmenting candidate interaction regions on tabletop environments.

Easy access to depth sensors has promoted exploring how point clouds can be leveraged to augment tabletops in the everyday context. However, point-cloud operations are computationally expensive and challenging to perform in real-time, notably absent dedicated GPU-compute resources. In this paper, we propose mitigating the high computational costs associated with processing 3D point clouds from commodity sensors by segmenting candidate-interaction regions near real-time. We put forward a CPU-based strategy that reduces the computational space of point-cloud representations and proposes candidate interaction regions for interactive tabletop implementations. We

[4]The research presented in this chapter has been published with published with the ACM International Conference on Interactive Surfaces and Spaces [149]. The work can also be freely accessed on the Open Science Framework [5].
[5]https://github.com/edisonslightbulbs/3DINTACToolkit

emphasize a generic solution apt for variable commodity depth cameras and modular implementation for future development. For validation, we employ a popular depth camera and apply the pipeline over a unique scene to establish initial performance measures. For the unique scene, our initial findings indicate that point-cloud data volumes are reduced by up to 70%, segmenting candidate-interaction regions under 35 ms. Going one step further, we condense the approach into an open-source solution and conclude the paper by elaborating on the benefits of the segmenting candidate-interaction regions near real-time.

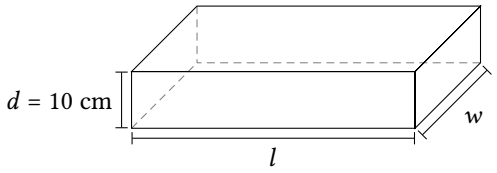
Additional Key Words and Phrases: depth cameras, 3D point-cloud segmentation, interactive surfaces

## 3.4 INTRODUCTION

Commodity depth cameras have promoted using projector-camera systems to augment un-instrumented surfaces such as tabletops [55], floors [226], and walls [80]. One prominent strategy for augmenting un-instrumented surfaces is combining image processing and depth perception-based techniques. This strategy can be seen in prevailing research implementations, e.g., Light-space [220], 3D hand gesture interaction [45], Ubeam [81], gesture recognition [35], Magic-paper [227], and adaptive augmented reality [165]. While 2D techniques enable detecting interactions, solely relying on 2D methods limits the solution space of interactive surface applications. One such limitation is the inability to perceive depth between two objects on coplanar surfaces. A promising alternative to address this shortcoming is to employ 3D point-cloud representations of scenes and augmenting 2D techniques with 3D information. However, processing 3D point clouds is non-trivial and computationally expensive, particularly with limited hardware resources. Wilson and Benko suggested constraining computations to an *interaction volume*, i.e., a 3D region 10 *cm* above a target tabletop surface [220]. Similarly, Kaltenbrunner and Echtler suggested defining a *fish tank*; a rectangular 3D volume extending above the table surface. [110].

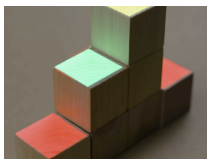
Given a 3D point-cloud representation of a tabletop's environment, segmenting the interaction volume can also be framed as a necessary preprocessing step that mitigates high computational space for downstream tasks. Existing model fitting techniques for segmenting the interaction volume favor primitive shapes, such as planes, cylinders, and spheres. However, the shape of the target surface is unknown ahead of time. Therefore, there is no guarantee that a model-fitting approach would generalize well for fitting arbitrary volumes. While there is a wealth of well-understood algorithms, prevailing segmentation techniques do not target interactive surface implementations. To our knowledge, no study has explored an open solution for segmenting candidate-interaction regions as a preprocessing solution, which we consider essential to allow for near real-time applications using limited hardware resources.

In this paper, we propose mitigating high computational costs on CPU architectures by segmenting candidate-interaction regions near real-time. We put forward a pipeline

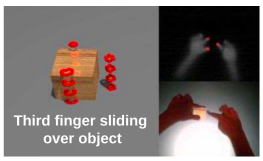



(a) The interaction volume on a tabletop surface [220].

(b) Interactive projection [36].



(c) Augmentation [227].




(d) Interaction [218].

Fig. 25. Using projector-camera systems to activate engaging interactions on un-instrumented tabletop surfaces [36, 218, 227]. Depth perception enables interpreting how actors and objects characterize a tabletop's interaction volume, an essential step towards augmenting tabletops with engaging mixed-reality.

for reducing compute-intense point-clouds from depth cameras. For validation, we employ a popular depth camera over a unique scene to establish initial performance measures. For the unique scene, our initial findings indicate that point-cloud data volumes are reduced by up to 70%, segmenting candidate-interaction regions in under 35 ms. To streamline the proposed approach and promote community-based development, we supplement an open-source project and conclude the paper by elaborating on the benefits of the segmenting candidate-interaction regions near real-time.

## 3.5 RELATED WORK

Given a 3D point-cloud representation of a tabletop environment, our review of related work considers two standpoints:

i. fundamental segmentation techniques, i.e., for the task of segmenting the interaction volume [110, 220] and
ii. speacialized segmentation strategies for segmeting objects on tabletop surfaces.

### 3.5.1 Segmenting the interaction volume

Segmenting the interaction volume can be framed as the task of dynamically inferring and fitting the interaction volume over a tabletop surface. The end goal is to define a homogeneous region of interest, i.e., reduced computational space over which downstream processes can be constrained. We review model fitting, region growing, and edge detection for this task.

*Model fitting.* Model fitting considers the mathematical properties of primitive shapes for detecting and clustering regions of interest. Schnabel et al. demonstrated the computational efficiency of optimized RANSAC [64] algorithms. The authors suggested

using an octree to establish spatial proximity between samples, while maintaining the generality and simplicity of the original algorithm [187]. Fayez et al. comparatively analyzed the performance between extended RANSAC algorithms and the Hough transform [8]. The authors highlighted the reliability of RANSAC over Hough transforms, presenting performance measures over different point-cloud sets [61]. Rusu et al. proposed statistical analysis and histogram feature estimation for re-sampling and fitting geometric shapes in unorganized point clouds [181]. In [95], Hettiarachchi and Wigdor proposed computing 2D convex hulls to create polygonal prisms, adopting Euclidean cluster extraction for segmentation [180].

*Region growing.* Region growing considers two key aspects: selecting a seed point and specifying criteria for determining that seed point's neighbourhood [15]. Region growing is an iterative process that is recursed until a base condition is met. [Rabbani et al.] suggested employing a smoothness constraint, surface normals, and neighborhood size [170]. Lingni Ma et al. proposed determining the lowest curvature point as an initial seed point and growing nearest neighborhoods based on a predefined angular threshold, i.e., comparing neighboring point normals to a modeled plane normal [130]. Oesau et al. proposed defining two conditions for seed selection: neighborhood planarity and minimal distance to centroid cells. The authors suggested prioritizing neighborhood planarity to achieve efficient growing for shapes with planar characteristics [160].

*Edge detection.* Given a point-cloud representation of multiple objects, edge detection aims to define object boundaries using maximum discriminatory information. Early approaches proposed gradient analysis, line fitting, and analysis of sharp changes in unit normal vectors [17]. In [211], Wang et al. highlighted two common approaches: polygonal-based methods [30, 213] and point based methods [82, 166]. Hubeli and Gross proposed identifying edges using normal classification, isolating and refining the edges over a thinning process [102]. Hildebrandt et al. explored anisotropic filtering, using third-order derivatives of surfaces to establish discrete differential geometric properties [96]. In [212], Watanabe and Belyaev also approximated Mean and Gaussian curvatures using discrete differential geometric operators. Pauly et al. proposed leveraging target points as features and using covariance analysis of local neighborhoods. The authors put forward a multi-resolution framework optimized for edge detection in noisy point clouds [166]. Gumhold et al. also explored covariance analysis for noisy point clouds. the authors proposed using scores to classify a point as either a crease, bounding edge, or corner [82]. In [209], Wang et al. proposed using height differences based on structural information within the point cloud.

### 3.5.2 Segmenting objects on tabletop surfaces

Our review considers [123] and [198]: two studies that have explored segmenting objects in tabletop environments. Lai et al. put forward an approach to labeling tabletop surface objects using sliding window detectors. In [123] the authors underlined how solely relying on point cloud labeling does not fully exploit RGBD data and proposed

employing RGBD views to project class probabilities onto reconstructed scenes based on voxel representation. As a result, Lai et al. demonstrated accurate labeling of objects and contributed towards more robust image object detection. Trevor et al. put forward an effective approach to semantic segmentation of organized 3D point clouds near real-time. The authors proposed a three-step process, namely, normal estimation, planar segmentation, and euclidean clustering [198]. Trevor et al. demonstrated fast semantic segmentation of contiguous eucleadian clusters.

### 3.5.3 Discussion

The reviewed literature suggests that model fitting techniques favor primitive shapes such as planes, cylinders, and spheres (i.e., given point cloud representations that correspond to mathematical models in real space). However, as discussed by Kaltenbrunner and Echtler and Wilson and Benko [110, 220], the interaction volume of a tabletop surface is not tangible. In theory, one could adapt model fitting techniques for defining the non-physical interaction volume. Although feasible, it diminishes the guarantee of optimal runtime efficiency. Also, the shape of the target surface is unknown ahead of time. Therefore, there is no guarantee that a model-fitting approach would generalize well for fitting arbitrary volumes without additional strategies. In Section 5, we elaborate on this pitfall by looking at an existing toolkit that employs a RANSAC-based strategy for segmenting the interaction volume. Approaches presented in [130, 160, 170] suggest region growing to be a promising approach for optimal runtime efficiency. Also, the approach put forward in [181] motivates exploring statistical analysis. As a general note, although robust, all techniques identified in the body of literature incur significantly high computational costs [211]; moreover, they do not target interactive surface applications. In order to achieve real-time interactive surface applications (e.g., gesture [35, 220], touch [219], and projected augmented reality [221] applications) the high computational costs must be mitigated.

Although Lai et al. demonstrated accurate semantic segmentation of 3D point cloud representations of tabletop surfaces, their approach focused on robust image object detection, omitting computational optimization. Interactive tabletop systems must consider computational efficiency essential for real-time applications. The real-time component is critical to achieving fluid user experience, i.e., applications with minimal lag (i.e., a relatively long time between user input, e.g., tabletop touch and display response). The voxelization and sliding window inference strategy would require significant computational optimization in order to adapt it for segmentating candidate-interaction regions in real-time, particularly on CPU architectures. On the other hand, Trevor et al. put forward an optimized strategy to fast semantic segmentation of tabletop surfaces. Towards preprocessing 3D point clouds for interactive applications, employing euclidean clustering for segmenting contiguous spatial densities [198] is a sound approach. However, some aspects of their strategy would need to be reconsidered to tailor the approach for interactive surface applications. The first, outlier removal: 3D point clouds from commodity sensors are typically noisy, sparse, and

at times, incoherent. A first critical step that would need to be addressed is outlier removal to ensure the quality of the point cloud data. The second, persistent normal estimation: Interactive surface implementations typically consider a fixed overhead project-camera unit. As such, continuous normal estimation would not be necessary as both the camera's viewport and the tabletop surface would be fixed, i.e., the tabletop's plane need only be evaluated once. The third, optimizing processing time: Trevor et al. achieved semantic segmentation in 80ms [198]. While fast, as a preprocessing filter for downstream tasks, 80ms would propagate significant delays for interactive applications. As a preprocessing step, the time for semantic segmentation would need to be reduced. Furthermore, while Trevor et al. have contributed a practicable approach, to our knowledge, their software implementation has not been made open to the research community and, as such, cannot be readily exploited as a building block for interactive surface applications.

## 3.6 OUR APPROACH

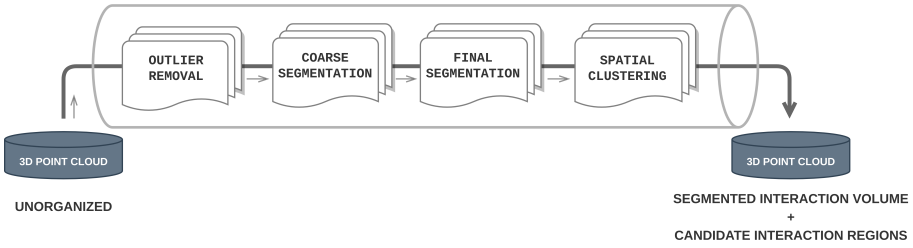



Fig. 26. Pipeline operations for segmenting interaction regions. Pipeline filters include an outlier removal filter, a coarse segmentation filter, a final segmentation filter, and a clustering filter.

We propose a pipeline with four filters (see Fig. 26): an outlier removal filter, a coarse segmentation filter, a final segmentation filter, and a clustering filter. Assumptions that need to be considered for these pipeline operations to be valid are as follows: (a) the depth sensor is mounted directly above a target tabletop surface; (b) the sensing direction is approximately perpendicular to the target tabletop; (c) the line-of-sight to the target tabletop is unoccluded; and (d) the sensing range, i.e., the distance between the depth sensor and the target tabletop surface, is within the boundary of operational limitations specified by a vendor. In the following subsections, we outline the implementation of each pipeline filter.

### 3.6.1 Segmenting interaction volume

*Outlier removal.* We denote an unorganized 3D point cloud using $\boldsymbol{p} = \{p_1, p_2, \ldots, p_m\},\ p_i = (x_i, y_i, z_i) \in \mathbb{R}^3$ and the corresponding centroid using $\overline{p} = (\overline{x}, \overline{y}, \overline{z}) \in \mathbb{R}^3$. We denote the set of Euclidean distance measures from the centroid $\overline{p}$ to all points $\{p_1, p_2, \ldots, p_m\}$ using $\boldsymbol{d} = \{d_1, d_2, \ldots, d_m\},\ d_i = d_i(p_i, \overline{p}) \in \mathbb{R}^1$ where,

$$d_i(p_i, \overline{p}) = \sqrt{(\overline{p} - p_i)^2}. \tag{1}$$

Linear discriminant analysis [234] and statistical analysis are used for removing outliers. Given a raw 3D point cloud, within-point variance is evaluated, and *"distance-outliers"* [115] are identified and filtered out in a 4-step process:

1. The centroid $\overline{p}$ is evaluated.
2. Distance measures $d_i = (p_i, \overline{p})$ are established.
3. Points $\{p_1, p_2, \ldots, p_m\}$ are sorted based on ascending order of distance measures $\{d_1, d_2, \ldots, d_m\}$.
4. Lastly, outliers are filtered out using the interquartile range test for normality of distribution:

$$\{d_i : d_i < \overline{d} + 1.5\sigma\} \tag{2}$$

where $\overline{d}$ is the mean and $\sigma$ is the standard deviation. Algorithm 1 presents the corresponding pseudocode.

*Coarse segmentation.* Given a set of denoised points $\boldsymbol{p'}$, $\overline{p'}$ is used to denote the centroid. The set of points $\boldsymbol{p'}$ are translated into distance vectors $\boldsymbol{r}$ such that $\boldsymbol{r} = \{r_1, r_2, \ldots, r_m\},\ r_i \mapsto p'_i - \overline{p'} \in \mathbb{R}^3$. Points that exist on a tabletop's surface are denoted using $\boldsymbol{t}$ such that $\boldsymbol{t} = \{t_1, t_2, \ldots, t_m\},\ t_i = (x_i, y_i, z_i) \in \mathbb{R}^3$. The set of points $\boldsymbol{q} = \{q_1, q_2, \ldots, q_m\},\ q_i = (x_i, y_i, z_i) \in \mathbb{R}^3$ is used to denote a coarse segment such that $\boldsymbol{t} \subseteq \boldsymbol{q}$ and $\boldsymbol{q} \subseteq \boldsymbol{p'}$. Given Assumptions 1 and 2, we exploit the depth sensor's view perspective: knowledge that the centroid $\overline{p'}$ exists in the subset of points $\boldsymbol{q}$. Coarse segmentation therefore considers determining the face normal of the tabletop and growing a region of interest that corresponds to the interaction volume. In order to determine the face normal of the tabletop, we frame the task as a least squares problem and employ the Eckart-Young-Mirsky matrix approximation theorem [75], computing the singular value decomposition over $\boldsymbol{p'}$ (Eq. 3).

$$A = U\Sigma V^T \tag{3}$$

where

$$U = (u_1, \ldots, u_m), \qquad V = (v_1, \ldots, v_3), \qquad \Sigma = diag(\sigma_1, \ldots, \sigma_3).$$

Given the standard form of the equation of a plane Eq. 4 and the normal vector of each point $p'_i$ as expressed in Eq. 5

$$Ax + By + Cz = D, \qquad \vec{N} = \langle A, B, C \rangle \tag{4, 5}$$

the left singular vectors ($U$) corresponds to the $x, y, z$ column vectors, where $m$ is used to denote the number 3D points. The right singular vectors ($V$) corresponds to 3D-row vectors, and $\Sigma$ corresponds to the diagonal singular values linked to the left and right singular vectors. The minimized error in determining norm (i.e., the Frobenius norm $n$) of the tabletop's surface is given by $\sigma_3$ which is associated with $v_3$. The abstract process for region growing follows:

1. Points $\boldsymbol{p'}$ are translated into direction vectors $\boldsymbol{r}$ using the centroid $\overline{p'}$, i.e., $r_i \mapsto p'_i - \overline{p'}$

2. Singular value decomposition.
3. Growing a coarse segment using Eq. 6.

$$\boldsymbol{E} = argmin \sum_{i=1}^{m} n \cdot r_i \tag{6}$$

where $\boldsymbol{E}$ denotes the region growing constraint. Algorithm 2 in Appendix A provides the pseudocode for coarse segmentation.

*Final segmentation.* Region growing implemented during coarse segmentation yields the desired segment height above the tabletop surface. However, the area of segment is overfitted. For final segmentation, we detect the tabletop's boundary, i.e., confine the area of the coarse segment to that of the tabletop's area, thereby defining the desired interaction volume. Our approach to detecting the boundary of the tabletop's surface area is two-fold: Given Eq. 4 and Eq. 5, we first we employ Silhouette edge detection (Eq. 7). Then, as a final cleaning step, we assess the discontinuity in depth measures.

$$Ax + By + Cz + D = \begin{cases} > 0 & \text{front facing} \\ = 0 & \text{parallel} \\ < 0 & \text{back facing} \end{cases} \tag{7}$$

The normal vector components utilized for Silhouette edge detection (i.e., $A, B, C$) correspond to those computed during coarse segmentation using Eq. 3. To detect the tabletop boundary, we leverage the knowledge that face normals of the tabletop's edge are perpendicular to the sensor's viewing direction. Therefore, face normals that correspond to the boundary of the tabletop are characterized by a vanishing depth component, i.e., a vanishing $z$ normal value. Fig. 27 depicts how discontinuity in depth measures is used to determine the tabletop's boundary. Once more, given Assumptions 1 and 2, we leverage knowledge that points in proximity to the centroid are a known subset of $\boldsymbol{t}$. We re-frame analysis of discontinuity as the assessment of the rate of change in depth measures. To this end, we heap and sort (in descending order) the depth measures, i.e., for the set of points $\boldsymbol{q}$. We then compute the maximum second derivative, which coincides with the maximum *"curvature elbow"* (see Fig. 27). Algorithm 3 in Appendix A provides the pseudocode for final segmentation.

### 3.6.2 Segmenting candidate interaction-regions

Given the set of points corresponding to interaction volume, the point cloud segment is clustered into candidate interaction regions. For clustering, we adopt DBSCAN [59], optimizing the original algorithm proposed by Ester et al. using nanoflann [18]. The nanoflann optimization affords a worst-case computational complexity of $O(n^2)$. Algorithm 4 in Appendix A presents the abstract algorithm for clustering using DBSCAN. Implementation of the DBSCAN algorithm first requires the approximation of two hyperparameters: $k$ and $\varepsilon$. We set $k = 4$ as suggested by Ester et al.[59], and estimate the $\varepsilon$ neighborhood size using steps $1 - 6$ in Algorithm 4. We note: estimating the $\varepsilon$ hyperparameter using the K-NN algorithm need only be done once as part of the initial

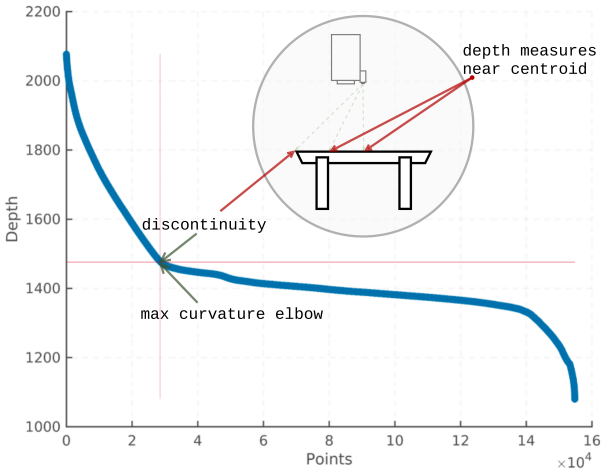



Fig. 27. Edge detection using discontinuity in depth measures. Discontinuity in depth measures from the region near the centroid is used for the final cleaning step.

setup for the depth sensor. If the sensor is changed, or should the initial position of the sensor be displaced significantly, a recalculation of the $\varepsilon$ hyperparameter is required. We also note that the time taken to approximate $\varepsilon$ varies significantly based on the number of points provided by the depth sensor. (a good proxy to $\varepsilon$ hyperparameter suggested by Ester et al.), i.e., the maximum second derivative of the angular differences between the sorted $4^{th}$ nearest neighbors of all points.

### 3.7 INITIAL PERFORMANCE ANALYSIS

A tabletop with walls and protruding structures in its vicinity was employed for validation. The Kinect was mounted directly above the tabletop surface, 1.42 $m$ above the tabletop, with a view direction approximately perpendicular to the target surface. The Kinect itself was configured with a color resolution of 720p and a 2x2 binned depth resolution (i.e., 512x512). [1] A desktop computer with 16G RAM and an AMD Ryzen 7 3700X 4GHz CPU was also employed. On average, segmenting the interaction volume took under 16 $ms$, with an average data reduction of 76.67%. Clustering candidate interaction regions within the segmented interaction volume took under ~19.5 $ms$. Accordingly, graphed performance measures indicate that the runtime for both processes increases steadily with an increase in point cloud size (Fig. 28—a). As expected, contrasting runtimes of clustering entire unsegmented point clouds representations against clustering segmented interaction volumes (Fig. 28—b) indicates a 44.28% reduction in runtime penalty in favor of the segmented point cloud. It is also necessary to note that resulting spatial density clusters from the unsegmented point cloud would still require additional computational strategies to determine the clusters that correspond to the actual interaction volume.

[1] https://docs.microsoft.com/en-us/azure/kinect-dk/hardware-specificationdepth-camera-supported-operating-modes

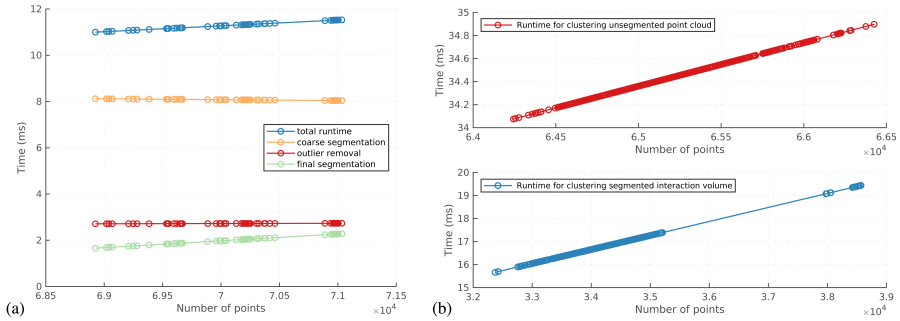



Fig. 28. Initial performance report: Runtime performance over 1000 run instances. In (a), outlier removal took under 6 *ms*, coarse segmentation took under 8 *ms*, and final segmentation took under 4 *ms*. In (b), clustering interaction regions without segmenting the interaction volume first took between 34 *ms* and 35 *ms*. Clustering interaction regions after segmenting the interaction volume took between 15.7 *ms* and 19.5 *ms*.

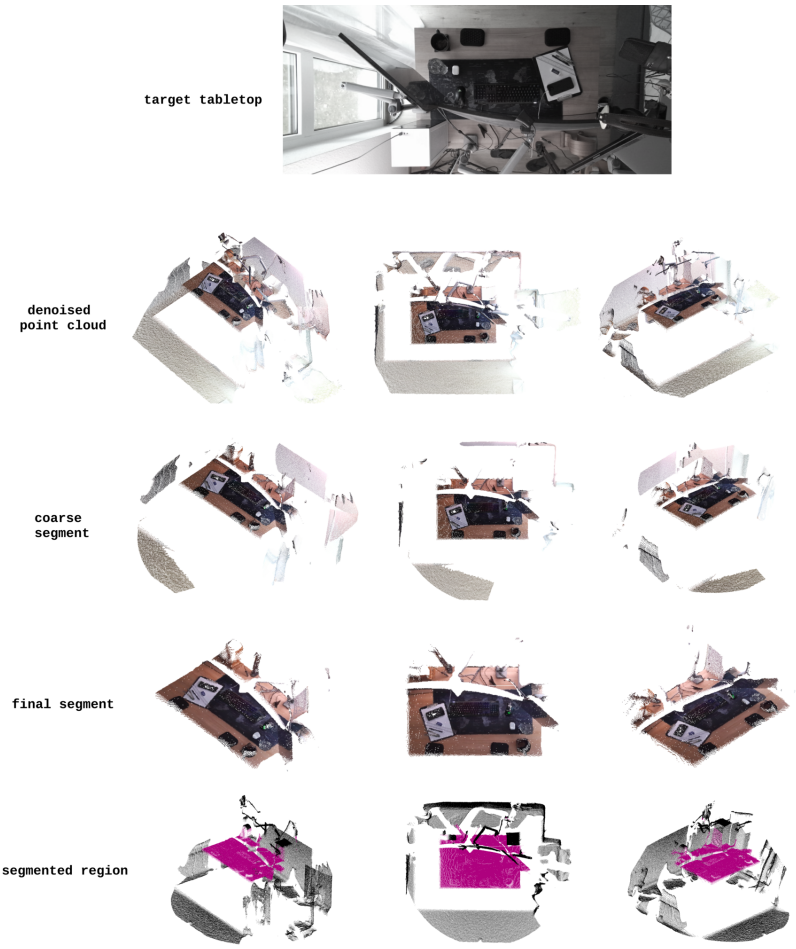
## 3.8 APPLICATION AND BENEFITS

To streamline exploring our approach, we also put forward 3DINTACT: an open-source project for segmenting interaction regions on tabletop surfaces near real-timeThe toolkit abstracts the proposed pipeline operations into small modular libraries that developers can modify, adapt, and extend flexibly. The open-source project elaborates using ready-made solutions for applications, including finding vacant surface space for interactive projection, real-time rendering, and object detection.

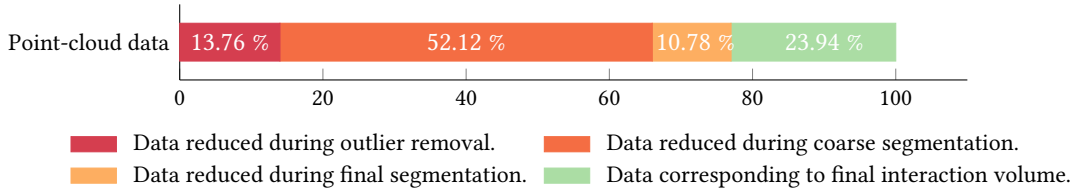
### 3.8.1 A robust preprocessing filter

To show how our approach can be leveraged for processing in existing applications, we form a concrete case using the SurfaceStreams toolkit [50].

SurfaceStreams is a toolkit that enables multiple display-camera systems to record and share digital interactive media. A critical step that the toolkit considers is segmenting the interaction volume. For this task, it employs a RANSAC-based model-fitting strategy. The resulting computational cost and high runtime (~295 *ms*) obligate SurfaceStreams to run the algorithm only once at startup. This approach leaves the problem of incidental target surface displacement unaddressed. In place of the model-fitting strategy, the toolkit could employ 3DINTACT to continuously segment the interaction volume correctly. For the given tabletop environment (Fig. 30), benchmarking our approach against the RANSAC-based strategy yielded results that indicate our approach to be faster by up to 94.89%; a promising initial result. The significance of segmenting the interaction volume near real-time is addressing the previously mentioned problem while also mitigating subsequent processing costs currently necessary to determine the interaction area [50]. 3DINTACT also simplifies real-time rendering processed 3D point clouds which could be useful for applications such as telepresence and spatial augmented reality.




(a) Point cloud visualization for each pipeline operation over a home-office test scenario.




(b) Average data reduction during outlier removal, coarse segmentation, and final segmentation for the test scenario in (a).

Fig. 29. Preprocessing point-cloud data to minimize computational costs for downstream tasks.

## 3.9 DISCUSSION

While the contribution we put forward targets a general solution for variable depth cameras, initial validation has been limited to using the Kinect over a unique tabletop environment. A well-rounded generalization of runtime and data reduction requires extending validation to variable depth sensors and tabletop environments. Although the computational complexity for each pipeline operation is outlined, the measured

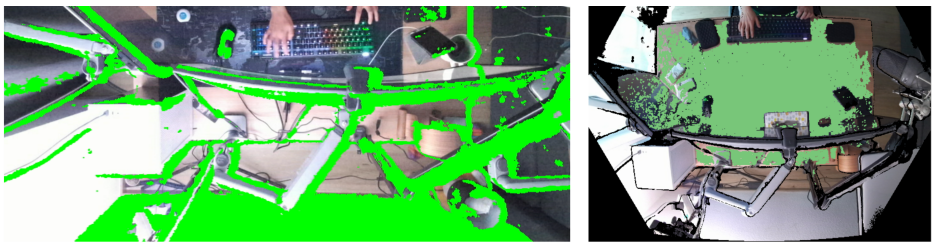



Fig. 30. Addressing challenges using our approach. In addition to RANSAC's high runtime penalty, fitting the dominant plane can yield undesired results in cases where the dominant surface does not correspond to the target tabletop surface as illustrated in (a), where the detected plane is the wall next to the tabletop. In (b), our approach correctly segments the desired interactive region on the tabletop. Our approach relies on the camera sensor being fixed directly above the tabletop surface. In instances where there are multiple tabletops in the camera's viewport, the surface most perpendicular to the camera will alwasy be segmented.

runtimes are subject to the capabilities of the specific computer in use. Future validation considers employing variable depth sensors, tabletop environments, and computers to realize a well-rounded generalization on performance. Given that our approach trades off robustness offered by exhaustive redundant operations [158] for minimal runtime penalty, future work must aim to optimize the robustness of the proposed approach without compromising generality or incurring additional runtime penalty. The work presented in this paper is a building block for interpreting how persons and objects interact within a tabletop's environment in real-time. Our next research step aims to employ the segmented interaction regions for spatial awareness and enabling interaction between mobile devices and tabletop surfaces using dynamic surface projection.

## 3.10 CONCLUSION

Given an unorganized 3D point-cloud representation of a tabletop's environment, we have presented one possible approach to reducing high computational costs for downstream operations on CPU architectures. The preprocessing strategy we propose considers segmenting the interaction volume of a tabletop surface and clustering the segment into candidate interaction regions. Our initial experiments have indicated that the proposed approach requires under ~16 *ms* to segment the interaction volume and ~19.5 *ms* to cluster candidate interaction regions. These initial runtimes can serve to benchmark comparable approaches in the future. In addition to presenting an initial performance report, we have shown tacit benefits for an existing toolkit. Equally exciting is the prospect of leveraging 3D point-cloud representations for interactive surface applications, which we look to explore in future work.

## ACKNOWLEDGMENTS

This paper was supported by the German Federal Ministry for Education and Research (BMBF) through grant number 16SV8288 within the VIGITIA project.

## 3.11 APPENDIX

## A ALGORITHMS

**Algorithm 1:** Outlier removal.
The computational complexity of the algorithm in worst-case performance is $O(n^2)$.

**Input:** point cloud (raw point cloud)
**Output:** point cloud (denoised point cloud)

Compute $\overline{p}$.
**for** *Each* $p_i$ *in* $\{p_1, p_2, \ldots, p_m\}$ **do**
    $d_i(p_i, \overline{p}) = \sqrt{(\overline{p} - p_i)^2}$
Quicksort points in ascending order of distance.
**for** *Each* $d_i$ *in* $\{d_1, d_2, \ldots, d_m\}$ **do**
    **if** $d_i > ( \overline{d} + (1.5 \times \sigma) )$ **then**
        $\boldsymbol{p} \setminus \{p_i\}$ w.r.t. $\{d_i\}$

**Algorithm 2:** Coarse segmentation of interaction context.
The computational complexity of the algorithm in worst-case performance is $O(n^2)$.

**Input:** point cloud (denoised point cloud)
**Output:** point cloud (coarse segment)

Compute $\overline{p}$.
**for** *Each* $p'_i$ *in* $\{p'_1, p'_2, \ldots, p'_m\}$ **do**
    $r_i \mapsto {p'}_i - \overline{p}$
Singular value decomposition of $\boldsymbol{r}$.
**for** *Each* $r_i$ *in* $\{r_1, r_2, \ldots, r_m\}$ **do**
    **if** $\boldsymbol{n} \cdot r_i < \boldsymbol{E}$ **then**
        $q_i \mapsto r_i + \overline{p}$

**Algorithm 3:** Final segmentation of tabletop interaction context.
The computational complexity of the algorithm in worst-case performance is $O(n^2)$.

**Input:** point cloud (Coarse segment)
**Output:** point cloud (Final segment)

```
// Silhouette-based analysis
```

1 **for** *Each* $q_i$ *in* $\{q_1, q_2, \ldots, q_m\}$ **do**

```
    // To detect the tabletop boundary, we leverage Assumptions 1 and 2, i.e.,
       the knowledge that face normals of the tabletop's edge face away from
       the sensor's viewing direction. Therefore, face normals that correspond
       to the boundary of the tabletop are characterized by a vanishing z
       normal component.
```

2 **if** $||\{q_i\}_z|| < 0$ **then**

3 $\boldsymbol{q} \setminus \{q_i\}$

```
// Depth-based analysis
```

4 Quicksort $\boldsymbol{q}$ in descending order of depth measures.

5 Compute the maximum second derivative.

```
// Use the point that coincides with the maximum second derivative {q_m}_z to
   confine the boundary of tabletop surface area, viz, final segmentation.
```

6 **for** *Each* $q_i$ *in* $\{q_1, q_2, \ldots, q_m\}$ **do**

7 **if** $\{q_i\}_z > \{q_m\}_z$ **then**

8 $\boldsymbol{q} \setminus \{q_i\}$

9 $\boldsymbol{t} \mapsto \boldsymbol{q}$

**Algorithm 4:** Clustering candidate interaction regions.
The computational complexity of the optimized DBSCAN algorithm in worst-case performance is $O(n^2)$.

**Input:** point cloud (segmented interaction volume)
**Output:** point cloud (candidate interaction regions)

1 Do DBSCAN using the maximum second derivative as a proxy to the $\varepsilon$-neighbourhood.

```
// Estimation of DBSCAN hyperparameters (a non real-time operation that need
   only be done once)
(1) Evaluate the 4^th nearest neighbors of all points in the segmented interaction
    volume.
(2) Heap the evaluated 4^th nearest neighbors.
(3) Sort the evaluated 4^th nearest neighbors in descending order.
(4) Compute the successive angular differences of the sorted 4^th nearest
    neighbors.
(5) Compute the second derivative of the successive angular differences.
(6) Determine the maximum second derivative.
```

## 3.2 CRITICAL REMARKS

Commodity depth cameras have played a central role in the exploration of spatial augmented reality for interactive surface environments. As a result, the research community has accumulated considerable software solutions that leverage modern depth cameras for interactive applications [2, 119, 177]. While the solutions are quintessential, this study underlines two significant setbacks surrounding software contributions in the HCI's research community.

(1) Current software contributions are, by and large, highly application-specific:
While this may result from researchers targeting specific application contexts, a common underlying factor observable from the different solutions is the re-implementation of similar building blocks. Speculation for this redundancy is the lack of emphasis on the reuse of components from existing solutions. A promising approach to address this pitfall is promoting the abstraction and reuse of fundamental building blocks in research contributions.
(2) Research software solutions are not open-sourced:
The space of HCI generally does not promote transparent technical development trails [51]. Study assets such as source-code and hardware specifications are generally left un-disclosed [208]. As a consequence, the research community is unable to crowdsource the development of academic contributions. The inability to leverage community-based development hinders prolonging the lifespan of research solutions which are antiquated by fast-evolving embedded hardware. Although few research contributions have been made openly accessible to the public [88, 222], as mentioned earlier, the solutions are highly application-specific. Therefore, abstracting fundamental building blocks from such structurally complex and targeted solutions is non-trivial.

Another short-coming identified by the study is as follows. Software contributions that have been put forward in the research community for processing 3D point clouds demand high-compute capability. A dedicated high-performance GPU is assumed to be a given [41, 103]. Although GPU-based processing yields remarkable computing performance, a sound argument can be made for the continued exploration of CPU-based processing techniques. In the research space interactive surface environments, non-obtrusive integration of interactive surface systems into physical environments is desirable. Thus, it follows: mobile, small-form-factor, embedded solutions that facilitate seamless integration are highly attractive for everyday living environments. We note here, while 3dintact targets embedded devices such as the RasPi, hardware performance remains subject to sensor specifications outlined by vendors. To elaborate, Microsoft's Azure Kinect requires a non-dedicated GPU for the optimal acquisition of synchronous RGBD frames. While it is feasible to use the Azure Kinect with an embedded board such as the Beaglebone, our exploratory tests pointed to significant frame dropping, which is undesirable.

The software solution put forward in this study centers around fast filtration of 3D point clouds to minimize computational effort for downstream tasks on CPU architectures. It promotes exploring 3D point cloud processing techniques on mobile devices and embedded solutions without compute-intense hardware requirements.

## 3.3 SUMMARY

The study presented in this chapter contributes an approach to preprocessing 3D point clouds from commodity depth cameras for interactive surface applications. It has focused on fast segmentation of candidate-interaction regions to reduce the computational effort required for downstream tasks. As a core contribution, the study has put forward a rich set of libraries (see Appendix) and a toolkit[2] freely accessible on GitHub.

[2]https://github.com/edisonslightbulbs/3DINTACToolkit

*"Every once in a while, a new technology, an old problem, and a big idea turn into an innovation." — Dean Kamen*

# 4

# Integrating mobile devices into interactive surface environments

## 4.1 CONTRIBUTORS


Co-author: Florian Echtler
email: floech@cs.aau.dk
affiliation: Aalborg University, Aalborg, Denmark

Contribution: Critical review

## 4.2 RESEARCH CONTEXT

The first two studies presented in this dissertation aim to contribute towards addressing the replication crisis. While the first introduces a shared syntax that enables exchanging interactive-surface models, the second promotes using Github for open-source development. The aim of this final study is twofold: First, to apply the software contribution from the second study to elaborate on the utility of fast filtration of 3D point clouds on CPU architectures. Second, to contribute a robust strategy for exploring interaction concepts between mobile devices and interactive surfaces. The study emphasizes a transparent development trail to promote repeatability. While the scholarly work follows the template for publication, all technical implementation strategies are outlined, and all software contributions are open-sourced. The interaction context of the research is discussed in the following.

Smartphones help us communicate and organize everyday activities. But beyond enabling instant access to today's digital world, smartphones have, inarguably, evolved into personal assistants inseparable from everyday life. In the context of prevalent assistants, in the non-digital domain, tables also enable a broad spectrum of arbitrary day-to-day functions. Drawing parallels, both assistants have pervasive interaction spaces. Moreover, an intersection between the two interaction spaces can be observed.

Multiple digital interactions from mobile devices are contiguous with physical interaction on tabletop surfaces. However, despite the intersection, there is no connection between the interaction spaces. That is to say, digital interactions on smartphones do not extend to the tabletop. Conversely, the physical interactions on tabletop surfaces do not extend to the digital interaction space on smartphones.

The motivation of this study is to establish a robust strategy that enables exploring the intersection between the digital and physical interactive spaces from mobile devices and tabletop surfaces. While this research direction has been explored in the past [57, 95, 185], advent SAR techniques and state-of-the-art vision sensors motivate developing new strategies. In order to address the final subproblem: Given the open-source solution contributed in the second study, what concrete application can be developed to highlight the interaction space between tabletops and personal mobile devices? the strategy we propose builds on the open-source solution contributed in the second study. As a core contribution, we detail a concrete hardware-software concept that enables leveraging state-of-art CNN-based object detection. [3]

# Manuscript 3

## 4.3 ABSTRACT

Fig. 31. *Device-surface* interaction using Traceless.

Multiple approaches have been put forward for augmenting interactive surface environments. However, spatial awareness while incorporating ubiquitous devices as active components in computing environments remains a foremost problem. Also, advances in deep learning algorithms and sensor technology over the last six years motivate furthering established approaches. We present Traceless, a projector-camera unit that enables communication between inanimate tabletop surfaces and personal mobile devices. Traceless uses a novel combination of spatial-density clustering and CNN-based object detection to augment tabletop surfaces with spatial awareness. It also employs Bluetooth to integrate personal mobile devices into augmented tabletop environments. In this paper, we describe the implementation of Traceless and demonstrate its potential for turning mobile devices into active components in a computing environment. We conclude with observations from a pilot study and discuss current limitations and potential future extensions. Our approach highlights the benefits of combining depth perception and deep learning algorithms, contributing a contemporary method with a broad range of applications.

[3]The research presented in this chapter remains on track for publication. In parallel, a current revision of the work can has been made freely accessible on the [4].

Additional Key Words and Phrases: Interactive surface environments, augmenting tabletop surfaces, spatial awareness, personal mobile devices, spatial-density clustering, CNN-based object detection, computing environment.

## 4.4 INTRODUCTION

Interactive surface environments have re-imagined the potential for concurrent interactions between ubiquitous mobile devices and tabletops (see Fig. 32). While tabletops provide assistive platforms that simplify interacting with physical and digital media, mobile devices have evolved into personal digital assistants inseparable from everyday life. The goal of interactive surface environments is to blur the lines between ubiquitous mobile devices and day-to-day surfaces, overlaying information onto real-world objects to enhance user experience. Toward this end, variable approaches have been proposed: use of markers, [172], instrumentation of tabletops [53], and augmenting surfaces with peripheral sensors [94]. These different approaches have in turn given rise to new interaction concepts such as *"proxemic"* interactions [9, 186] and *"spatially-aware cross-device"* interactions [167, 171]. A cornerstone for interactive surface environments is integrating ubiquitous devices [77, 104, 171]. Although different approaches have been put forward [94, 185, 224], enabling spatial awareness while incorporating ubiquitous devices as active components in computing environments is still a challenge. Moreover, recent advances in vision techniques [196, 210], small-form-factor sensors [121, 142], and Bluetooth technology [19] motivate revising existing approaches to realize a contemporary approach. In this paper, we present Traceless, a projector-camera unit that

(a) PhoneTouch [185]. (b) Cross-device [186]. (c) CI [54]. (d) LuminAR [129].

Fig. 32. Interactive surface environments [54, 129, 185, 186].

enables communication between inanimate tabletop surfaces and personal mobile devices. Traceless combines spatial-density clustering and CNN-based object detection to apply a novel technique for augmenting tabletop surfaces with spatial awareness. It also employs Bluetooth to integrate mobile devices into augmented tabletop environments. Our goal is to contribute a robust new method for integrating personal digital assistants into augmented tabletop environments, thereby supporting further development of HCI concepts that center around integrating mobile devices into augmented surface environments. As such, our study outlines Traceless's development trail, shows proof-of-concept, and concludes with observations from a pilot study.

## 4.5 LITERATURE REVIEW

Traceless centers around spatial augmented reality: design for minimal obtrusiveness while augmenting un-instrumented day-to-day tabletop surfaces. The envisioned concept considers overhead mounting of a projector-camera unit with fixed rotation and translation between the camera and projector. For spatial awareness, we targeted leveraging depth perception and CNN-based object detection. Given these considerations, our literature review encompassed (a) configuring projector-camera systems for spatial augmented reality, (b) object detection using state-of-the-art deep learning techniques, (c) depth perception using new small-form-factor depth cameras, and (d) studying prevailing approaches for integrating mobile devices into interactive surface environments.

### 4.5.1 Projector-camera calibration

Geometric calibration can be classed into four primary methods: local homography, global homography, object reference, and self-calibration. Local homography methods evaluate the mapping between camera space and image space to address image misalignment [127, 147]. Global homography methods evaluate the mapping between world space and image space [49, 138, 163]. Object reference methods rely on real-world objects with known geometry [60]. Self-calibration methods involve moving the camera about a static scene [23, 139]. Given a homography, Zhang proposed applying a closed-form solution to evaluate the homogeneous equations used for determining the intrinsic and extrinsic parameters. Zhang pointed out that noise compromises the rotation matrix's correctness and suggested single value decomposition for optimization [76, 238]. In [46], Dhillon and Govindu put forward an image reconstruction pipeline. Their approach employed a calibration filter based on a Matlab calibration toolkit [24]. Fiala explored using a fiducial marker system and in [62], the authors demonstrated how the ARToolkit and ARTag [63] could be utilized to evaluate correspondences. Chen and Chien and Moreno and Taubin addressed the challenge of nonlinear distortion from off-the-shelf projectors [32, 147]. In [32], the authors put forward a self-correcting closed-loop system to support evaluating the intrinsic and extrinsic parameters. Their approach extended [237] with an iterative nonlinear corrective model. Yang et al. addressed the loss of image focus over long range projection distances in [233]. They proposed establishing point correspondences using the local geometric consensus [232]: an algorithm for matching random point patterns undergoing geometric transformations in real-time. Their approach minimized re-projection error by evaluating randomly generated dot patterns. Yang et al. implemented a projector-camera system and demonstrated optimized reduction in re-projection error, contrasting their results with results from [5]. Boroomand et al. proposed projective correction, estimating calibration parameters together with surface geometry simultaneously to address distortion of projection images consequent from uneven surfaces [22]. Their approach extended [231], substituting the one-parameter model [65, 193], with the

Brown camera distortion model [28], and using the Gold standard [87] in place of the Simpson approximation [141] for evaluating the projection error and correcting noisy point correspondences. Boroomand et al. projected a reference black and white checkerboard image onto a non-flat surface and comparatively analyzed the extracted corners from different images to determine the mean squared error between the initial reference and corrected checkerboard image. The authors demonstrated re-projection of a video to illustrate natural image re-projection, showcasing the mitigation of distortion caused by uneven surfaces[22]. Willi and Grundhöfer proposed automatically calibrating multi-projector-camera systems using light pattern sequences. In [216], they proposed a self-calibration algorithm to generate sub-pixel correspondences for intrinsic and extrinsic parameters between multiple projector-camera systems.

Shahpaski et al. also proposed a structured light-based algorithm, focusing on pattern design [14]. In [190], they used an image set of four different objects, comparatively analyzing the performance of their method against the performance of the method presented in [5]. Li et al. also presented a study that considered structured light for automatically estimating intrinsic, extrinsic, and distortion parameters [126]. Given noisy point correspondences, they proposed a multi-factor objective function for determining maximum-likelihood estimates. For validation, Li et al. demonstrated robust self-calibration, effectively projecting and texture mapping on an outdoor building wall with transient outdoor lighting conditions. Huang et al. highlighted inaccurate homography estimation from incorrect camera parameters and noise caused by imperfect calibration-target planes and proposed a graph-theory-based correspondence algorithm with color-coded structured light patterns for optimizing against noise from imperfect target calibration planes. In [101] the authors also suggested photometric compensation [79, 99, 100, 157] as an alternative for addressing inaccurate homography estimation.

### 4.5.2 Object detection

Object detection typically consists of three main stages: region selection, feature extraction, and classification. Region selection considers identifying image areas with definable characteristics using maximum discriminatory information. Feature extraction is the evaluation of image areas to determine subsets containing maximum discriminatory information. Feature extraction uses cognitive representations of observable object features [179] such as blobs, corners, and edges [20, 74]. The cognitive representations commonly used to perceive objects include SIFT [132], HOG [42], Haar-like [128], SURF [156], BRIEF [29] and ORB [34]. Once region selection and feature extraction have been achieved, the final stage in object detection is classification. Given an object in an image, classification is the process of predicting that object's class and labeling it [120, 191]. Modern approaches frame object detection as a regression problem and wrap region selection, feature extraction, and classification into a single model. This model is then used as a solution for labeling and inferring bounding boxes around detected objects.

From here on out, we use the term *detection models* to refer to various existing object detection networks. Detection models are pre-trained, end-to-end networks that are optimized for object detection. Examples of detection models include R-CNN [74], Fast R-CNN [73], Faster R-CNN [176], SPPnet [90], YOLO [173], Mask R-CNN [89], and YOLO9000 [174]. Detection models use machine learning and annotated datasets to automate manual feature extraction [20, 21]. The annotated datasets typically group images of real-world objects categorically, using unique scenes as a grouping criteria. Dataset images are taken from multiple views and annotated over color and depth information [122]. Detection models address feature engineering, thus enabling automated detection of both image features [42, 133, 164], and convolutional features [48].

### 4.5.3 Integrating mobile devices

Wilson and Sarin presented an interactive surface system (ISS) that combines image processing and Bluetooth to handshake mobile devices. The authors proposed using an ISS as a beacon and locating mobile devices utilizing peripheral mobile-device sensors [224]. Akin to [224], Mäkelä et al. also leveraged an ISS as a beacon, combining image processing with peripheral mobile device sensors to realize spatial-aware interaction [136]. Vepsäläinen et al. explored different strategies for initiating communication between an ISS and mobile devices, contrasting NFC and QR code over Wi-Fi and 3G [206]. In [172], Rädle et al. proposed leveraging co-located mobile devices to achieve spatial awareness (i.e., awareness of a mobile device relative to another). Extending [172], Rädle et al. suggested using a polarization filter to support detecting mobile devices characterized by OLED-based (organic light-emitting diode) screens. The authors proposed shining polarized light to mitigate light reflections on the OLED screens [171]. Bazo and Echtler proposed using NFC tags and optical markers on instrumented tabletop surface [11].

*Discussion.* For practical application in the everyday context, we argue that spatial awareness on tabletops must consider the geometry of all physical tabletop objects, i.e., global spatial awareness. While the approach presented in [224] has been influential [136], it does not address spatial awareness. We also put forward that an unobtrusive sensing solution is desirable for everyday environments, one that does not require instrumenting the tabletop [11] nor additional hardware [171] besides a small-form-factor RGBD camera. While detecting OLED screens using depth cameras is challenging, we elaborate on a robust approach that does not require specialized sensing hardware in the section that follows. Given that day-to-day environments are characterized by various activities, a spatial augmented reality-based approach needs to address the critical challenge of occlusion.

Our literature review on depth perception and scene understanding for tabletops has digressed into a separate study focusing on spatial awareness using 3D-point clouds. The study contributes 3DINTACT [1]: an open-source project for segmenting candidate interaction regions on tabletop surfaces near real-time. In view of the study and the

[1] https://github.com/edisonslightbulbs/3DINTACToolkit

tool contributed by the authors of this paper, we have excluded a literature review on depth perception. Literature reviewed for configuring projector-camera systems extends a classical approach put forward by Tsai. As this approach is expansive, we present a concise summary of its formulation in Appendix A. Given our consideration of fixed rotation and translation between the camera and projector, we adopt Boroomand et al.'s approach to projector-camera calibration. However, as opposed to MATLAB, we use OpenCV [162] to exploit Zhang's technique. Our review has identified Faster R-CNN [176], SPPnet [90], and Mask R-CNN [89] as prominent, accessible, and adaptable solutions. In addition, Ultralytic's YOLOv5 [203] facilitates a detection model well-adapted for real-time object detection: what we underline as an essential component for interactive surface applications. As evinced in [136, 224], Bluetooth remains a pervading technology for most mobile devices, which we also look to exploit.

## 4.6 REALIZING TRACELESS

### 4.6.1 Hardware components

We employ Microsoft's Azure Kinect to exploit its 1-MP time-of-flight depth sensor, a 12-MP RBG sensor, and synchronized RGBD streams for vision sensing and depth perception. Besides a configurable depth resolution and field-of-view, the kinect's small-form-factor supports unobtrusive hardware installation. For projection, Philip's NeoPix Ultra (NPX640/INT) is used. Lastly, we adopt a workstation computer with 16G RAM, a GeForce RTX2070S 8G GPU, and an AMD Ryzen 7 3700X 4GHz CPU for local processing. The workstation uses an ASUS X570-I, which facilitates 802.11a/b/g/n/ac and Bluetooth wireless communication peripherals.

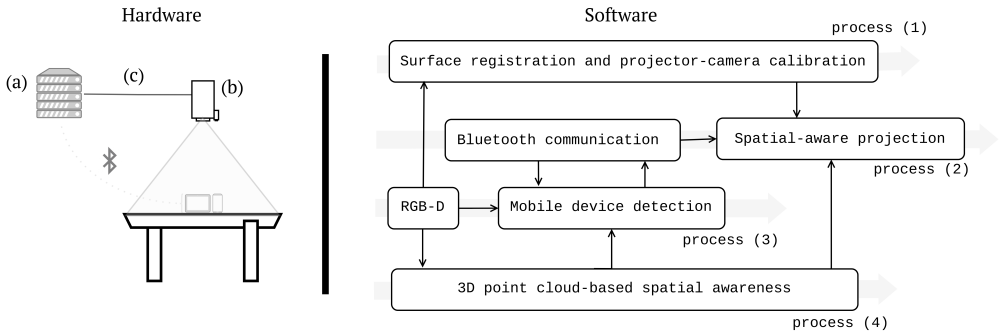



Fig. 33. Traceless comprises a workstation computer (a), projector-camera unit (b), and HDMI and USB cable connection (c). The software implementation considers four essential parallel processes to augment non-instrumented tabletops and integrate mobile devices.

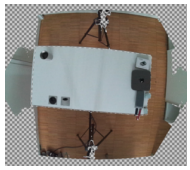
### 4.6.2 Software components

Software is implemented on a Unix platform; Ubuntu 20.04.2.0. With reference to Fig. 33, this section outlines how we implement

a. spatial awareness,
b. communication,

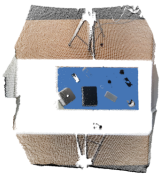
c. mobile-device detection,
d. surface registration,
e. projector-camera calibration, and
f. spatial-aware projection.

*3D point cloud-based spatial awareness.* Given synchronous RGBD streams [143], 3DINTACT segments the tabletop interaction region [110, 220] and provides candidate interaction regions (CIRs). The CIRs are then used to define *profiles*: generic geometric (3D) representations of objects of interest, e.g., mobile devices, and non-occluded tabletop space.



(a) Tabletop environment.

(b) 3D point cloud of tabletop environment.



(c) Segment of target tabletop (colored blue).

(d) 3D point-cloud segment of target tabletop.

(e) 3D spatial-density clusters of segmented tabletop surface.

(f) Profiles of tabletop objects.

Fig. 34. Using 3DINTACT to extract profiles of tabletop objects. Given a 3D point-cloud representation of a tabletop environment, 3DINTACT segments the target region of interest and does boundary segmentation under 30 *ms*. A limitation not addressed in this study is detecting mobile devices that do not rest directly on the tabletop surface. This challenge can be observed with the "vanishing" watch on the book stack: light reflections on OLED displays of devices not directly on the tabletop surface have not been addressed.

*Bluetooth communication.* To realize wireless communication between Traceless and mobile devices, we use the Bluez DBUS API to implement Bluetooth services over RFCOMM sockets. We also implement custom mobile-device software for streaming transport over the RFCOMM sockets. Given that Android APIs provide open and extensive support for data transmission using RFCOMM sockets, initial development of the mobile-device software is restricted to Android devices. Communication channels between Traceless and mobile devices are established as follows. Devices are first required to pair with the ISS. This conventional security measure needs only be done once and is necessary to inform the ISS about the existence of trusted devices. The custom mobile-device software automates the process of pairing and facilitates seamless

connection to the ISS. Given that modern Android smartphones are equipped with multiple sensors, we also employ custom mobile-device software to support contextual-device discovery. When available, accelerometer and gravity sensor data is fused and used to identify devices whose context (i.e., position and relative orientation) corresponds to devices lying on a horizontal surface; candidate mobile devices typically not in use. This feature aids our approach to correctly detecting mobile devices, which we discuss further below.

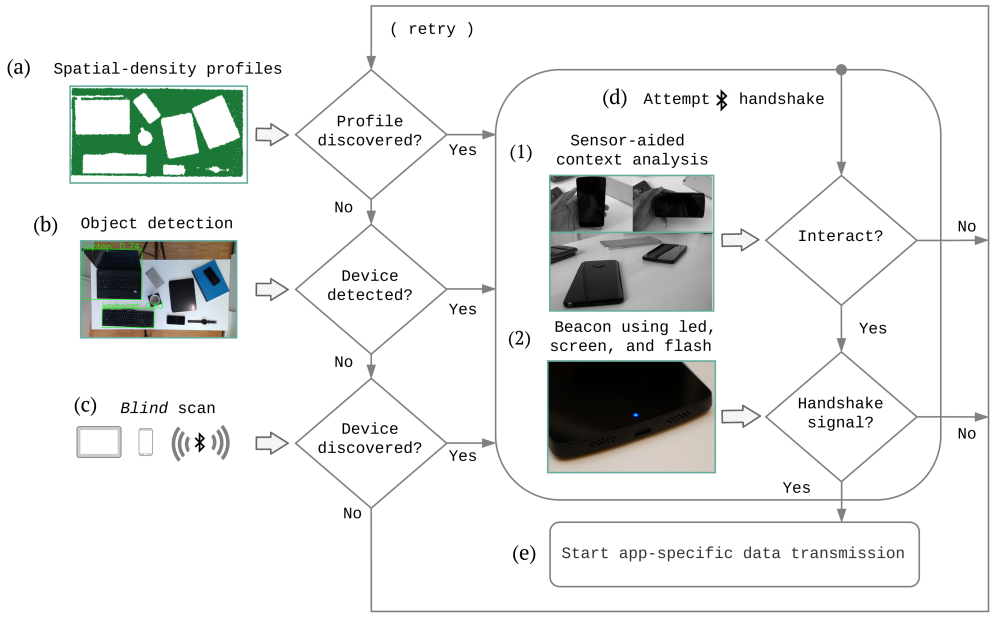



Fig. 35. Traceless employs three strategies for robust detection of mobile devices. The first strategy (a) assesses spatial-density profiles from 3D point clouds. The second strategy (b) relies on image-object detection. The third strategy (c) does a Bluetooth sweep to discover paired within-range mobile devices.

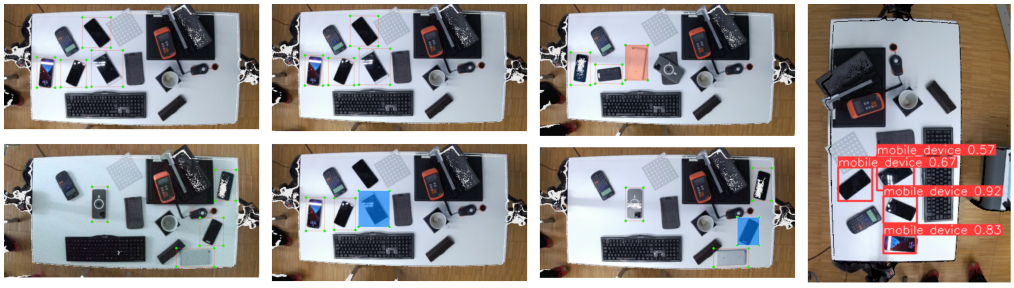



Fig. 36. Leveraging the Yolov5 detection model, transfer learning is used to train a custom detection model. 250 color-to-depth-transformed images captured in variable lighting conditions were annotated for training a detection model.

*Mobile device detection.* Fig. 35 summarizes our methodology for detecting and handshaking with mobile devices. Foreground masks from successive background subtraction are employed. If changes on the tabletop surface, as indicated by the foreground

masks, are greater than a preset threshold, a detection sequence is triggered. Once triggered, Traceless utilizes three detection schemes (see Fig 35).

In the first scheme, if a spatial-density profile corresponding to a mobile device is detected, a handshake is attempted (Fig 35.d). If the handshake is successful, the bounding edge of the mobile device is referenced from the spatial-density profile for projecting application-specific media. This approach enables Traceless to find vacant space nearest to the mobile device, thus allowing for spatial-aware projection.

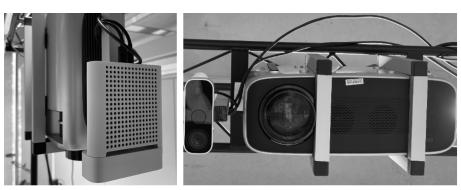
The second scheme employs a custom detection model trained based on YOLOv5 [203]. The custom model is trained using transfer learning to detect mobile devices from synchronized RGBD streams captured by the kinect. Once a mobile device is detected, the class labels, confidence, and bounding boxes from the custom detection model are used to identify the corresponding boundaries of devices on the tabletop. Akin to the first scheme, the bounding edge of the mobile device is referenced from the bounding box and used to project application-specific media intuitively.

The third scheme uses a *blind* Bluetooth scan to discover paired within-range mobile devices. We refer to the scan as blind as it is triggered if and only if vision-based sensing fails. Given a mobile device on the tabletop, for instances where vision-based detection is unsuccessful and a device is detected through a Bluetooth scan, Traceless attempts to learn the device's position using background subtraction. This fail-safe contingency attempts to identify foreground masks that correspond to the geometry of typical mobile devices. Traceless leverages sensor-aided context analysis (Fig. 35.d.1) and handshake indicators (Fig. 35.d.2) to recover a mobile device's bounding edge, which is necessary for spatial-aware projection. Upon successful detection and handshaking with mobile devices, application-specific data transmission begins.

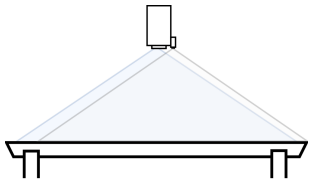
The shortcomings of each detection scheme are mitigated through redundancy. Ambient lighting limits detecting mobile devices solely using an object detection model, and thus depth perception is also employed. While Bluetooth scans and background subtraction are adequate for evaluating device position, spatial-density profiles are necessary to enable Traceless with spatial awareness. The spatial awareness, in turn, enables intuitive media projection on the tabletop surface.

*Surface registration and projector-camera calibration.* As mentioned in Section 4.5, mounting is such that rotation and translation between the camera and projector are fixed (see Fig. 37). In appendices B.1— B.2, we outline our approach to projector-camera calibration. To supplement the adopted approach, we also contribute a C++ library that abstracts the complexity of the calibration process

*Spatial-aware projection.* As mentioned earlier, Traceless uses synchronous RGBD streams to achieve spatial awareness and detect objects' boundaries in world space. Given depth perception from the Kinect and a calibrated projector-camera unit, the spatial-density profiles from synchronized RGBD point clouds are a proportionate representation (i.e., with millimeter accuracy) of the geometric characteristics of objects in world space. In other words, the spatial-density profiles from synchronized RGBD



(a) Projector-camera installation.



(b) Overlapping field of view between the projector and camera.

Fig. 37. Overhead mounting of the projector-kinect unit. Installation is such that the projector and kinect share field of view.

point clouds can be superimposed onto world space. The synchronized depth and color thus enable operations on 2D images to be mapped to 3D point clouds and vice versa. We utilize this perception to realize *"device-to-surface"* interaction: identifying vacant space near a mobile device and extending the display area of that device to the tabletop surface intuitively. Successful re-projection of the synchronized depth image implies the correct superposition of the corresponding spatial-density profiles. Fig. 38 illustrates our approach to re-projecting the synchronized depth image onto the tabletop surface.

## 4.7 INTERCEPTING DEVICE NOTIFICATIONS

A typical scenario in everyday living and working environments is the arbitrary placement of smartphones on tabletop surfaces. Setting aside a mobile device frees the user to perform other tasks. However, while engaging in other tasks on the tabletop, or non-digital assistant, the user is often obligated to disengage from their personal digital assistant momentarily. We evaluate:

- The activation of spatial awareness on a tabletop surface.
- Robust detection and handshaking with mobile devices.
- Seamless communication between mobile devices and Traceless.

For a concrete application, we integrate ad-hoc mobile devices into a tabletop environment and demonstrate using media projection and the interception of device notifications. Once notifications are intercepted, Traceless augments the tabletop surface with spatially aware media prompts. This spatial-aware projection the basis of what we refer to as device-surface interaction. The aim of device-surface interaction is to bridge the interaction gap between mobile devices and tabletop surfaces in the context of digital and non-digital assistants in the daily life. We invite users to participate in a pilot study with an active demonstration as part of our evaluation. The pilot study is leverage to collect an initial set of observations and learn end-users' perceptions towards informing the development of future applications.

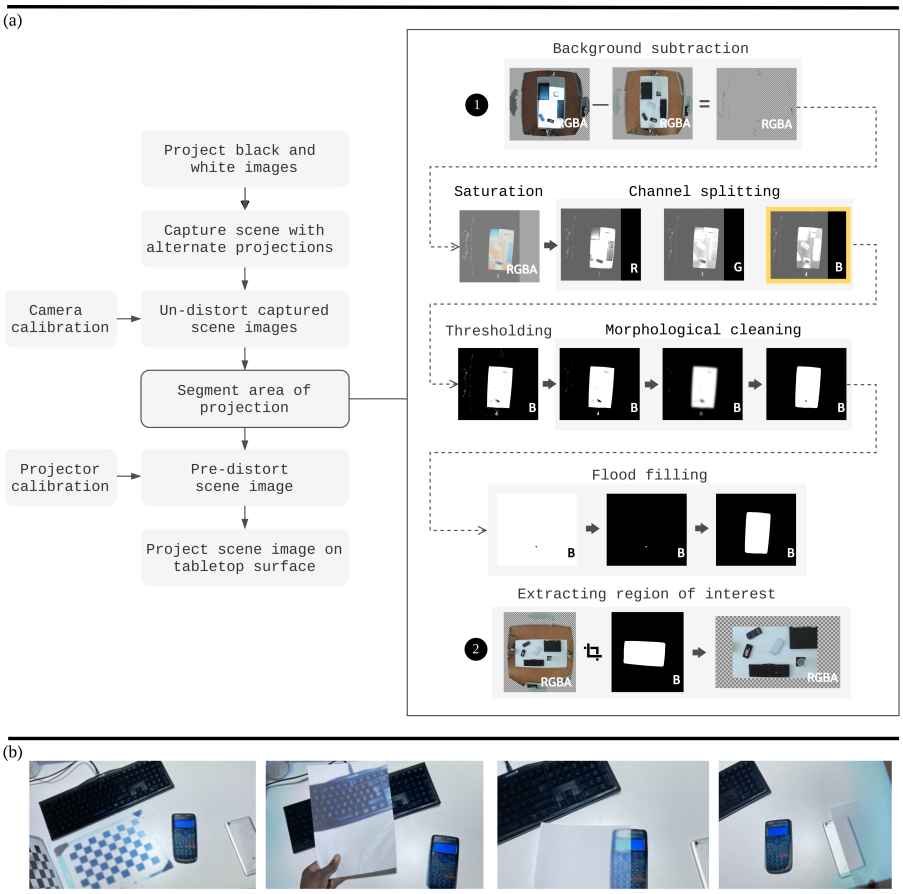



Fig. 38. Surface registration and re-projection. (a) describes how we apply projector-camera calibration to un-distort captured images and pre-distort projected images. We also elaborate on the methodology adopted to segment the projection area (a.1) and extract the region of interest (a.2). Once extracted, we reproject the region of interest as shown in (b).

### 4.7.1 Device-to-surface interaction

When a profile that corresponds to a mobile device (i.e., a cuboid shape with area less than $20000mm^2$ and breadth less than $10mm$) is identified, image-object detection is triggered. Once a device is detected, handshaking is initialized. During the handshake process, a red boundary is projected around the detected mobile device. After handshaking, the red-colored boundary turns green and blinks twice before turning off. This visual feedback indicates that a communication channel has been successfully linked between the mobile device and Traceless. Once a communication channel is established, Traceless queries the list of application services over which permission is granted for intercepting notifications. When a notification is intercepted, Traceless evaluates the location of the mobile device and the spatial orientation of the tabletop to find the nearest unoccluded surface to project media. Device-surface interaction is stopped when the user picks up the mobile device.

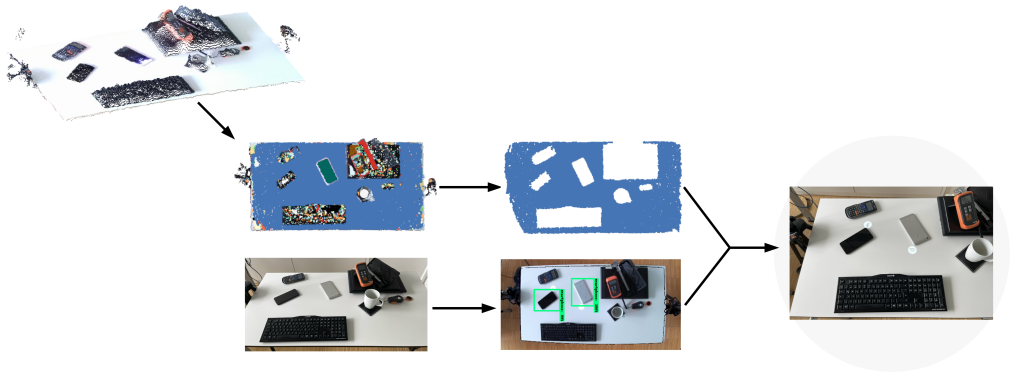


Fig. 39. Intercepting shared notification services and prompting users using the tabletop.

### 4.7.2 Pilot study

While no formal usability study has been done, we invited four users to participate in a qualitative pilot study. The study sought to identify drawbacks associated with using Traceless while learning the perceptions of end-users. The study also targeted discovering applications envisioned by users. Acquaintances and work colleagues were invited to participate. The only requirement outlined for the study was owning a personal mobile device. Each user was allocated a session where:

a. The goal of the research was outlined.
b. Traceless's features and limitations were described.
c. Two Android smartphones were presented (Nexus 5 and Motorola G7 already paired with Traceless).

We informed the users that both devices came pre-installed with Google Hangouts and Discord and paired with Traceless beforehand. We also elaborated on how Traceless intercepts message notification-service signatures and not the messages themselves. The users were then asked to connect one device to Traceless by merely placing it on the tabletop's surface at an arbitrary position. Traceless would then attempt to connect with that device. Upon successful handshake, we asked the user to utilize the other device to send a chat message to the one on the tabletop. Once sent, Traceless would intercept the notification and project the corresponding chat icon on the vacant surface space closest to the device laying on the tabletop. We then asked users to interact with the mobile device or surface projection. They could slide the device across the surface to see how the projected icon would follow along, intuitively avoiding occluded surface space. Alternatively, the users could occlude the projection with their hands. Once more, Traceless would sense the occlusion, intuitively moving the projected icon onto vacant surface space closest to the device. To dis-engage device-surface interaction, the users were asked to simply pick the device up. After the active demonstration, we ensued with semi-structured interviews to conclude each session. The interview questions (see Table 7) were open-ended and inclined towards broader discussions, as led by the users.

Table 7. Semi-structured interview questions.

| |
|---|
| How easy was it to connect a device to Traceless? |
| How pleasant were the interactions? |
| How practical are the interactions? |
| What possible applications can you envision using Traceless? |
| Open-ended questions |

### 4.7.3 Observations

All users had no observable difficulties with connecting mobile devices, which merely required placing a device on the tabletop. We observed how users expected initiating some action to start things off and were surprised to learn that nothing more was required outside resting a device on the tabletop. At first, users deliberately avoided occluding the area of projection. After we explained they could act as they would in daily life, they occluded the icon and were surprised how the projected icon responded to their movements. After experiencing the aforementioned, the users acted more freely and started occluding the projection with objects on the tabletop to get the projected media to move.

### 4.7.4 User perceptions

The first user discussed connecting a device with Traceless, "…its automatic and you dont have think about it or press anything. Super easy!" After occluding the projected media several times using hands and tabletop objects, they remarked, "When too much is on the table, I guess its harder to find the icon." User #1 conclude with how they saw themselves using Traceless in the everyday context, "This would be great for cooking. I could plan my recipes and have Traceless put them on the wall or kitchen counter without moving anything around." The second user discussed improvement that could be made to better suite Traceless to their office workflow, "The projected media seems to flicker and is not high resolution …its definitely practical but better resolution would be best especially for instance when I synchronize my calendar, tasks and need it to be legible from a distant." The third user remarked on the use of Bluetooth and the need to install third-party software on mobile devices to use Traceless, "Everyone's already familiar with Bluetooth …it's convenient …I reckon installing custom software may be a barrier, but if people see the benefits, they'll probably be like 'Oh yeah, that's cool—I could definitely use that."' They also pointed out the need for control to be left to the end-user, "…while it's helpful to view apps on the large surface, would the control aspect still be married to the touch screen?" User#3 concluded with sharing thoughts on possible applications, "Creating a collaborative space or showing people what's on your phone, or making a shared space would let people share stuff without crossing the boundary of personal space …I can imagine thats quite practical …a to-do list or a progress bar throughout the day without grabbing too much attention would be handy." The fourth user discussed challenges, suggesting possible extensions and

features, “This application is interesting; I mean—if you really want to not touch your phone and still interact with it . . . one question though, how does the application choose the best possible vacant space? Could the user point with their finger to suggest an area of interest?” User #4 also shared their thoughts on possible applications, “There’s definitely environments where you don’t want to be holding your phone. You could be cooking or doing a dangerous job. I mean—no one wants to drop their phone. So, it might be helpful to use the nearest surface for support. Another application would be in a meeting where you would want to share information on a shared surface.”

The perceptions of users converge towards a variety of exciting applications using Traceless. A critical issue to address is scaling projected media in a manner that promotes legibility (User #2). Also, later features need to consider de-coupling some of the control from mobile devices to incorporate mid-air gestural control (User #4). A commonality between users #3 and #4 was a conscious sense of personal space. In this regard, users identified how Traceless could be leveraged for collaborative workspaces (User #3) and sharing information on personal mobile devices using shared surfaces (User #4) without crossing personal space boundaries. User #4 also spoke to Traceless’s role in safety-critical work environments in addition to everyday environments where end-users may wish to free their hands without disconnecting from their digital assistants.

## 4.8 DISCUSSION

*Limitations.* Our implementation assumes the factory-calibrated extrinsic parameters between the depth and color camera of the kinect are accurate. However, distorted color-to-depth-transformed images suggest there is a margin of error. While this limitation can be addressed, e.g., by using an optical alumina calibration chessboard for calibrating the depth camera [205], it has not been addressed in this study. Adopting YOLOv5 allows Traceless to leverage state-of-the-art in object detection. However, it remains necessary to collected an annotated image-object dataset specific to tabletop environments. While this is feasible for our proof of concept, we look to a more general solution for future applications and discuss it as part of future recommendations. Another limitation is that if two or more mobile devices are placed onto the tabletop surface simultaneously, our implementation currently has no way of distinguishing concurrently established Bluetooth links. Limiting our initial prototype to Android smartphones simplifies validating the hardware—software concept. However, this initial concept needs to be extended to enable integrating variable mobile devices.

*Future work.* For the limitation of identifying and tracking devices placed simultaneously onto a tabletop’s surface, one approach would be to cache devices profiles, i.e., store the physical dimensions and Bluetooth address of each detected device [54]. This cache could then be utilized for future disambiguation. In addition to extending custom software to enable integration of variable mobile devices, future work needs to consider migration from Bluetooth to Bluetooth Low Energy (BLE) that is mesh-enabled [19].

In principle, mesh-enabled BLE would allow mobile devices to communicate with Traceless over a self-configuring network. It would also enable multiple devices to share the same services by broadcasting state. Future work must also consider capturing an image-object dataset that generalizes well for tabletop surface applications in the everyday context. Such a dataset would contribute to robust detection of mobile devices by accounting for the overview and the principal distance typical of overhead projector-camera systems, providing an alternative to existing datasets that predominantly comprise front-view captured images. Detection models trained using such a dataset would also, in theory, be able to characterize generic interactions between mobile devices and surfaces in the everyday context. Our future work also considers a formal usability study to learn user behavior, needs, and expectations. Such a study would ensure the development of features that are congruent with user requirements.

## 4.9 CONCLUSION

In this paper, we have highlighted the gap between existing approaches for augmenting interactive surface environments and advances in small-form-factor sensors and CNN-based object detection. Toward bridging the gap, a projector-camera unit has been employed to develop a concept for augmenting uninstrumented tabletop surfaces and integrating personal mobile devices. In addition to outlining hardware configuration, a detailed development trail of how to combine depth perception with modern CNN-based object detection to achieve spatial awareness has been presented, appending theoretical formulations and open-source resources to streamline exploring the proposed approach Our work highlights how emerging CNN-based approaches can be leveraged for spatial augmented reality. "…projection must be continually adjusted to respond to dynamic, moving surfaces sensed by depth cameras." — [221]. Acknowledging this fact, the contributions presented in this paper facilitate one possible approach to integrating mobile devices into augmented tabletop environments taking spatial awareness into account while highlighting the promising aspect of developing interaction concepts based on emerging technologies and techniques.

## ACKNOWLEDGMENTS

The work presented in this paper contributes towards Vernetzte Intelligente Gegenstände durch, auf und um interaktive Tische im Alltag (VIGITIA); research related to networking intelligent objects through, on, and around interactive tables in everyday life; a project funded by the Bundesministerium für Bildung und Forschung (BMBF), grant number 16SV8288.

## 4.10 APPENDIX

### A TSAI'S APPROACH

Tsai proposed approaching geometric calibration in two stages: In the first stage, the intrinsic, extrinsic, scale, and distortion parameters are estimated using least-squares.

Rotation is omitted in the first stage to simplify initial parameter estimation to analysis of linear equations. In the second stage, estimated parameters are optimized using non-linear methods, reducing error using the Levenberg-Marquardt algorithm [146]. The resulting parameters are then utilized to determine correspondences between the world space, camera space, and image space coordinates systems.

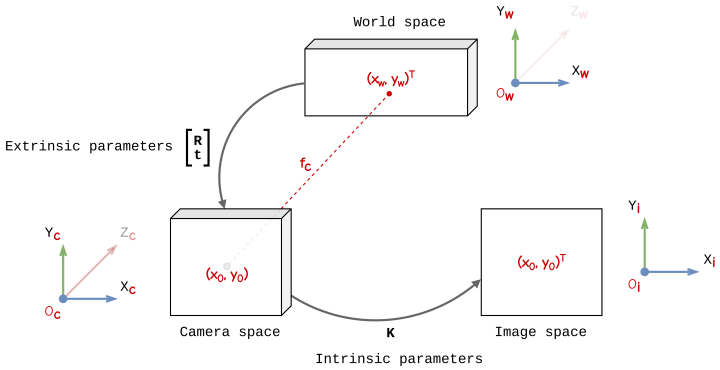



Fig. 40. Geometric camera calibration. Calibration considers: (1) the recovery of interior orientation, i.e., **K** and (2) the recovery of exterior orientation, i.e., **R** and **t**.

## A.1 Notation

Bold letters are used to represent vectors and matrices. Italicized letters are used to represent scalar coefficients. Italicized case letters are used to represent points. For orientation, we use the right-hand rule for all coordinate systems, i.e., the $+x$, $+y$, and $-z$ axes point right, up, and forward respectively. The superscript $^\tau$ is used to denote the transpose of a vector or a matrix. Bold subscripts are used to indicate relative space of coordinate points, such that $x_\mathbf{c}$, $y_\mathbf{c}$, $z_\mathbf{c}$ corresponds to a point in camera space, $x_\mathbf{i}$, $y_\mathbf{i}$, $z_\mathbf{i}$ corresponds to a point in image space, and $x_\mathbf{w}$, $y_\mathbf{w}$, $z_\mathbf{w}$ corresponds to a point in world space. The camera space, image space, and world space origins are denoted using $O_\mathbf{c}$, $O_\mathbf{i}$, $O_\mathbf{w}$ respectively, where $O_\mathbf{c}$ corresponds to the point $(x_0, y_0)$, and $O_\mathbf{i}$ corresponds to the point $(x_0, y_0, )^\tau$. We use $f$ to represent the principal distance that runs along the $z_\mathbf{i}$ axis, i.e., the distance along the optical axis from $O_\mathbf{i}$ to the target image plane. Lastly, we use $E$ to represent the error, i.e., the difference between predicted $(\hat{x}_\mathbf{i}, \hat{y}_\mathbf{i})^\tau$ and observed $(x_\mathbf{i}, y_\mathbf{i})^\tau$ based on the known $x_\mathbf{w}$, $y_\mathbf{w}$, $z_\mathbf{w}$ coordinate points.

## A.2 Intrinsic parameters

The intrinsic parameters $x_0$, $y_0$ and $f$ (see Fig. 40) are recovered to describe the geometric relationship between camera space and image space.

$$\frac{x_\mathbf{i} - x_0}{f} = \frac{x_\mathbf{c}}{z_\mathbf{c}} \qquad \frac{y_\mathbf{i} - y_0}{f} = \frac{y_\mathbf{c}}{z_\mathbf{c}} \tag{8, 9}$$

Assuming $x_\mathbf{c}$ and $y_\mathbf{c}$ are parallel to $x_\mathbf{i}$ and $y_\mathbf{i}$, perspective projection (Eq. 8 and Eq. 9) is used to recover $x_0$,$y_0$, and $f$.

### A.3 Extrinsic parameters

The extrinsic parameters, i.e., rotation **R** and translation **t** are recovered to describe the geometric relationship between camera space and world space. **R** and **t** each have three degrees of freedom. Given a point in world space $P_{\mathbf{w}}$, and assuming that $z_{\mathbf{c}}$ is perpendicular to and intersects the target surface in world space where $z_{\mathbf{c}} = 0$, the corresponding point in camera space $P_{\mathbf{c}}$ is given by Eq. 10.

$$P_{\mathbf{c}} = \mathbf{R}(P_{\mathbf{w}}) + \mathbf{t} \tag{10}$$

where,

$$P_{\mathbf{c}} = (x_{\mathbf{c}}, y_{\mathbf{c}}, z_{\mathbf{c}})^{\tau}, P_{\mathbf{w}} = (x_{\mathbf{w}}, y_{\mathbf{w}}, z_{\mathbf{w}})^{\tau}, and\ \mathbf{t} = (t_x, t_y, t_z)^{\tau}.$$

Given **R** and **t**, the orthonormal matrix is expressed component form Eq. 11.

$$\begin{bmatrix} x_{\mathbf{c}} \\ y_{\mathbf{c}} \\ x_{\mathbf{c}} \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} x_{\mathbf{w}} \\ y_{\mathbf{w}} \\ x_{\mathbf{w}} \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} \tag{11}$$

### A.4 Rotation

To recover the rotation from camera space to image space, $x_0, y_0$ is firstly expressed in the form

$$x'_{\mathbf{i}} = x_{\mathbf{i}} - x_0 \qquad\qquad y'_{\mathbf{i}} = y_{\mathbf{i}} - y_0 \tag{12, 13}$$

such that,

$$\frac{x'_{\mathbf{i}}}{f} = s\frac{x_{\mathbf{c}}}{z_{\mathbf{c}}} \qquad\qquad \frac{y'_{\mathbf{i}}}{f} = s\frac{y_{\mathbf{c}}}{z_{\mathbf{c}}} \tag{14, 15}$$

The direction of a point on the image, as measured from the principal point (being independent of radial distortion), is used in Eq. 16.

$$\frac{x'_{\mathbf{i}}}{y'_{\mathbf{i}}} = s\frac{X_C}{Y_C} \tag{16}$$

For non-planar target surfaces:
Expanding Eq. 16 in terms of Eq. 11 yields the linear homogeneous equation Eq. 17.

$$s(r_{11}x_{\mathbf{w}} + r_{12}y_{\mathbf{w}} + r_{13}z_{\mathbf{w}} + t_x)y'_{\mathbf{i}} - (r_{21}x_{\mathbf{w}} + r_{22}y_{\mathbf{w}} + r_{23}z_{\mathbf{w}} + t_y)x'_{\mathbf{i}} = 0 \tag{17}$$

with eight unknowns:

$$sr_{11},\ sr_{12},\ sr_{13},\ r_{21},\ r_{22},\ r_{23},\ st_x,\ and\ t_y,$$

For each point $(x_{\mathbf{w}}, y_{\mathbf{w}}, z_{\mathbf{w}})^{\tau}$ in world space, Eq. 17 is used to evaluate the corresponding point $(x_{\mathbf{i}}, y_{\mathbf{i}})^{\tau}$ in image space. Tsai converts the homogeneous equation into an in-homogeneous equation by arbitrarily setting a single unknown to 1 and evaluating the rows of the rotation matrix to estimate a scale factor, i.e., assuming the rows of the rotation matrix to be normal.

$$r_{11}^2 + r_{12}^2 + r_{13}^2 = 1 \qquad\qquad r_{21}^2 + r_{22}^2 + r_{23}^2 = 1 \tag{18, 19}$$

A factor, $c$, is introduced and used to recover the ratio of the horizontal to vertical pixel spacing $s$ Eq. 18 and Eq. 19.

$$c = \frac{1}{\sqrt{{r'_{21}}^2 + {r'_{22}}^2 + {r'_{23}}^2}} \qquad\qquad \frac{c}{s} = \frac{1}{\sqrt{(sr'_{11})^2 + (sr'_{12})^2 + (sr'_{13})^2}} \tag{20, 21}$$

For planar target surfaces:
The scaling for the image coordinates is assumed to be correct. Given $z_{\mathbf{w}} = 0$ for all points on the target surface, $r_{13}$, $r_{23}$, and $r_{33}$ are dropped from the image coordinates which results in Eq. 22

$$(r_{11}x_{\mathbf{w}} + r_{12}y_{\mathbf{w}} + t_x)y_{\mathbf{i}'} - (r_{21}x_{\mathbf{w}} + r_{22}y_{\mathbf{w}} + t_y)x_{\mathbf{i}'} = 0 \tag{22}$$

with the six unknowns:

$$r_{11}, r_{12}, r_{21}, r_{22}, t_x\ and\ t_y.$$

For each point $(x_{\mathbf{w}}, y_{\mathbf{w}}, z_{\mathbf{w}})^{\tau}$ in world space, Eq. 17 is used to evaluate the corresponding point $(x_{\mathbf{i}}, y_{\mathbf{i}})^{\tau}$ in image space. Once more, the homogeneous is made in-homogeneous by arbitrarily setting one unknown to 1, i.e.,

$$r'_{11}, r'_{12}, r'_{21}, r'_{22}, t'_x\ and\ t'_y = 1.$$

### A.5 Translation

To determine the translation $(t_x, t_y, t_z)^{\tau}$, the components $t_x$ and $t_y$, recovered from Eq. 17, are used to evaluate the translation from camera space to image space $t_z$.

$$s(r_{11}x_{\mathbf{w}} + r_{12}y_{\mathbf{w}} + r_{13}z_{\mathbf{w}} + tx)f - x'_{\mathbf{i}}t_z = (r_{31}x_{\mathbf{w}} + r_{32}y_{\mathbf{w}} + r_{33}z_{\mathbf{w}})x'_{\mathbf{i}} \tag{23}$$

$$(r_{21}x_{\mathbf{w}} + r_{22}y_{\mathbf{w}} + r_{23}z_{\mathbf{w}} + ty)f - y'_{\mathbf{i}}t_z = (r_{31}x_{\mathbf{w}} + r_{32}y_{\mathbf{w}} + r_{33}z_{\mathbf{w}})y'_{\mathbf{i}} \tag{24}$$

The equations Eq. 11, Eq. 14, and Eq. 15 are evaluated to linear homogeneous equations Eq. 23 and Eq. 24 and used to determine $f$ and $t_z$.

### A.6 Horizontal scale factor

Tsai factors the deviation in image cell size ratios caused by signal processing for frame grabber standard output in cameras.

$$\frac{x_{\mathbf{i}} - x_0}{f} = s\frac{x_{\mathbf{c}}}{z_{\mathbf{c}}} \tag{25}$$

A horizontal scale factor $s$ is introduced to recover the correct ratio of image cell size in the horizontal and vertical directions and effectively account for the unknown pixel spacing ratios in the $x$ and $y$ directions.

### A.7 Distortion coefficients

Projectors and cameras introduce variable lens distortion [161, 183]. Power series coefficients are recovered using Eq. 26 and Eq. 27 to mitigate radial distortion, viz, pin-cushion and barrel distortion.

$$\delta\, x = x(\kappa_1 r^2 + \kappa_2 r^4 + \ \dots) \qquad \delta\, y = y(\kappa_1 r^2 + \kappa_2 r^4 + \ \dots) \tag{26, 27}$$

$$\delta\, x = -y(\in_1 r^2 + \in_2 r^4 + \ \dots) \qquad \delta\, y = x(\in_1 r^2 + \in_2 r^4 + \ \dots) \tag{28, 29}$$

Similarly, tangential distortion coefficients $(\in_1, \dots)$ are recovered using Eq. 28 and Eq. 29.

### A.8 Error minimization

As a final step, the error between the predicted $(\hat{x}_{\mathbf{i}}, \hat{y}_{\mathbf{i}})^\tau$ and observed $(x_{\mathbf{i}}, y_{\mathbf{i}})^\tau$ is minimized using Levenberg-Marquardt.

$$E = \sum_{n=1}^{N} (x_n - \hat{x}_n)_{\mathbf{i}}^2 + \sum_{n=1}^{N} (y_n - \hat{y}_n)_{\mathbf{i}}^2 \tag{30}$$

## B PROJECTOR-CAMERA CALIBRATION

### B.1 Calibrating the kinect

To calibrate the kinect, a conventional chessboard with known dimensions is utilized. The chessboard is moved about the fixed field of view of the kinect and multiple chessboard images are captured from different orientations. The kinect's extrinsic parameters **R** and **t** and intrinsic parameters **K** (see Appendix A) are recovered using OpenCV [162]. The parameters are then used to determine the projection matrix **P**

(Eq. 31) which is utilized to map points from world space to image space (Eq. 32).

$$\mathbf{P} = \mathbf{K} \times [\ \mathbf{R} \mid \mathbf{t}\ ], \qquad \begin{bmatrix} X_\mathbf{i} \\ Y_\mathbf{i} \\ Z_\mathbf{i} \end{bmatrix} = \mathbf{P} \begin{bmatrix} X_\mathbf{w} \\ Y_\mathbf{w} \\ Z_\mathbf{w} \\ 1 \end{bmatrix} \tag{31, 32}$$

where $X_\mathbf{w}$, $Y_\mathbf{w}$, $Z_\mathbf{w}$ denotes world-space coordinates and $X_\mathbf{i}$, $Y_\mathbf{i}$, $Z_\mathbf{i}$ denotes image-space coordinates. With know $\mathbf{R}$, $\mathbf{t}$, and $\mathbf{K}$, OpenCV's `cv::getOptimalNewCameraMatrix` is used to refine the resultant camera matrix and `cv::undistort` is used to undistort the camera image. `cv::projectPoints` is also used to evaluate the reprojection error.

## B.2 Projector-kinect calibration

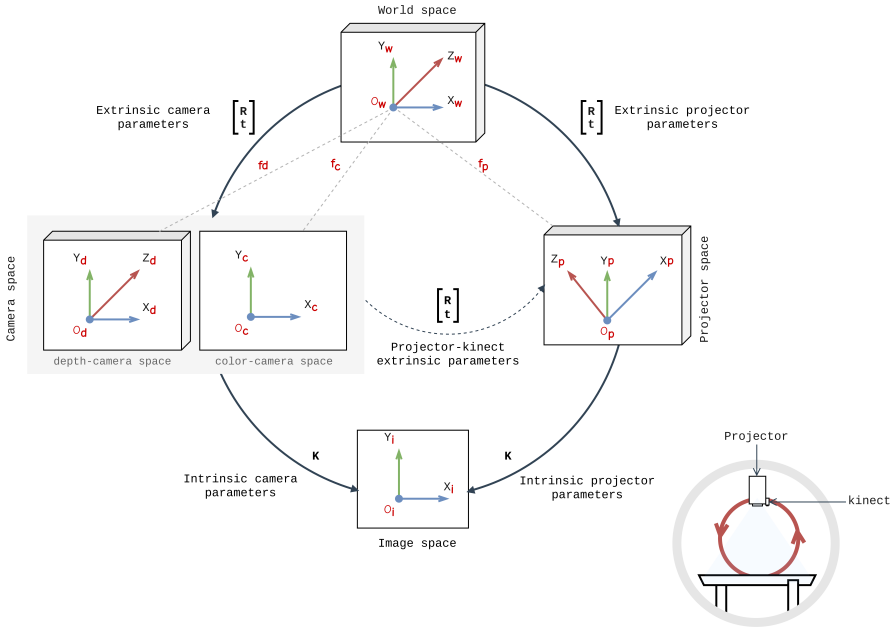



Fig. 41. Projector-kinect calibration. Mapping between world space and image space is achieved using both the intrinsic and extrinsic parameters of the camera and projector together with the extrinsic parameters between the camera and the projector.

*Notation.* Bold case letters are used to represent vectors, matrices and sets of points. Italicized letters are used to represent coordinate points. Lastly, bold subscripts are used to indicate relative space. Fig. 41 depicts the calibration parameters necessary for projector-kinect calibration and Algorithm 5 presents an abstract outline of the method.

**Algorithm 5:** Abstract algorithm for projector-kinect calibration.

1 Generate chessboard images.
2 Project chessboard image onto tabletop surface.
3 Capture a minimum of three synchronous RGBD images of the projected chessboard from different poses.
4 Evaluate camera-space coordinates using `cv::findChessboardCorners`.
5 Estimate world-space coordinates using synchronous RGBD captures.
6 Recover the intrinsic and extrinsic parameters of the projector using `cv::calibrateCamera`.
7 Recover the extrinsic parameters between the projector and kinect using `cv::stereoCalibrate`.

First, a chessboard image is generated, and the coordinate points of the chessboard's inner corners are collected.

$$\mathbf{C_i} = \{c_1, c_2, \ldots, c_m\},\ c_i = (x_i, y_i)_\mathbf{i} \in \mathbb{R}^2, \tag{33}$$

where $_m$ is the size of the chessboard and $(x_i, y_i)$ denote the image-space coordinates of the chessboard's inner corners. The generated chessboard image is projected onto the tabletop surface, and the kinect is used to capture synchronous RGBD images of the chessboard. Note, a minimum of three synchronous RGBD image captures are necessary [237]. We variate poses using a rigid whiteboard slightly larger than the projected chessboard. With the chessboard image cast onto the whiteboard, we move the whiteboard about the field of view of the kinect. After capturing the projected chessboard images, OpenCV's `cv::findChessboardCorners` is used to evaluate the camera-space coordinates from the captures. Given synchronized RGBD captures,

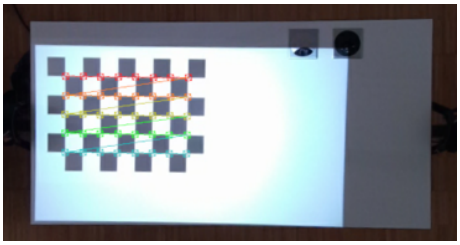


Fig. 42. Using a projected chessboard and OpenCV's `cv::findChessboardCorners` for projector calibration.

the resultant camera-space coordinates found using `cv::findChessboardCorners` are expressed as Eq. 34 and 35 (relative coordinate spaces depicted in Fig. 41).

$$\mathbf{C_c} = \{c_1, c_2, \ldots, c_m\},\ c_i = (x_i, y_i)_\mathbf{c} \in \mathbb{R}^2, \quad \mathbf{C_d} = \{c_1, c_2, \ldots, c_m\},\ c_i = (x_i, y_i, z_i)_\mathbf{d} \in \mathbb{R}^3. \tag{34, 35}$$

The world-space coordinates are approximated as follows. The $\mathbf{X_d}$ and $\mathbf{Y_d}$ coordinate pairs,i.e., the 2D-plane points from Eq. 37, are translated along the $z$-axis to the origin $O_\mathbf{d}$. This transformation is done using the centering matrix (Eq. 36).

$$\mathbf{M}_n = \mathbf{I}_n - \frac{1}{n}\mathbf{J}_n, \qquad \mathbf{W} = \mathbf{M}_n\mathbf{C_d}. \tag{36, 37}$$

$\mathbf{M}_n$ demotes the centering matrix, $\mathbf{I}_n$ is the identity matrix of size $n$ and $\mathbf{J}_n$ is an $n \times n$ matrix of 1's. $\mathbf{W}$ is the canonical view of the image in depth-camera space. Singular value decomposition (Eq. 38), is then used evaluate the eigen decomposition of $\mathbf{W}$.

$$A = U\Sigma V^T \tag{38}$$

$$U = (u_1, \ldots, u_m), \qquad V = (v_1, \ldots, v_3), \qquad \Sigma = diag(\sigma_1, \ldots, \sigma_3),$$

where $_m$ denotes the number 3D points. For singular value decomposition, we recall: the left singular vectors $U$ correspond to the $x, y, z$ column vectors, the right singular vectors $V$ correspond to 3D-row vectors, and $\Sigma$ corresponds to the diagonal singular values linked to the left and right singular vectors. For the final step, we rotate the $\mathbf{X_d}$ coordinate values using a transpose of the unitary matrix: $U^{-1}\mathbf{X_d}$, and zero the $\mathbf{Z_d}$ coordinate values to estimate world-space coordinates;

$$\mathbf{C_w} = \{c_1, c_2, \ldots, c_m\}, \; c_i = (x_i, y_i, z_i)_\mathbf{d} \in \mathbb{R}^3. \tag{39}$$

Given image-space coordinates and corresponding world-space coordinates, OpenCV's `cv::calibrateCamera` is used to recover the projectors intrinsic and extrinsic parameters. To recover the extrinsic parameters between the projector and the kinect, OpenCV's `cv::stereoCalibrate` is used with: (i) the intrinsic parameters of the projector, (ii) the intrinsic parameters of the kinect's depth camera, (iii) world-space coordinates of the chessboard corners, (iv) image-space coordinates of the chessboard corners, and (v) camera-space coordinates of the chessboard corners from the kinect's depth camera.

## 4.3 CRITICAL REMARKS

Exploring the interaction space between mobile devices and tabletop surfaces using Bluetooth was initially put forward by Wilson and Sarin in [224], i.e., Bluetable in 2007. However, while recent advances have propelled concepts such as collaborative spaces and shared media, a body of literature suggests that implementation strategies remain constrained to classical 2D computer vision techniques [11, 171]. In contrast Bluetable [224], the hardware-software concept presented in this chapter has introduced:

i. A spatial-augmented-reality strategy that exploits synchronous RGBD.

ii. A 3D point-cloud processing technique based on the open-source software solution contributed in the second study,[2].
iii. CNN-based object detection.

Synchronous RGBD is leveraged to combine classical 2D computer vision techniques with 3D point-cloud processing. For segmenting point clouds, 3dintact is employed to mitigate high computational costs on a CPU architecture, thereby enabling the processing of point clouds near real-time. As a core contribution: the method combining classical 2D computer vision techniques, 3D point-cloud segmentation, CNN-based object detection, and Bluetooth communication has been detailed to promote repeatability.

Besides outlining a robust approach to augmenting interactive surface environments for seamless interaction with mobile devices, the study has underlined the following. On the one hand, pervasive commodity depth cameras have vastly increased access to 3D point clouds. This proliferation is evident in fast-expanding 3D point cloud repositories, which are openly accessible to the public. On the other hand, advances in machine learning techniques motivate leveraging these repositories to train models specifically for detecting actor-object interactions around everyday surfaces, e.g., a dataset of input gesture commands performed by an actor. Using machine learning models that exploit large 3D point cloud data sets for detecting actor-object interactions around everyday surfaces is a research avenue that may be the de facto approach in the near future [13, 91]. As such, the research we have presented in this paper promotes this research direction.

### 4.4 SUMMARY

A hardware-software concept for enabling the interaction spaces between mobile devices and interactive surface environments has been presented. The concept demonstrates how the open-source solution contributed in the second study can be leveraged with state-of-art in CNN-based object detection. Moreover, The research has outlined a transparent development trail to support the development of new interaction concepts using the novel approach.

[2]https://github.com/edisonslightbulbs/3DINTACToolkit

*"Scientific theory and its application to the growing needs of mankind advance hand in hand." — Cargill Gilston Knott*

# 5
# Significance and research impact

This section discusses the significance of the research contributions presented in chapters 2, 3, and 4.

## 5.1 ARTEFACT

The Artefact framework facilitates documenting and communicating technical hardware, middleware, and software specifications unambiguously. It outlines a practical approach to sharing interactive-surface models as end products. In the following, we discuss the implication of models as end products, within and beyond the research community.

### 5.1.1 Transparent development trails

Besides a system for documenting technical specifications, the Artefact framework captures and expresses the ontology between hardware, middleware, and software. These captured properties and relations provide insight into system design, conveying the motivation behind component selection in each layer. Capturing and expressing the ontology between concepts supports artifact replication in two respects:

1. The rationale behind system design can be communicated.
2. Technical requirements driving component selection can be conveyed.

As indicated by the literature presented in chapter 2, there are no strategies for sharing prototypical designs in HCI, viz, documenting and communicating technical specifications unambiguously. The Artefact framework puts forward one approach to addressing this challenge. It enables capturing technical specifications unambiguously. Concept ontology, i.e., for the hardware, middleware, and software layers, is presented in a rationable unified view. While documenting technical specifications supports replications, conveying ontology bridges the communication gap between originators of prototypes and future developers. This proposition is supported by findings from the case study presented in chapter 2. As affirmed by the software developers from the case study:

"Another use-case of the Artefact framework is knowledge transfer. It makes it simple to learn how research prototypes have been developed, from design to implementation."

### 5.1.2 Technical discussions

The Artefact framework outlines properties and relationships between concepts across the separate domains by giving prominence to modeling the hardware, middleware, and software layers as independent subsystems. By modeling each subsystem using formal notation, the framework provides a method for contrasting different prototypes unambiguously at a subsystem level: a tacit benefit observed during the case study with the software developers (discussed in chapter 2). Another tacit benefit observed during the case study: model representations of interactive surface prototypes elicit technical discussions. On a broader scale, comparative analysis of between existing prototypes can be utilized to drive technical discussions in workshops and focus groups, which in turn would promoted advancing interactive surface concepts.

### 5.1.3 Representation of structurally complex systems

The software developers from the case study also pointed to how models significantly increase the learnability of interactive surface systems, particularly for prototypes with structurally complex software APIs. Improved learnability for software implementations is highly significant in two ways: First, steep learning curves for complex open-source projects can be mitigated by employing unambiguous models to describe API designs, thereby making them more straightforward to rationalize. And second, API-model depictions (see software models in Fig 43.b and Fig 44.c) can be leveraged to onboard new developers and support them in navigating structurally complex APIs when targeting optimization entry points. Moreover, the Artefact framework can be employed as an academic guideline to inform developers new to the space of interactive surface environments.

### 5.1.4 Code and doc generation

Another tacit benefit identified by software developers from the case study is using the resulting UML-based model representations to generate code and documentation. The instant generator [3] can be leveraged to generate source files from UML class diagrams. Also, as already noted in chapter 2, the UML models could be leveraged to generate software documentation using doxygen [4] and graphviz [5].

## 5.2 3DINTACT

3dintact [6] streamlines segmenting interaction volumes [220] from 3D point-cloud representations of tabletop environments, i.e., using CPU architectures. It provides

[3] https://www.visual-paradigm.com/features/code-engineering-tools/
[4] https://www.doxygen.nl/index.html
[5] https://graphviz.org/
[6] https://github.com/edisonslightbulbs/3DINTACToolkit

developers with a simple API that leverages non-trivial point-cloud operations to facilitate a preprocessing filter that mitigates the computational space in dense point-cloud representations. By facilitating such a preprocessing building block, 3dintact promotes exploiting 3D point clouds for interactive surface applications. The following elaborates on the significance of the toolkit as a research contribution.

### 5.2.1 Modular software design

3dintact's software design emphasizes modularity (see Fig 43.b). The applied algorithms, viz, DBSCAN, KNN, and the SVD are implemented as loosely coupled modules [7] to simplify module optimization. Loose coupling also promotes swapping the algorithmic modules with more advanced implementations in future. Similarly, 3dintact's API backend is modularized to promote code maintenance, extensibility, and future optimization.

### 5.2.2 Preprocessing for variable depth cameras

By design, the algorithmic strategy implemented by 3dintact generalizes to variable depth cameras. As such, it provides flexible preprocessing for depth cameras from different vendors, e.g., Intel's RealSense depth camera and Microsoft's Azure Kinect depth camera. While preprocessing time varies according to number of depth cameras employed and depth camera configuration, in principle, 3dintact can be leveraged with any combination of depth cameras simultaneously.

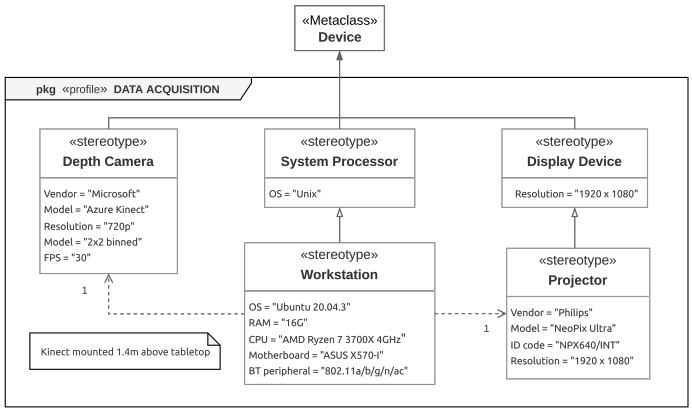
### 5.2.3 Aggregating 3D point-cloud representations of interaction volumes

While dense 3D point clouds are desirable and can be easily achieved by registering point clouds from multiple depth cameras, the consequent computational space is challenging to process on CPU architectures. Although the algorithmic strategy employed by 3dintact reduces computational space significantly, 3dintact does not scale optimally with an increase in the number of depth cameras employed. However, given dense 3D point-cloud representations from multiple depth cameras, 3dintact remains highly significant for aggregating 3D interaction volumes into data repositories. These repositories can be employed to train machine learning models to segment actors and objects together with candidate interaction regions and interactions. Such segmentation models could then be used in place of 3dintact and propel interactive surface applications on CPU architectures.

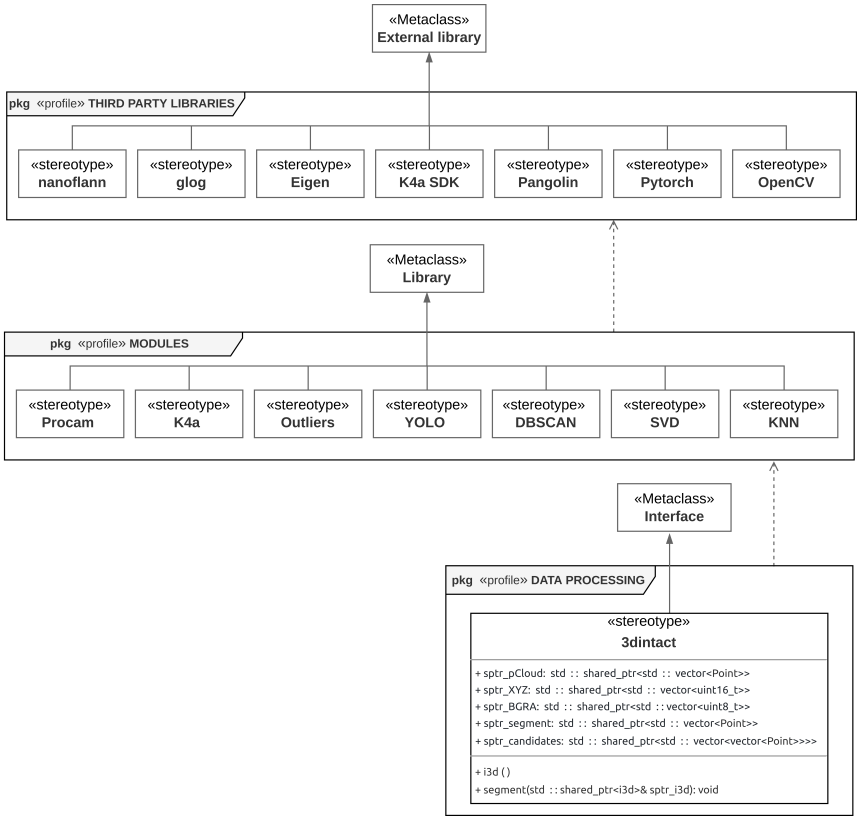
## 5.3 TRACELESS

Traceless is an academic interactive surface prototype. It is an example of how to leverage a project-camera system to integrate smartphones as active components in an interactive tabletop's computing environment. Traceless provides a clear engineering

[7] All modular implementations have been open-sourced on GitHub. The corresponding list of repositories has been appended to this document.



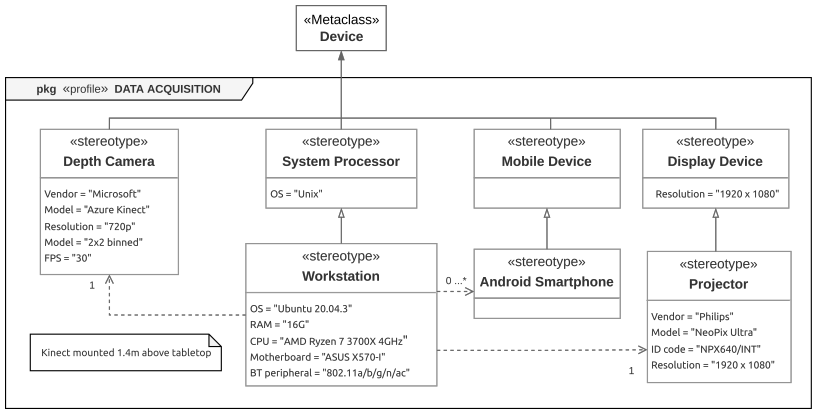

(a) Hardware model.



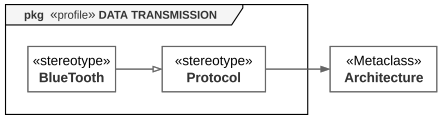

(b) Software model.

Fig. 43. A model of 3dintact using the Artefact framework. Regardless of structural complexity, the Artefact framework provides an unambiguous model of 3dintact's implementation. A concise and accurate overview of the system's implementation promotes rationalizing system architecture, identifying limitations, and navigating optimization entry points.

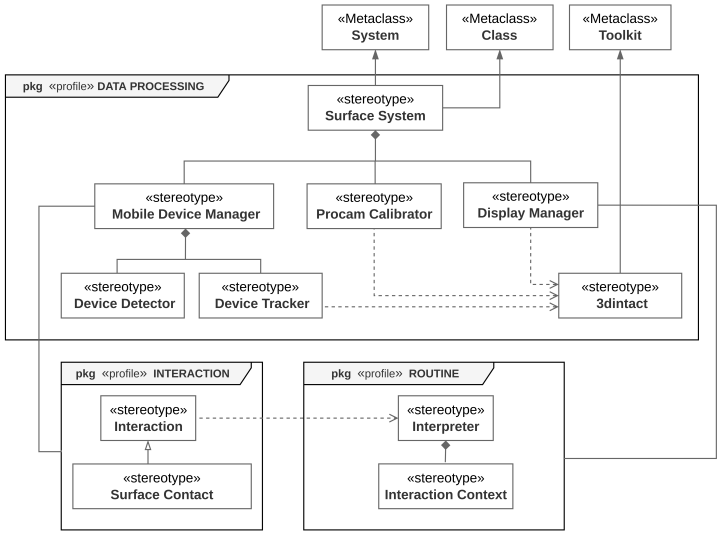
outline that supports developing technical applications. The following, elaborates on the significance of Traceless as an implementation guideline.




(a) Hardware model.




(b) Middleware model.




(c) Software model.

Fig. 44. A model of Traceless using the Artefact framework. The Artefact framework enables developers to flexibly select levels of abstraction. While the framework can capture low-level details precisely, it can also express structurally complex systems.

### 5.3.1 Supporting the exploration of tabletop and smartphone interactions

In the research space of HCI, the replication crisis is coupled with the consistent omission of technical implementation details. As a side effect: academic manuscripts serve little-to-no utility in supporting developers with design decisions and technical implementation details. Traceless outlines traceable design decisions and repeatable engineering steps while detailing associated theory as a basis for future development.

Moreover, the Traceless contribution aims to serve as a workbench that can be utilized to explore interaction concepts between tabletop surfaces and smartphones.

### 5.3.2 Integrating smartphones as active components in computing environments

Traceless combines machine learning with 3D point-cloud segmentation. While a machine-learning model, i.e., YOLOv5, [8] is used for detecting smartphones from 2D images, 3dintact is exploited to segment and propose candidate-interaction regions near real-time. Given synchronous RGBD data, combining object detection and spatial awareness enables Traceless to understand dynamic scenes adequately enough to identify targeted interaction contexts. From a technical vantage, Traceless is a robust approach to detecting and integrating smartphones as active components in computing environments. The model employed for the object detection model is a well-known open-source project accessible to the public. To support training the model, two notebooks detailing transfer learning applied, as applied in Traceless, have also been made freely available to the public: one outlining a TensorFlow-based approach, [9] and another outlining a PyTorch-based approach. [10]

## 5.4 SUMMARY

This chapter has presented the significance of the research contributions put forward in this dissertation. The impact of the Artefact framework, the 3dintact toolkit, and the Traceless prototype has been discussed. The contributions are freely open to the research community and the public at large. Hyper-references to research assets have also been provided.

[8] https://github.com/ultralytics/yolov5
[9] https://github.com/edisonslightbulbs/PTOD-model-training
[10] https://github.com/edisonslightbulbs/TFOD-model-training

*"Progress is made by trial and failure; the failures are generally a hundred times more numerous than the successes; yet they are usually left unchronicled." — William Ramsay*

# 6 Future work and conclusion

This final chapter discusses potential future work and concludes with a reflection on each study.

## 6.1 FUTURE WORK

In addition to the contributions put forward, the studies presented in chapters 2, 3, and 4 highlight promising research directions. The following discusses these research avenues.

### 6.1.1 Validating models

The Artefact framework lends from standard UML and employs UML's formal syntax to impose consistent and unambiguous model representation. However, currently, no methodology exists for validating models [159, 184]. To recollect: the Artefact framework can be used to communicate interactive surface implementations as well as model new prototypes. While modeling existing implementations merely requires capturing and expressing established concepts, implementing a prototype based on a newly developed model demands that the model first be validated. While a model-driven approach promotes unambiguous diffuse of hardware, middleware, and software models, there is still a need to establish a shared approach for validating models. This shortcoming needs to be addressed in future research.

### 6.1.2 A baseline for point-cloud processing using CPU architectures

Given an interactive surface application that exploits point clouds for scene understanding, employing the modern GPU is one approach to meeting the computational requirements necessary for processing 3D point clouds in real-time. However, small-form-factor and unobtrusiveness of compact projector-camera systems are simply more attractive for seamless installation in day-to-day environments. Given this significant factor, an important research step is establishing an initial set of benchmarks for processing point clouds absent a dedicated GPU. This initial set of benchmarks

would outline minimum compute performance required for small-form-factor solutions absent dedicated GPUs, as well as a reference for future CPU-based optimizations.

### 6.1.3 Synchronous-RGBD data sets

Beyond a robust strategy for enabling communication between smartphones and interactive tabletop surfaces, Traceless draws out the role of machine learning for future interactive surface environments. The prototypical concept can also be employed to accumulate an extensive repository of synchronous RGBD data in the wild, targeting specific interaction contexts on tabletops. The resulting data set could be employed for training machine learning models. RGB data could be used for training object detection models, and synchronous RGBD could be used to train a model to detect specific interaction contexts. A practical use case for such models would be as follows: finding the best projection area on day-to-day surface environments based on scene understanding, i.e., relative to multiple actors, their positions, detected interaction contexts, and surface clutter. Such machine learning models would augment 2D techniques without demanding additional resources for high-end compute performance.

### 6.1.4 Benchmarking pre-trained models

Traceless employs YOLOv5 [11] as an integral component for detecting smartphones based on image-object detection. YOLOv5 was selected based on three engineering aspects:

1. Open-source. In addition to being fast and performant, the development of YOLOv5 is community-driven. Moreover, application development is supported by triage on Github.
2. Extensive pre-training. YOLOv5 is pre-trained based on the COCO dataset: a dataset consisting of 80 object classes. [12]
3. Extensive documentation for C++ development. YOLOv5 is based on the PyTorch framework which targets C++ applications. [13] This consideration is highly significant as the PyTorch framework allows for seamless integration with 3dintact [14] which has also been developed in C++.

As a pre-trained model, YOLOv5 facilitated transfer learning. The model's knowledge of the 80 object classes from the COCO dataset was exploited to fine-tune detecting smartphones. This was achieved by further training the model using a sample of 250 smartphone images.

You et al. have discussed the challenge of pre-trained model selection. In [236], the authors have emphasized how assessing pre-trained models for a target task and selecting the best model is an underexplored problem. While the study presented in chapter

[11] https://github.com/ultralytics/yolov5
[12] https://cocodataset.org
[13] https://pytorch.org/
[14] https://github.com/edisonslightbulbs/3DINTACToolkit

4 outlines engineering criteria motivating the selection of YOLOv5, it omits benchmarking different pre-trained models for the task of detecting smartphones. Although YOLOv5's knowledge was successfully transferred to a detector solely targeting smartphones, a research avenue that must be considered in future work is benchmarking different detection models to baseline model selection [236].

### 6.1.5 Benchmarking mobile-device detection

Another critical consideration for future work is evaluating mobile-device detection. The first step in this direction would be establishing evaluation metrics, i.e., quantifiable measures to assess the detection of mobile devices. Examples of measures that could be considered for evaluating mobile-device detection include detection latency and false positives/negatives.

These evaluation metrics would, in turn, facilitate not only analyzing detection but also benchmarking performance in interactive tabletop applications. For example, given duration and frame rate, metrics could be employed to determine accuracy, i.e., the total number of correct detections over the total number of expected detections.

## 6.2 CONCLUSION

The first study has proposed a UML-based framework that promotes models as end products. The framework facilitates a shared view for interactive surface prototypes. On the one hand, the proposed model-driven approach enables detailed descriptions of multi-layered interactive surface prototypes: a significant step towards mitigating the replication crisis. And on the other, disseminating unambiguous model descriptions based on formal syntax can bridge the gap between originators of prototypes and future developers. The contribution of the first study promotes communicating technical implementation aspects towards increasing the transparency and reproducibility of research prototypes. Besides bridging the communication gap between originators of concepts and future developers, the framework promotes an intuitive entry point for designing multi-layered interactive surface prototypes: a practical solution to the first subproblem.

The second study has considered the accelerated increase of variable commodity depth cameras. Given the common use of depth cameras for research artifacts in the space of interactive surface environments, the study underlines a need to support developers exploit 3D point clouds. The contribution of the second study is a preprocessing building block for interactive surface implementations that require point cloud-based scene understanding. The toolkit put forward by the study is open-source and thus openly accessible to all developers. For flexibility, the toolkit's software design emphasizes modularity. Moreover, the accompanying academic manuscript details implemented algorithms to promote learnability and future optimization.

An archetypical prototype has been implemented toward tackling the final subproblem. The prototype exemplifies how to leverage the open-source solution contributed by the preceding study. The prototype's implementation is coupled with a technical

outline to provide a transparent development trail. The technical outline elaborates on how to leverage Bluetooth communication, CNN-based object detection, and point cloud-based depth perception. Given the growing interest in extending ubiquity from mobile devices to surface objects through spatial augmented reality, the proposed prototype provides a robust strategy for integrating mobile devices into virtual computing environments.

This dissertation has presented three cumulative studies that contribute towards supporting the design and implementation of interactive surface concepts. The contributions are freely open to the research community. The established findings collectively target supporting developers implement interactive surface environments.

*"A man will turn over half a library." — Samuel Johnson*

# Bibliography

## REFERENCES


[1] Kai Adam, Judith Michael, Lukas Netz, Bernhard Rumpe, and Simon Varga. 2019. Enterprise Information Systems in Academia and Practice-Lessons learned from a MBSE Project. In *EMISA Forum*. De Gruyter, Gesellschaft für Informatik e.V, Bonn, Germany, 59–66.

[2] Matt Adcock, David Feng, and Bruce Thomas. 2013. Visualization of off-surface 3D viewpoint locations in spatial augmented reality. In *Proceedings of the 1st Symposium on Spatial User Interaction (SUI '13)*. Association for Computing Machinery, New York, NY, USA, 1–8. https://doi.org/10.1145/2491367.2491378

[3] Nathalie Aquino. 2009. Adding Flexibility in the Model-Driven Engineering of User Interfaces. In *Proceedings of the 1st ACM SIGCHI Symposium on Engineering Interactive Computing Systems* (Pittsburgh, PA, USA) *(EICS '09)*. Association for Computing Machinery, New York, NY, USA, 329–332. https://doi.org/10.1145/1570433.1570496

[4] C Atkinson and T Kuhne. 2003. Model-driven development: a metamodeling foundation. *IEEE Software* 20, 5 (2003), 36–41. https://doi.org/10.1109/MS.2003.1231149

[5] S Audet and M Okutomi. 2009. A user-friendly method to geometrically calibrate projector-camera systems. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*. IEEE, Miami, FL, USA, 47–54. https://doi.org/10.1109/CVPRW.2009.5204319

[6] Claudia P Ayala, Daniela Cruzes, Øyvind Hauge, and Reidar Conradi. 2011. Five Facts on the Adoption of Open Source Software. *IEEE Software* 28, 2 (mar 2011), 95–99. https://doi.org/10.1109/MS.2011.32

[7] Ronald M Baecker, Jonathan Grudin, William A S Buxton, and Saul Greenberg. 1995. A Historical and Intellectual Perspective. In *Human-Computer Interaction: Toward the Year 2000*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 35–47.

[8] Dana H Ballard. 1981. Generalizing the Hough transform to detect arbitrary shapes. *Pattern recognition* 13, 2 (1981), 111–122.

[9] Till Ballendat, Nicolai Marquardt, and Saul Greenberg. 2010. Proxemic Interaction: Designing for a Proximity and Orientation-Aware Environment. In *ACM International Conference on Interactive Tabletops and Surfaces (ITS '10)*. Association for Computing Machinery, New York, NY, USA, 121–130. https://doi.org/10.1145/1936652.1936676

[10] Jakob E Bardram. 2005. The Java Context Awareness Framework (JCAF) – a Service Infrastructure and Programming Framework for Context-Aware Applications. In *Proceedings of the 3rd International Conference on Pervasive Computing*. Springer-Verlag, Berlin, Heidelberg, 98–115. https://doi.org/10.1007/11428572_7

[11] Alexander Bazo and Florian Echtler. 2014. Phone Proxies: Effortless Content Sharing between Smartphones and Interactive Surfaces. In *Proceedings of the 2014 ACM SIGCHI Symposium on Engineering Interactive Computing Systems (EICS '14)*. Association for Computing Machinery, New York, NY, USA, 229–234. https://doi.org/10.1145/2607023.2610276

[12] Martin Becker and Andreas Schäfer. 2020. Variability Realization in UML/SysML Models. In *Proceedings of the 24th ACM Conference on Systems and Software Product Line (SPLC '20)*. Association for Computing Machinery, New York, NY, USA, 1. https://doi.org/10.1145/3382025.3414975

[13] Saifullahi Aminu Bello, Shangshu Yu, Cheng Wang, Jibril Muhmmad Adam, and Jonathan Li. 2020. Review: Deep learning on 3D point clouds. *Remote Sensing* 12, 11 (2020), 1729. https://doi.org/10.3390/rs12111729

[14] Dirk Bergmann. 1995. New approach for automatic surface reconstruction with coded light. In *Remote Sensing and Reconstruction for Three-Dimensional Objects and Scenes*, Toni F Schenk (Ed.), Vol. 2572. International Society for Optics and Photonics, SPIE, San Diego, CA, USA, 2–9. https://doi.org/10.1117/12.216931

[15] P J Besl and R C Jain. 1988. Segmentation through variable-order surface fitting. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 10, 2 (mar 1988), 167–192. https://doi.org/10.1109/34.3881

[16] Valentin Besnard, Frédéric Jouault, Matthias Brun, Ciprian Teodorov, Philippe Dhaussy, and Jérôme Delatour. 2020. Modular Deployment of UML Models for V and V Activities and Embedded Execution. In *Proceedings of the 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings (MODELS '20)*. Association for Computing Machinery, New York, NY, USA, 10. https://doi.org/10.1145/3417990.3419227

[17] Bir Bhanu, Sungkee Lee, Chih-Cheng Ho, and Tom Henderson. 1986. Range data processing: Representation of surfaces by edges. In *Proceedings of the eighth International Conference on Pattern Recognition*. IEEE Computer Society Press, IEEE, Paris, France, 236–238.

[18] Pranjal Kumar Blanco, Jose Luis and Rai. 2014. nanoflann: a C++ header-only fork of FLANN, a library for Nearest Neighbor with KD-trees. https://github.com/jlblancoc/nanoflann

[19] Bluetooth SIG. 2021. Specifications. https://www.bluetooth.com/specifications/specs/

[20] M Blum, Jost Tobias Springenberg, J Wülfing, and M Riedmiller. 2012. A learned feature descriptor for object recognition in RGB-D data. In *2012 IEEE International Conference on Robotics and Automation*. IEEE, Saint Paul, MN, USA, 1298–1303. https://doi.org/10.1109/ICRA.2012.6225188

[21] Liefeng Bo, Xiaofeng Ren, and Dieter Fox. 2013. Unsupervised Feature Learning for RGB-D Based Object Recognition. In *Experimental Robotics: The 13th International Symposium on Experimental Robotics*, Jaydev P Desai, Gregory Dudek, Oussama Khatib, and Vijay Kumar (Eds.). Springer International Publishing, Heidelberg, 387–402. https://doi.org/10.1007/978-3-319-00065-7_27

[22] A Boroomand, H Sekkati, M Lamm, D A Clausi, and A Wong. 2016. Saliency-guided projection geometric correction using a projector-camera system. In *2016 IEEE International Conference on Image Processing (ICIP)*. IEEE, Phoenix, AZ, USA, 2951–2955. https://doi.org/10.1109/ICIP.2016.7532900

[23] Sylvain Bougnoux. 1998. From projective to euclidean space under any practical situation, a criticism of self-calibration. In *Sixth International Conference on Computer Vision*. IEEE, IEEE, Bombay, India, 790–796.

[24] J Bouguet. 2015. Camera Calibration Toolbox for Matlab. http://www.vision.caltech.edu/bouguetj/calib_doc/

[25] Judy Bowen. 2015. Creating Models of Interactive Systems with the Support of Lightweight Reverse-Engineering Tools. In *Proceedings of the 7th ACM SIGCHI Symposium on Engineering Interactive Computing Systems* (Duisburg, Germany) *(EICS '15)*. Association for Computing Machinery, New York, NY, USA, 110–119. https://doi.org/10.1145/2774225.2774840

[26] Judy Bowen and Steve Reeves. 2017. Generating Obligations, Assertions and Tests from UI Models. *Proc. ACM Hum.-Comput. Interact.* 1, EICS, Article 5 (jun 2017), 18 pages. https://doi.org/10.1145/3095807

[27] Kai Breiner, Marc Seissler, Gerrit Meixner, Peter Forbrig, Ahmed Seffah, and Kerstin Klöckner. 2010. Pattern-Driven Engineering of Interactive Computing Systems (PEICS). In *Proceedings of the 2nd ACM SIGCHI Symposium on Engineering Interactive Computing Systems* (Berlin, Germany) *(EICS '10)*. Association for Computing Machinery, New York, NY, USA, 367–368. https://doi.org/10.1145/1822018.1822085

[28] Duane C Brown. 1966. Decentering distortion of lenses. *Photogrammetric Engineering and Remote Sensing* 1 (1966), 1–1.

[29] M Calonder, V Lepetit, M Ozuysal, T Trzcinski, C Strecha, and P Fua. 2012. BRIEF: Computing a Local Binary Descriptor Very Fast. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34, 7 (jul 2012), 1281–1298. https://doi.org/10.1109/TPAMI.2011.222

[30] Juming Cao, Slamu Wushour, Xinhuix Yao, N Li, Jin Liang, and Xinhe Liang. 2012. Sharp feature extraction in point clouds. *IET Image Processing* 6, 7 (2012), 863–869.

[31] Center for Open Science. 2021. Open science framework. https://osf.io/

[32] Chia-Yen Chen and Hsiang-Jen Chien. 2013. An incremental target-adapted strategy for active geometric calibration of projector-camera systems. *Sensors (Basel, Switzerland)* 13, 2 (feb 2013), 2664–2681. https://doi.org/10.3390/s130202664

[33] Christian Cherek, David Asselborn, Simon Voelker, and Jan Borchers. 2019. Off-surface tangibles: Exploring the design space of midair tangible interaction. In *Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, New York, NY, USA, 1–6. https://doi.org/10.1145/3290607.3312966

[34] H Chien, C Chuang, C Chen, and R Klette. 2016. When to use what feature? SIFT, SURF, ORB, or A-KAZE features for monocular visual odometry. In *2016 International Conference on Image and Vision Computing New Zealand*. IEEE, Palmerston North, New Zealand, 1–6. https://doi.org/10.1109/IVCNZ.2016.7804434

[35] Patrick Chiu, Chelhwon Kim, and Hideto Oda. 2018. Recognizing gestures on projected button widgets with an RGB-D camera using a CNN. In *Proceedings of the 2018 ACM International Conference on Interactive Surfaces and Spaces*. Association for Computing Machinery, New York, NY, USA, 369–374. https://doi.org/10.1145/3279778.3279907

[36] Jean Ho Chu, Paul Clifton, Daniel Harley, Jordanne Pavao, and Ali Mazalek. 2015. Mapping Place: supporting cultural learning through a Lukasa-inspired tangible tabletop museum exhibit. In *Proceedings of the Ninth International Conference on Tangible, Embedded, and Embodied Interaction*. Association for Computing Machinery, New York, NY, USA, 261–268. https://doi.org/10.1145/2677199.2680559

[37] Elizabeth F Churchill, David N Snowdon, and Alan J Munro. 2012. *Collaborative virtual environments: digital places and spaces for interaction*. Springer-Verlag London, London, United Kingdom. https://doi.org/10.1007/978-1-4471-0685-2

[38] Kevin Crowston, Kangning Wei, James Howison, and Andrea Wiggins. 2008. Free/Libre Open-Source Software Development: What We Know and What We Do Not Know. *ACM Comput. Surv.* 44, 2 (mar 2008). https://doi.org/10.1145/2089125.2089127

[39] Carolina Cruz-Neira, Daniel J Sandin, and Thomas A DeFanti. 1993. Surround-screen projection-based virtual reality: the design and implementation of the CAVE. In *Proceedings of the 20th annual conference on computer graphics and interactive techniques*. Association for Computing Machinery, New York, NY, USA, 135–142. https://doi.org/10.1145/166117.166134

[40] Leandro Flores Da Silva and Edson Oliveira. 2020. Evaluating Usefulness, Ease of Use and Usability of an UML-Based Software Product Line Tool. In *Proceedings of the 34th Brazilian Symposium on Software Engineering (SBES '20)*. Association for Computing Machinery, New York, NY, USA, 798–807. https://doi.org/10.1145/3422392.3422402

[41] Angela Dai, Matthias Niessner, Michael Zollhöfer, Shahram Izadi, and Christian Theobalt. 2017. BundleFusion: Real-time globally consistent 3D reconstruction using on-the-fly surface reintegration. *ACM Trans. Graph.* 36, 4 (may 2017), 18. https://doi.org/10.1145/3072959.3054739

[42] N Dalal and B Triggs. 2005. Histograms of oriented gradients for human detection. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Vol. 1. IEEE, San Diego, CA, USA, 886–893 vol. 1. https://doi.org/10.1109/CVPR.2005.177

[43] Ryan Anthony J de Belen, Tomasz Bednarz, and Dennis Del Favero. 2019. Integrating Mixed Reality and Internet of Things as an Assistive Technology for Elderly People Living in a Smart Home. In *The 17th International Conference on Virtual-Reality Continuum and Its Applications in Industry (VRCAI '19)*. Association for Computing Machinery, New York, NY, USA, 2. https://doi.org/10.1145/3359997.3365742

[44] Andrea Delgado, Ignacio de Guzman, Francisco Ruiz, and Mario Piattini. 2010. From BPMN business process models to SoaML service models: A transformation-driven approach. In *2nd International Conference on Software Technology and Engineering*, Vol. 1. IEEE, San Juan, PR, USA, 314–319. https://doi.org/10.1109/ICSTE.2010.5608855

[45] Michael den Bergh and Luc Van Gool. 2011. Combining RGB and ToF cameras for real-time 3D hand gesture interaction. In *IEEE Workshop on Applications of Computer Vision (WACV)*. IEEE, Kona, HI, USA, 66–72. https://doi.org/10.1109/WACV.2011.5711485

[46] Daljit Singh Dhillon and Venu Madhav Govindu. 2015. Geometric and radiometric estimation in a structured-light 3D scanner. *Machine Vision and Applications* 26, 2-3 (2015), 339–352.

[47] Anke Dittmar, Alfonso García Frey, and Sophie Dupuy-Chessa. 2012. What Can Model-Based UI Design Offer to End-User Software Engineering?. In *Proceedings of the 4th ACM SIGCHI Symposium on Engineering Interactive Computing Systems* (Copenhagen, Denmark) *(EICS '12)*. Association for Computing Machinery, New York, NY, USA, 189–194. https://doi.org/10.1145/2305484.2305515

[48] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. 2014. DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32 (ICML'14)*. JMLR.org, Beijing, China, I–647–I–655.

[49] J Drareni, S Roy, and P Sturm. 2009. Geometric video projector auto-calibration. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*. IEEE, Miami, FL, USA, 39–46. https://doi.org/10.1109/CVPRW.2009.5204317

[50] Florian Echtler. 2018. Surfacestreams: A content-agnostic streaming toolkit for interactive surfaces. In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology*. Association for Computing Machinery, New York, NY, USA, 10–12. https://doi.org/10.1145/3266037.3266085

[51] Florian Echtler and Maximilian Haussler. 2018. Open Source, Open Science, and the Replication Crisis in HCI. In *Extended Abstracts of the 2018 CHI Conference on Human Factors in Computing Systems (CHI EA '18)*. Association for Computing Machinery, New York, NY, USA, 1–8. https://doi.org/10.1145/3170427.3188395

[52] Florian Echtler and Gudrun Klinker. 2008. A multitouch software architecture. In *Proceedings of the 5th Nordic Conference on Human-Computer Interaction*, Vol. 358. Citeseer, Lund, Sweden, 463–466. https://doi.org/10.1145/1463160.1463220

[53] Florian Echtler and Gudrun Klinker. 2008. Tracking Mobile Phones on Interactive Tabletops. In *INFORMATIK 2008. Beherrschbare Systeme – dank Informatik. Band 1*, Heinz-Gerd Hegering, Axel Lehmann, Hans Jürgen Ohlbach, and Christian Scheideler (Eds.). Gesellschaft für Informatik e. V., Bonn, 285–290.

[54] Florian Echtler, Simon Nestler, Andreas Dippon, and Gudrun Klinker. 2009. Supporting Casual Interactions between Board Games on Public Tabletop Displays and Mobile Devices. *Personal Ubiquitous Comput.* 13, 8 (nov 2009), 609–617. https://doi.org/10.1007/s00779-009-0246-3

[55] Florian Echtler and Raphael Wimmer. 2014. The interactive dining table, or pass the weather widget, please. In *Proceedings of the 9th ACM International Conference on Interactive Tabletops and Surfaces*. Association for Computing Machinery, New York, NY, USA, 419–422. https://doi.org/10.1145/2669485.2669525

[56] Jochen Ehnes, Koichi Hirota, and Michitaka Hirose. 2004. Projected augmentation - augmented reality using rotatable video projectors. In *Proceedings of the 3rd IEEE/ACM International Symposium on Mixed and Augmented Reality*. IEEE Computer Society, USA, 26–35. https://doi.org/10.1109/ISMAR.2004.47

[57] Senem Ezgi Emgin, Amirreza Aghakhani, T. Metin Sezgin, and Cagatay Basdogan. 2019. HapTable: An interactive tabletop providing online haptic feedback for touch gestures. *IEEE Transactions on Visualization and Computer Graphics* 25, 9 (2019), 2749–2762. https://doi.org/10.1109/TVCG.2018.2855154

[58] Jürgen Engel. 2010. A Model- and Pattern-Based Approach for Development of User Interfaces of Interactive Systems. In *Proceedings of the 2nd ACM SIGCHI Symposium on Engineering Interactive Computing Systems* (Berlin, Germany) *(EICS '10)*. Association for Computing Machinery, New York, NY, USA, 337–340. https://doi.org/10.1145/1822018.1822075

[59] Martin Ester, Hans-Peter Kriegel, J"org Sander, Xiaowei Xu, and Others. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd*. AAAI Press, Portland, Oregon, USA, 226–231.

[60] Olivier Faugeras and OLIVIER AUTOR FAUGERAS. 1993. *Three-dimensional computer vision: a geometric viewpoint*. MIT press, Cambridge, MA, USA.

[61] Tarsha-Kurdi Fayez, Tania Landes, and Grussenmeyer Pierre. 2007. Hough-transform and extended RANSAC algorithms for automatic detection of 3d building roof planes from lidar data. In *ISPRS Workshop on Laser Scanning and SilviLaser*. Citeseer, Espoo, Finland, 407.

[62] M Fiala. 2005. Automatic Projector Calibration Using Self-Identifying Patterns. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Workshops*. IEEE, San Diego, CA, USA, 113. https://doi.org/10.1109/CVPR.2005.416

[63] M Fiala. 2005. Comparing ARTag and ARToolkit Plus fiducial marker systems. In *IEEE International Workshop on Haptic Audio Visual Environments and their Applications*. IEEE, Ottawa, ON, Canada, 6. https://doi.org/10.1109/HAVE.2005.1545669

[64] Martin A Fischler and Robert C Bolles. 1981. Random sample consensus: A paradigm for model titting with applications to image analysis and automated cartography. *Commun. ACM* 24, 6 (jun 1981), 381–395. https://doi.org/10.1145/358669.358692

[65] A W Fitzgibbon. 2001. Simultaneous linear estimation of multiple view geometry and lens distortion. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Vol. 1. IEEE, Kauai, HI, USA, 1. https://doi.org/10.1109/CVPR.2001.990465

[66] W B Frakes and S Isoda. 1994. Success Factors of Systematic Reuse. *IEEE Software* 20, 05 (sep 1994), 14–19. https://doi.org/10.1109/52.311045

[67] Robert B France, James M Bieman, Sai Pradeep Mandalaparty, Betty H C Cheng, and Adam Jensen. 2012. Repository for Model Driven Development (ReMoDD). In *2012 34th International Conference on Software Engineering (ICSE)*. IEEE, Zurich, Switzerland, 1471–1472. https://doi.org/10.1109/ICSE.2012.6227059

[68] Ingmar S Franke, Mathias Müller, Thomas Gründer, and Rainer Groh. 2014. FlexiWall: Interaction in-between 2D and 3D Interfaces. In *HCI International 2014 - Posters' Extended Abstracts*, Constantine Stephanidis (Ed.). Springer International Publishing, Cham, 415–420.

[69] Eisuke Fujinawa, Kenji Goto, Atsushi Irie, Songtao Wu, and Kuanhong Xu. 2019. Occlusion-aware hand posture based interaction on tabletop projector. In *Proceedings of the 2nd Annual ACM Symposium on User Interface Software and Technology*. Association for Computing Machinery, New Orleans, LA, USA, 113–115. https://doi.org/10.1145/3332167.3356890

[70] Shun Fujita, Naoki Yanagihara, Buntarou Shizuki, and Shin Takahashi. 2019. Ray-casting based interaction using an extended pull-out gesture for interactive tabletops. In *Proceedings of the 31st Australian Conference on Human-Computer-Interaction*. Association for Computing Machinery, New York, NY, USA, 546–549. https://doi.org/10.1145/3369457.3369528

[71] Alfonso García Frey, Eric Céret, Sophie Dupuy-Chessa, and Gaëlle Calvary. 2011. QUIMERA: A Quality Metamodel to Improve Design Rationale. In *Proceedings of the 3rd ACM SIGCHI Symposium on Engineering Interactive Computing Systems* (Pisa, Italy) *(EICS '11)*. Association for Computing Machinery, New York, NY, USA, 265–270. https://doi.org/10.1145/1996461.1996534

[72] Marcela Genero, Ana Fernández-Sáez, H Nelson, Geert Poels, and Mario Piattini. 2011. A Systematic Literature Review on the Quality of UML Models. *J. Database Manag.* 22 (2011), 46–70. https://doi.org/10.4018/jdm.2011070103

[73] Ross Girshick. 2015. Fast R-CNN. In *IEEE International Conference on Computer Vision*. IEEE, Santiago, Chile, 1440–1448. https://doi.org/10.1109/iccv.2015.169

[74] R Girshick, J Donahue, T Darrell, and J Malik. 2014. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, Columbus, OH, USA, 580–587. https://doi.org/10.1109/CVPR.2014.81

[75] Gene H Golub, Alan Hoffman, and Gilbert W Stewart. 1987. A generalization of the Eckart-Young-Mirsky matrix approximation theorem. *Linear Algebra and its applications* 88 (1987), 317–327.

[76] G H Golub and C F Van Loan. 1996. Matrix Computations.

[77] Jens Grubert and Matthias Kranz. 2017. HeadPhones: Ad Hoc Mobile Multi-Display Environments through Head Tracking. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, New York, NY, USA, 3966–3971. https://doi.org/10.1145/3025453.3025533

[78] Thomas Gründer, Dietrich Kammer, Marius Brade, and Rainer Groh. 2013. Towards a design space for elastic displays. In *CHI 2013 Workshop. Displays Take New Shape: An Agenda for Future Interactive Surfaces*. Association for Computing Machinery, New York, NY, USA, 1–4.

[79] A Grundhöfer and D Iwai. 2015. Robust, Error-Tolerant Photometric Projector Compensation. *IEEE Transactions on Image Processing* 24, 12 (dec 2015), 5086–5099. https://doi.org/10.1109/TIP.2015.2478388

[80] Yizheng Gu, Chun Yu, Zhipeng Li, Weiqi Li, Shuchang Xu, Xiaoying Wei, and Yuanchun Shi. 2019. Accurate and low-latency sensing of touch contact on any surface with finger-worn IMU sensor. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*. Association for Computing Machinery, New Orleans, LA, USA, 1059–1070. https://doi.org/10.1145/3332165.3347947

[81] Jan Gugenheimer, Enrico Rukzio, Pascal Knierim, and Julian Seifert. 2014. UbiBeam: An interactive projector-camera system for domestic deployment. In *Proceedings of the 9th ACM International Conference on Interactive Tabletops and Surfaces*. Association for Computing Machinery, New York, NY, USA, 305–310. https://doi.org/10.1145/2669485.2669537

[82] Stefan Gumhold, Xinlong Wang, Rob S MacLeod, and Others. 2001. Feature extraction from point clouds. In *Proceedings of the 10th International Meshing Roundtable*. Citeseer, Newport Beach, CA, USA, 293–305.

[83] Franziska Hannß, Mathias Müller, and Dietrich Kammer. 2021. Designing Interfaces for Elastic Displays Using Workshop and Prototyping Methods. In *Mensch und Computer 2021 - Workshopband*, Carolin Wienrich, Philipp Wintersberger, and Benjamin Weyers (Eds.). Gesellschaft für Informatik e.V., Bonn. https://doi.org/10.18420/muc2021-mci-ws09-391

[84] John Hardy and Jason Alexander. 2012. Toolkit Support for Interactive Projected Displays. In *Proceedings of the 11th International Conference on Mobile and Ubiquitous Multimedia (MUM '12)*. Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/2406367.2406419

[85] Richard Harper, Tom Rodden, Yvonne Rogers, and Abigail Sellen. 2020. HUMAN-COMPUTER INTERACTION IN THE YEAR 2020.

[86] Chris Harrison, Hrvoje Benko, and Andrew Wilson. 2011. OmniTouch : Wearable multi-touch interaction everywhere. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*. Association for Computing Machinery, New York, NY, USA, 441–450. https://doi.org/10.1145/2047196.2047255

[87] Richard Hartley and Andrew Zisserman. 2003. *Multiple view geometry in computer vision*. Cambridge university press, Cambridge, United Kingdom.

[88] Ammar Hattab and Gabriel Taubin. 2019. Rough carving of 3D models with spatial augmented reality. In *Proceedings of the ACM Symposium on Computational Fabrication (SCF '19)*. Association for Computing Machinery, New York, NY, USA, 10. https://doi.org/10.1145/3328939.3328998

[89] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. 2017. Mask R-CNN. arXiv:1703.06870 [cs.CV]

[90] K He, X Zhang, S Ren, and J Sun. 2015. Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 37, 9 (2015), 1904–1916. https://doi.org/10.1109/TPAMI.2015.2389824

[91] Yong He, Hongshan Yu, Xiaoyan Liu, Zhengeng Yang, Wei Sun, Yaonan Wang, Qiang Fu, Yanmei Zou, and Ajmal Mian. 2021. Deep Learning based 3D Segmentation: A Survey. arXiv:2103.05423 [cs.CV]

[92] Regina Hebig, Truong Ho Quang, Michel R V Chaudron, Gregorio Robles, and Miguel Angel Fernandez. 2016. The Quest for Open Source Projects That Use UML: Mining GitHub. In *Proceedings of the ACM/IEEE 19th International Conference on Model Driven Engineering Languages and Systems (MODELS '16)*. Association for Computing Machinery, New York, NY, USA, 173–183. https://doi.org/10.1145/2976767.2976778

[93] Felix Heinrichs, Daniel Schreiber, Jochen Huber, and Max Mühlhäuser. 2011. W5: A Meta-Model for Pen-and-Paper Interaction. In *Proceedings of the 3rd ACM SIGCHI Symposium on Engineering Interactive Computing Systems* (Pisa, Italy) *(EICS '11)*. Association for Computing Machinery, New York, NY, USA, 47–52. https://doi.org/10.1145/1996461.1996493

[94] Tobias Hesselmann, Niels Henze, and Susanne Boll. 2010. FlashLight: Optical Communication between Mobile Phones and Interactive Tabletops. In *ACM International Conference on Interactive Tabletops and Surfaces*. Association for Computing Machinery, New York, NY, USA, 135–138. https://doi.org/10.1145/1936652.1936679

[95] Anuruddha Hettiarachchi and Daniel Wigdor. 2016. Annexing Reality: Enabling Opportunistic Use of Everyday Objects as Tangible Proxies in Augmented Reality. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. Association for Computing Machinery, New York, NY, USA, 1957–1967. https://doi.org/10.1145/2858036.2858134

[96] Klaus Hildebrandt, Konrad Polthier, and Max Wardetzky. 2005. Smooth feature lines on surface meshes. In *Symposium on Geometry Processing*. Association for Computing Machinery, Vienna, Austria, 85–90.

[97] Annika Hinze, Judy Bowen, Yuting Wang, and Robi Malik. 2010. Model-Driven GUI & Interaction Design Using Emulation. In *Proceedings of the 2nd ACM SIGCHI Symposium on Engineering Interactive Computing Systems* (Berlin, Germany) *(EICS '10)*. Association for Computing Machinery, New York, NY, USA, 273–278. https://doi.org/10.1145/1822018.1822061

[98] Truong Ho-Quang, Regina Hebig, Gregorio Robles, Michel R V Chaudron, and Miguel Angel Fernandez. 2017. Practices and Perceptions of UML Use in Open Source Projects. In *Proceedings of the 39th International Conference on Software Engineering: Software Engineering in Practice Track (ICSE-SEIP '17)*. IEEE, Buenos Aires, Argentina, 203–212. https://doi.org/10.1109/ICSE-SEIP.2017.28

[99] Bingyao Huang and Haibin Ling. 2019. CompenNet++: End-to-End Full Projector Compensation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. IEEE, Seoul, Korea (South), 7164–7173.

[100] Bingyao Huang and Haibin Ling. 2019. End-To-End Projector Photometric Compensation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. IEE, Long Beach, CA, USA, 6810–6819.

[101] B Huang, Y Tang, S Ozdemir, and H Ling. 2020. A Fast and Flexible Projector-Camera Calibration System. *IEEE Transactions on Automation Science and Engineering* 18, 3 (2020), 1–15. https://doi.org/10.1109/TASE.2020.2994223

[102] A Hubeli and M Gross. 2001. Multiresolution feature extraction for unstructured meshes. In *Proceedings Visualization, 2001. VIS '01.* IEEE, San Diego, CA, USA, 287–294. https://doi.org/10.1109/VISUAL.2001.964523

[103] Shahram Izadi, David Kim, Otmar Hilliges, David Molyneaux, Richard Newcombe, Pushmeet Kohli, Jamie Shotton, Steve Hodges, Dustin Freeman, Andrew Davison, and Andrew Fitzgibbon. 2011. KinectFusion: Real-Time 3D Reconstruction and Interaction Using a Moving Depth Camera. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology (UIST '11)*. Association for Computing Machinery, New York, NY, USA, 559–568. https://doi.org/10.1145/2047196.2047270

[104] Haojian Jin, Christian Holz, and Kasper Hornbaek. 2015. Tracko: Ad-Hoc Mobile 3D Tracking Using Bluetooth Low Energy and Inaudible Signals for Cross-Device Interaction. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software and Technology (UIST '15)*. Association for Computing Machinery, New York, NY, USA, 147–156. https://doi.org/10.1145/2807442.2807475

[105] B Johanson, A Fox, and T Winograd. 2002. The Interactive Workspaces project: experiences with ubiquitous computing rooms. *IEEE Pervasive Computing* 1, 2 (apr 2002), 67–74. https://doi.org/10.1109/MPRV.2002.1012339

[106] Nikhita Joshi and Daniel Vogel. 2019. An evaluation of touch input at the edge of a table. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, New York, NY, USA, 2865–2874. https://doi.org/10.1145/3290605.3300476

[107] Martin Kaltenbrunner. 2009. ReacTIVision and TUIO: A Tangible Tabletop Toolkit. In *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces*. Association for Computing Machinery, New York, NY, USA, 9–16. https://doi.org/10.1145/1731903.1731906

[108] Martin Kaltenbrunner, Till Bovermann, Ross Bencina, and Enrico Costanza. 2005. TUIO: A protocol for table-top tangible user interfaces. In *Proceedings of the 6th international conference on Gesture in Human-Computer Interaction and Simulation*. Association for Computing Machinery, Berder Island, France, 1–5.

[109] Martin Kaltenbrunner and Florian Echtler. 2015. Shared Infrastructures for Tangible Tabletops and Interactive Surfaces. In *Proceedings of the 2015 International Conference on Interactive Tabletops and Surfaces*. Association for Computing Machinery, New York, NY, USA, 499–500. https://doi.org/10.1145/2817721.2835070

[110] Martin Kaltenbrunner and Florian Echtler. 2018. The TUIO 2.0 protocol: An abstraction framework for tangible interactive surfaces. *Proceedings of the ACM on Human-Computer Interaction* 2, EICS (jun 2018), 35. https://doi.org/10.1145/3229090

[111] M Kaltenbrunner, S Jorda, G Geiger, and M Alonso. 2006. The reacTable*: A collaborative musical instrument. In *Proceedings of the 15th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises*. IEEE, Manchester, UK, 406–411. https://doi.org/10.1109/WETICE.2006.68

[112] Dietrich Kammer, Mandy Keck, Mathias Müller, Thomas Gründer, and Rainer Groh. 2017. Exploring Big Data Landscapes with Elastic Displays. In *Mensch und Computer 2017 - Workshopband*, Manuel Burghardt, Raphael Wimmer, Christian Wolff, and Christa Womser-Hacker (Eds.). Gesellschaft für Informatik e.V., Regensburg. https://doi.org/10.18420/muc2017-ws08-0342

[113] Dietrich Kammer, Mathias Müller, Jan Wojdziak, and Ingmar S Franke. 2018. New Impressions in Interaction Design: A Task Taxonomy for Elastic Displays. *i-com* 17, 3 (2018), 247–256. https://doi.org/10.1515/icom-2018-0021

[114] Hirokazu Kato, Mark Billinghurst, I. Poupyrev, K. Imamoto, and K. Tachibana. 2000. Virtual object manipulation on a table-top AR environment. In *Proceedings of the IEEE and ACM International Symposium on Augmented Reality*. IEEE, New York, NY, USA, 111–119. https://doi.org/10.1109/ISAR.2000.880934

[115] Edwin M Knorr, Raymond T Ng, and Vladimir Tucakov. 2000. Distance-Based Outliers: Algorithms and Applications. *The VLDB Journal* 8, 3–4 (feb 2000), 237–253. https://doi.org/10.1007/s007780050006

[116] Nora Koch. 2006. Transformation Techniques in the Model-Driven Development Process of UWE. In *Workshop Proceedings of the Sixth International Conference on Web Engineering (ICWE '06)*. Association for Computing Machinery, New York, NY, USA, 3–es. https://doi.org/10.1145/1149993.1149997

[117] Bruce Kogut and Anca Metiu. 2001. Open-Source Software Development and Distributed Innovation. *Oxford Review of Economic Policy* 17, 2 (2001), 248–264. https://doi.org/10.1093/oxrep/17.2.248

[118] Hideki Koike, Yoshinori Kobayashi, and Yoichi Sato. 2001. Integrating paper and digital Information on EnhancedDesk: A method for realtime finger tracking on an augmented desk system. In *ACM Transactions on Computer-Human Interaction*, Vol. 8. Association for Computing Machinery, New York, NY, USA, 307–322. https://doi.org/10.1145/504704.504706

[119] Ryo Koizumi, Daisuke Kobayashi, and Naoki Hashimoto. 2015. Acceleration of dynamic spatial augmented reality system with a depth camera. In *International Conference on Cyberworlds*. IEEE, Visby, Sweden, 50–53. https://doi.org/10.1109/CW.2015.42

[120] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2017. ImageNet Classification with Deep Convolutional Neural Networks. *Commun. ACM* 60, 6 (2017), 84–90. https://doi.org/10.1145/3065386

[121] Y W Kuan, N O Ee, and L S Wei. 2019. Comparative Study of Intel R200, Kinect v2, and Primesense RGB-D Sensors Performance Outdoors. *IEEE Sensors Journal* 19, 19 (oct 2019), 8741–8750. https://doi.org/10.1109/JSEN.2019.2920976

[122] K Lai, L Bo, X Ren, and D Fox. 2011. A large-scale hierarchical multi-view RGB-D object dataset. In *2011 IEEE International Conference on Robotics and Automation*. IEEE, Shanghai, China, 1817–1824. https://doi.org/10.1109/ICRA.2011.5980382

[123] Kevin Lai, Liefeng Bo, Xiaofeng Ren, and Dieter Fox. 2012. Detection-based object labeling in 3D scenes. In *IEEE International Conference on Robotics and Automation*. IEE, Saint Paul, MN, USA, 1330–1337. https://doi.org/10.1109/ICRA.2012.6225316

[124] Chanhwi Lee, Jaehan Kim, Seoungbae Cho, Jinwoong Kim, Jisang Yoo, and Soonchul Kwon. 2020. Development of Real-Time Hand Gesture Recognition for Tabletop Holographic Display Interaction Using Azure Kinect. *Sensors* 20, 16 (2020). https://doi.org/10.3390/s20164566

[125] Clayton Lewis, Peter Polson, and Tim McKay. 1996. *Human-Computer Interface Design: success stories, emerging methods, and real-world context*. Morgan Kaufmann Publishers Inc., Boulder, CO, USA.

[126] F Li, H Sekkati, J Deglint, C Scharfenberger, M Lamm, D Clausi, J Zelek, and A Wong. 2017. Simultaneous Projector-Camera Self-Calibration for Three-Dimensional Reconstruction and Projection Mapping. *IEEE Transactions on Computational Imaging* 3, 1 (mar 2017), 74–83. https://doi.org/10.1109/TCI.2017.2652844

[127] T T Li, H Y Zhang, and J Geng. 2010. Geometric calibration of a camera-projector 3D imaging system. In *2010 25th International Conference of Image and Vision Computing New Zealand*. IEEE, Queenstown, New Zealand, 1–8. https://doi.org/10.1109/IVCNZ.2010.6148798

[128] R Lienhart and J Maydt. 2002. An extended set of Haar-like features for rapid object detection. In *Proceedings. International Conference on Image Processing*, Vol. 1. IEEE, Rochester, NY, USA, I–I. https://doi.org/10.1109/ICIP.2002.1038171

[129] Natan Linder and Pattie Maes. 2010. LuminAR: Portable Robotic Augmented Reality Interface Design and Prototype. In *Adjunct Proceedings of the 23nd Annual ACM Symposium on User Interface Software and Technology*. Association for Computing Machinery, New York, New York, USA, 395–396. https://doi.org/10.1145/1866218.1866237

[130] Lingni Ma, R Favier, Luat Do, E Bondarev, and P H N de With. 2013. Plane segmentation and decimation of point clouds for 3D environment reconstruction. In *10th Consumer Communications and Networking Conference*. IEEE, Las Vegas, NV, USA, 43–49. https://doi.org/10.1109/CCNC.2013.6488423

[131] Grzegorz Loniewski, Emilio Insfran, and Silvia Abrahão. 2010. A Systematic Review of the Use of Requirements Engineering Techniques in Model-Driven Development. In *Model Driven Engineering Languages and Systems*, Dorina C Petriu, Nicolas Rouquette, and Øystein Haugen (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 213–227.

[132] David Lowe. 2004. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision* 60 (2004), 91–. https://doi.org/10.1023/B:VISI.0000029664.99615.94

[133] D G Lowe. 1999. Object recognition from local scale-invariant features. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, Vol. 2. IEEE, Kerkyra, Greece, 1150–1157 vol.2. https://doi.org/10.1109/ICCV.1999.790410

[134] Kris Luyten, Davy Vanacken, Malte Weiss, Jan Borchers, and Miguel Nacenta. 2011. Second Workshop on Engineering Patterns for Multi-Touch Interfaces. In *Proceedings of the 3rd ACM SIGCHI Symposium on Engineering Interactive Computing Systems* (Pisa, Italy) *(EICS '11)*. Association for Computing Machinery, New York, NY, USA, 335–336. https://doi.org/10.1145/1996461.1996553

[135] Kalle Lyytinen and Youngjin Yoo. 2002. Ubiquitous computing. *Commun. ACM* 45, 12 (2002), 63–96.

[136] Ville Mäkelä, Mohamed Khamis, Lukas Mecke, Jobin James, Markku Turunen, and Florian Alt. 2018. Pocket Transfers: Interaction Techniques for Transferring Content from Situated Displays to Mobile Devices. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. Association for Computing Machinery, New York, NY, USA, 1–13. https://doi.org/10.1145/3173574.3173709

[137] Bhargava Manevarthe and Ramakrishnan Kalpathi. 2018. Depth-based movable display using projector kinect system. In *Proceedings of the 12th International Conference on Distributed Smart Cameras*. Association for Computing Machinery, New York, NY, USA, 6. https://doi.org/10.1145/3243394.3243691

[138] Ivan Martynov, Joni-Kristian Kamarainen, and Lasse Lensu. 2011. Projector Calibration by "Inverse Camera Calibration". In *Proceedings of the 17th Scandinavian Conference on Image Analysis (SCIA'11)*. Springer-Verlag, Berlin, Heidelberg, 536–544.

[139] Stephen J Maybank and Olivier D Faugeras. 1992. A theory of self-calibration of a moving camera. *International journal of computer vision* 8, 2 (1992), 123–151.

[140] James McGlade, Luke Wallace, Bryan Hally, Andrew White, Karin Reinke, and Simon Jones. 2020. An early exploration of the use of the Microsoft Azure Kinect for estimation of urban tree Diameter at Breast Height. *Remote Sensing Letters* 11, 11 (2020), 963–972. https://doi.org/10.1080/2150704X.2020.1802528

[141] William Marshall McKeeman. 1962. Algorithm 145: Adaptive numerical integration by Simpson's rule. *Commun. ACM* 5, 12 (1962), 604.

[142] J D Mejia-Trujillo, Y J Castano-Pino, A Navarro, J D Arango-Paredes, D Rincón, J Valderrama, B Muñoz, and J L Orozco. 2019. Kinect™ and Intel RealSense™ D435 comparison: a preliminary study for motion analysis. In *Proceedings of the 2019 IEEE International Conference on E-health Networking, Application Services*. IEEE, Bogota, DC, Colombia, 1–4. https://doi.org/10.1109/HealthCom46333.2019.9009433

[143] Microsoft. 2019. A cross platform (Linux and Windows) user mode SDK to read data from your Azure Kinect device. https://github.com/microsoft/Azure-Kinect-Sensor-SDK

[144] Nicholas Ross Milton. 2007. *Knowledge acquisition in practice: a step-by-step guide*. Springer, London. 176 pages. https://doi.org/10.1007/978-1-84628-861-6

[145] Audris Mockus, Roy T Fielding, and James D Herbsleb. 2002. Two Case Studies of Open Source Software Development: Apache and Mozilla. *ACM Trans. Softw. Eng. Methodol.* 11, 3 (jul 2002), 309–346. https://doi.org/10.1145/567793.567795

[146] Jorge J Moré. 1978. The Levenberg-Marquardt algorithm: implementation and theory. In *Numerical analysis*. Springer, Berlin, Heidelberg, 105–116.

[147] D Moreno and G Taubin. 2012. Simple, Accurate, and Robust Projector-Camera Calibration. In *2012 Second International Conference on 3D Imaging, Modeling, Processing, Visualization Transmission*. IEEE, Zurich, Switzerland, 464–471. https://doi.org/10.1109/3DIMPVT.2012.77

[148] Everett Mondliwethu Mthunzi and Florian Echtler. 2021. Artefact: A UML-based framework for model-driven development of interactive surface prototypes. In *Interactive Surfaces and Spaces (ISS '21 Companion)*. Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/3447932.3490523

[149] Everett Mondliwethu Mthunzi, Christopher Getschmann, and Florian Echtler. 2021. Fast 3D point-cloud segmentation for interactive surfaces. In *Interactive Surfaces and Spaces (ISS '21 Companion)*. Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/3447932.3491141

[150] Mathias Müller, Thomas Gründer, and Rainer Groh. [n.d.]. No Title. In *xCoAx 2015: Proceedings of the Third Conference on Computation, Communication, Aesthetics and X*.

[151] Mathias Müller, Mandy Keck, Thomas Gründer, Natalie Hube, and Rainer Groh. 2017. A zoomable product browser for elastic displays. In *xCoAx 2017: Proceedings of the Fifth Conference on Computation, Communication, Aesthetics and X*. 127–136.

[152] Mathias Müller, Anja Knöfel, Thomas Gründer, Ingmar Franke, and Rainer Groh. 2014. FlexiWall: Exploring Layered Data with Elastic Displays. In *Proceedings of the Ninth ACM International Conference on Interactive Tabletops and Surfaces (ITS '14)*. Association for Computing Machinery, New York, NY, USA, 439–442. https://doi.org/10.1145/2669485.2669529

[153] Mathias Müller, Erik Lier, Rainer Groh, and Franziska Hannß. 2020. A Tangible Concept for Layered Map Visualizations: Supporting on-site Civil Engineering Construction Consultations using Elastic Displays. In *Mensch und Computer 2020 - Workshopband, Magdebug, Germany, September 6-9, 2020*, Christian Hansen, Andreas N"urnberger, and Bernhard Preim (Eds.). Gesellschaft f"ur Informatik e.V., Bonn. https://doi.org/10.18420/muc2020-ws121-368

[154] Mathias Müller, Erik Lier, and Thomas Gründer. 2018. Zoomable User Interfaces für Elastic Displays. In *Mensch und Computer 2018 - Workshopband*, Raimund Dachselt and Gerhard Weber (Eds.). Gesellschaft für Informatik e.V., Bonn. https://doi.org/10.18420/muc2018-ws05-0502

[155] Sundar Murugappan, Vinayak, Niklas Elmqvist, and Karthik Ramani. 2012. Extended Multitouch: Recovering touch posture and differentiating users using a depth camera. In *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology (UIST '12)*. Association for Computing Machinery, New York, NY, USA, 487–496. https://doi.org/10.1145/2380116.2380177

[156] M Muthugnanambika and S Padmavathi. 2017. Feature detection for color images using SURF. In *2017 4th International Conference on Advanced Computing and Communication Systems*. IEEE, Coimbatore, India, 1–4. https://doi.org/10.1109/ICACCS.2017.8014572

[157] Shree K Nayar, Harish Peri, Michael D Grossberg, and Peter N Belhumeur. 2003. A projection system with radiometric compensation for screen imperfections. In *ICCV workshop on projector-camera systems*, Vol. 3. IEEE, New York, NY, USA, 1–1.

[158] A Nguyen and B Le. 2013. 3D point cloud segmentation: A survey. In *6th IEEE Conference on Robotics, Automation and Mechatronics*. IEEE, Manila, Philippines, 225–230. https://doi.org/10.1109/RAM.2013.6758588

[159] Object Management Group. 2017. A specification defining a graphical language for visualizing, specifying, constructing, and documenting the artifacts of distributed object systems. https://www.omg.org/spec/UML/About-UML/

[160] Sven Oesau, Florent Lafarge, and Pierre Alliez. 2016. Planar shape detection and regularization in tandem. In *Proceedings of the 37th Annual Conference of the European Association for Computer Graphics*. Wiley Online Library, Lisbon, Portuga, 203–215.

[161] Harri Ojanen. 1999. Automatic correction of lens distortion by using digital image processing. *Rutgers University, Dept. of Mathematics technical report* 1 (1999), 1–1.

[162] Opencv.org. 2014. The OpenCV Reference Manual. https://docs.opencv.org/4.5.3/

[163] A R Orghidan, C M Gordan, D A Vlaicu, and B J Salvi. 2012. Projector-camera calibration for 3D reconstruction using vanishing points. In *2012 International Conference on 3D Imaging (IC3D)*. IEEE, Liège, Belgium, 1–6. https://doi.org/10.1109/IC3D.2012.6615143

[164] C P Papageorgiou, M Oren, and T Poggio. 1998. A general framework for object detection. In *Sixth International Conference on Computer Vision (IEEE Cat. No.98CH36271)*. IEEE, Bombay, India, 555–562. https://doi.org/10.1109/ICCV.1998.710772

[165] Yoon Jung Park, Hyocheol Ro, Jung-Hyun Byun, and Tack-Don Han. 2019. Adaptive Projection Augmented Reality with Object Recognition Based on Deep Learning. In *Proceedings of the 24th International Conference on Intelligent User Interfaces: Companion*. Association for Computing Machinery, New York, NY, USA, 51–52. https://doi.org/10.1145/3308557.3308678

[166] Mark Pauly, Richard Keiser, and Markus Gross. 2003. Multi-scale feature extraction on point-sampled surfaces. In *Computer graphics forum*. Blackwell Publishing, Inc and Eurographics Association, Oxford, UK, 281–289.

[167] Wozniak Pawel, Nitesh Goyal, Przemyslaw Kucharski, Lars Lischke, Sven Mayer, and Morten Fjeld. 2016. RAMPARTS: Supporting Sensemaking with Spatially-Aware Mobile Interactions. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, New York, NY, USA, 2447–2460. https://doi.org/10.1145/2858036.2858491

[168] Joshua Peschke, Fabian Göbel, Thomas Gründer, Mandy Keck, Dietrich Kammer, and Rainer Groh. 2012. DepthTouch: An Elastic Surface for Tangible Computing. In *Proceedings of the International Working Conference on Advanced Visual Interfaces (AVI '12)*. Association for Computing Machinery, New York, NY, USA, 770–771. https://doi.org/10.1145/2254556.2254706

[169] Larry Press. 1993. Before the Altair: The History of Personal Computing. *Commun. ACM* 36, 9 (sep 1993), 27–33. https://doi.org/10.1145/162685.162697

[170] Tahir Rabbani, Frank Van Den Heuvel, and George Vosselmann. 2006. Segmentation of point clouds using smoothness constraint. *International archives of photogrammetry, remote sensing and spatial information sciences* 36, 5 (2006), 248–253.

[171] Roman Rädle, Hans-Christian Jetter, Jonathan Fischer, Inti Gabriel, Clemens N Klokmose, Harald Reiterer, and Christian Holz. 2018. PolarTrack: Optical Outside-In Device Tracking That Exploits Display Polarization. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, New York, NY, USA, 1–9. https://doi.org/10.1145/3173574.3174071

[172] Roman Rädle, Hans-Christian Jetter, Nicolai Marquardt, Harald Reiterer, and Yvonne Rogers. 2014. HuddleLamp: Spatially-Aware Mobile Displays for Ad-Hoc Around-the-Table Collaboration. In *Proceedings of the Ninth ACM International Conference on Interactive Tabletops and Surfaces*. Association for Computing Machinery, New York, NY, USA, 45–54. https://doi.org/10.1145/2669485.2669500

[173] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. 2015. You Only Look Once: Unified, Real-Time Object Detection. arXiv:1506.02640 [cs.CV]

[174] J Redmon and A Farhadi. 2017. YOLO9000: Better, Faster, Stronger. In *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, Honolulu, HI, USA, 6517–6525. https://doi.org/10.1109/CVPR.2017.690

[175] Jun Rekimoto and Masanori Saitoh. 1999. Augmented surfaces: A spatially continuous work space for hybrid computing environments. *Proceedings of the SIGCHI conference on Human Factors in Computing Systems* (1999), 378–385. https://doi.org/10.1145/302979.303113

[176] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. arXiv:1506.01497 [cs.CV]

[177] Brett Ridel, Patrick Reuter, Jeremy Laviole, Nicolas Mellado, Nadine Couture, and Xavier Granier. 2014. The revealing flashlight: Interactive spatial augmented reality for detail exploration of cultural heritage artifacts. *Journal on Computing and Cultural Heritage* 7, 2 (jun 2014), 18. https://doi.org/10.1145/2611376

[178] Yvonne Rogers and Tom Rodden. 2003. Configuring Spaces and Surfaces to Support Collaborative Interactions. In *Public and Situated Displays: Social and Interactional Aspects of Shared Display Technologies*. Springer Netherlands, Dordrecht, 45–79. https://doi.org/10.1007/978-94-017-2813-3_3

[179] Stephanie M Roldan. 2017. Object Recognition in Mental Representations: Directions for Exploring Diagnostic Features through Visual Mental Imagery. *Frontiers in Psychology* 8 (2017), 833. https://doi.org/10.3389/fpsyg.2017.00833

[180] Radu Bogdan Rusu. 2010. Semantic 3d object maps for everyday manipulation in human living environments. *KI-K"unstliche Intelligenz* 24, 4 (2010), 345–348.

[181] Radu Bogdan Rusu, Zoltan Csaba Marton, Nico Blodow, Mihai Dolha, and Michael Beetz. 2008. Towards 3D point cloud based object maps for household environments. *Robotics and Autonomous Systems* 56, 11 (2008), 927–941.

[182] Daniel Salber, Anind K. Dey, and Gregory D. Abowd. 1999. The context toolkit: Aiding the development of context-enabled applications. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*. ACM, Pittsburgh, Pennsylvania, USA, 434–441. https://doi.org/10.1145/302979.303126

[183] Kai San Choi, Edmund Y Lam, and Kenneth K Y Wong. 2006. Automatic source camera identification using the intrinsic lens radial distortion. *Optics express* 14, 24 (2006), 11551–11565.

[184] R G Sargent. 2010. Verification and validation of simulation models. In *Proceedings of the 2010 Winter Simulation Conference*. IEEE, Baltimore, MD, USA, 166–183. https://doi.org/10.1109/WSC.2010.5679166

[185] Dominik Schmidt, Fadi Chehimi, Enrico Rukzio, and Hans Gellersen. 2010. PhoneTouch: A Technique for Direct Phone Interaction on Surfaces. In *Proceedings of the 23nd Annual ACM Symposium on User Interface Software and Technology*. Association for Computing Machinery, New York, NY, USA, 13–16. https://doi.org/10.1145/1866029.1866034

[186] Dominik Schmidt, Julian Seifert, Enrico Rukzio, and Hans Gellersen. 2012. A Cross-Device Interaction Style for Mobiles and Surfaces. In *Proceedings of the Designing Interactive Systems Conference (DIS '12)*. Association for Computing Machinery, New York, NY, USA, 318–327. https://doi.org/10.1145/2317956.2318005

[187] Ruwen Schnabel, Roland Wahl, and Reinhard Klein. 2007. Efficient RANSAC for point-cloud shape detection. In *Computer graphics forum*. Wiley Online Library, Blackwell Publishing Ltd, Oxford, UK, 214–226.

[188] B Selic. 2003. The pragmatics of model-driven development. *IEEE Software* 20, 5 (2003), 19–25. https://doi.org/10.1109/MS.2003.1231146

[189] Orit Shaer, Guy Kol, Megan Strait, Chloe Fan, Catherine Grevet, and Sarah Elfenbein. 2010. G-Nome Surfer: A Tabletop Interface for Collaborative Exploration of Genomic Data. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, New York, NY, USA, 1427–1436. https://doi.org/10.1145/1753326.1753539

[190] M Shahpaski, L R Sapaico, G Chevassus, and S Süsstrunk. 2017. Simultaneous Geometric and Radiometric Calibration of a Projector-Camera Pair. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, Honolulu, HI, USA, 3596–3604. https://doi.org/10.1109/CVPR.2017.383

[191] Ling Shao, Ziyun Cai, Li Liu, and Ke Lu. 2017. Performance Evaluation of Deep Feature Learning for RGB-D Image/Video Classification. *Inf. Sci.* 385, C (apr 2017), 266–283. https://doi.org/10.1016/j.ins.2017.01.013

[192] Gaurav Sinha, Rahul Shahi, and Mani Shankar. 2010. Human Computer Interaction. In *3rd international conference on emerging trends in engineering and technology*. IEEE, USA, 1–4. https://doi.org/10.1109/ICETET.2010.85

[193] R Matt Steele and Christopher Jaynes. 2006. Overconstrained linear estimation of radial distortion and multi-view geometry. In *European Conference on Computer Vision*. Springer, Graz, Austria, 253–264.

[194] Norbert A Streitz, Jorg Geissler, Torsten Holmer, Shin'ichi Konomi, Christian Muller-Tomfelde, Wolfgang Reischl, Petra Rexroth, Peter Seitz, and Ralf Steinmetz. 1999. I-LAND: An Interactive Landscape for Creativity and Innovation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '99)*. Association for Computing Machinery, New York, NY, USA, 120–127. https://doi.org/10.1145/302979.303010

[195] Simon Stusak and Markus Teufel. 2014. Projection augmented physical visualizations. In *Proceedings of the 2nd ACM Symposium on Spatial User Interaction*. Association for Computing Machinery, New York, NY, USA, 145. https://doi.org/10.1145/2659766.2661210

[196] Kang Tong, Yiquan Wu, and Fei Zhou. 2020. Recent advances in small object detection based on deep learning: A review. *Image and Vision Computing* 97 (2020), 103910.

[197] Damiano Torre, Yvan Labiche, Marcela Genero, Maged Elaasar, and Claudio Menghi. 2020. UML Consistency Rules: A Case Study with Open-Source UML Models. In *Proceedings of the 8th International Conference on Formal Methods in Software Engineering (FormaliSE '20)*. Association for Computing Machinery, New York, NY, USA, 130–140. https://doi.org/10.1145/3372020.3391554

[198] Alexander J B Trevor, Suat Gedikli, Radu B Rusu, and Henrik I Christensen. 2013. Efficient organized point cloud segmentation with connected components. *Semantic Perception Mapping and Exploration* 1 (2013), 6.

[199] R Tsai. 1987. A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses. *IEEE Journal on Robotics and Automation* 3, 4 (aug 1987), 323–344. https://doi.org/10.1109/JRA.1987.1087109

[200] Jessica Turner, Judy Bowen, and Steve Reeves. 2020. Model-Based Testing of Interactive Systems Using Interaction Sequences. *Proc. ACM Hum.-Comput. Interact.* 4, EICS, Article 85 (jun 2020), 37 pages. https://doi.org/10.1145/3397873

[201] Brygg Ullmer and Hiroshi Ishii. 1997. The metaDESK: Models and prototypes for tangible user interfaces. In *Symposium on user interface software and technology*. Association for Computing Machinery, Banff, Alberta, Canada, 223–232. https://doi.org/10.1145/263407.263551

[202] Brygg Ullmer and Hiroshi Ishii. 1998. The metaDESK: Models and prototypes for tangible user interfaces. *Proceedings of the 10th annual ACM Symposium on User Interface Software and Technology* 8 (1998), 223–232. https://doi.org/10.1145/263407.263551

[203] Ultralytics. 2021. YOLOv5: A family of object detection architectures and models pretrained on the COCO dataset. https://github.com/ultralytics/yolov5

[204] Baris Unver, Alexander Thayer, and Svetlana Yarosh. 2017. Asymmetric device interaction with projector-camera systems. In *Proceedings of the 2017 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2017 ACM International Symposium on Wearable Computers (UbiComp '17)*. Association for Computing Machinery, New York, NY, USA, 301–304. https://doi.org/10.1145/3123024.3123192

[205] R Usamentiaga, D F Garcia, C Ibarra-Castanedo, and X Maldague. 2017. Highly accurate geometric calibration for infrared cameras using inexpensive calibration targets. *Measurement* 112 (2017), 105–116.

[206] Jouni Vepsäläinen, Antonella Di Rienzo, Matti Nelimarkka, Jouni A Ojala, Petri Savolainen, Kai Kuikkaniemi, Sasu Tarkoma, and Giulio Jacucci. 2015. Personal Device as a Controller for Interactive Surfaces: Usability and Utility of Different Connection Methods. In *Proceedings of the 2015 International Conference on Interactive Tabletops and Surfaces*. Association for Computing Machinery, New York, NY, USA, 201–204. https://doi.org/10.1145/2817721.2817745

[207] Stephen Voida, Mark Podlaseck, Rick Kjeldsen, and Claudio Pinhanez. 2005. A study on the manipulation of 2D-objects in a projector-camera -based augmented reality environment. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, New York, NY, USA, 611–620. https://doi.org/10.1145/1054972.1055056

[208] Chat Wacharamanotham, Lukas Eisenring, Steve Haroz, and Florian Echtler. 2020. Transparency of CHI Research Artifacts: Results of a Self-Reported Survey. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems (CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–14. https://doi.org/10.1145/3313831.3376448

[209] R Wang, X Lai, and W Hou. 2011. Study on Edge Detection of LIDAR Point Cloud. In *2011 International Conference on Intelligent Computation and Bio-Medical Instrumentation*. IEEE, Wuhan, China, 71–73. https://doi.org/10.1109/ICBMI.2011.82

[210] Wenguan Wang, Qiuxia Lai, Huazhu Fu, Jianbing Shen, Haibin Ling, and Ruigang Yang. 2021. Salient Object Detection in the Deep Learning Era: An In-depth Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence* Early Acce (2021), 1–1. https://doi.org/10.1109/TPAMI.2021.3051099

[211] Y Wang, D Ewert, D Schilberg, and S Jeschke. 2013. Edge extraction by merging 3D point cloud and 2D image data. In *10th International Conference and Expo on Emerging Technologies for a Smarter World*. IEEE, Melville, NY, USA, 1–6. https://doi.org/10.1109/CEWIT.2013.6713743

[212] Kouki Watanabe and Alexander G Belyaev. 2001. Detection of salient curvature features on polygonal surfaces. In *Computer Graphics Forum*. Blackwell Publishers Ltd, Oxford, UK and Boston, USA, 385–392.

[213] Christopher Weber, Stefanie Hahmann, and Hans Hagen. 2011. Methods for feature detection in point clouds. In *Visualization of Large and Unstructured Data Sets-Applications in Geospatial Planning, Modeling and Engineering*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, Kaiserslautern, Germany, 1.

[214] Mark Weiser. 1994. Ubiquitous Computing (Abstract). In *Proceedings of the 22nd Annual ACM Computer Science Conference on Scaling up : Meeting the Challenge of Complexity in Real-World Computing Applications: Meeting the Challenge of Complexity in Real-World Computing Applications (CSC '94)*. Association for Computing Machinery, New York, NY, USA, 418. https://doi.org/10.1145/197530.197680

[215] Pierre Wellner. 1991. The DigitalDesk Calculator: Tangible Manipulation on a Desk Top Display. In *Proceedings of the 4th Annual ACM Symposium on User Interface Software and Technology (UIST '91)*. Association for Computing Machinery, New York, NY, USA, 27–33. https://doi.org/10.1145/120782.120785

[216] S Willi and A Grundhöfer. 2017. Robust Geometric Self-Calibration of Generic Multi-Projector Camera Systems. In *2017 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. IEEE, Nantes, France, 42–51. https://doi.org/10.1109/ISMAR.2017.21

[217] Andrew D. Wilson. 2005. PlayAnywhere: A Compact Interactive Tabletop Projection-Vision System. In *Proceedings of the Annual ACM Symposium on User Interface Software and Technology*. Association for Computing Machinery, New York, NY, USA, 83–92. https://doi.org/10.1145/1095034.1095047

[218] Andrew D Wilson. 2009. Simulating Grasping Behavior on an Imaging Interactive Surface. In *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces*. Association for Computing Machinery, New York, NY, USA, 125–132. https://doi.org/10.1145/1731903.1731929

[219] Andrew D. Wilson. 2010. Using a Depth Camera as a Touch Sensor. In *ACM International Conference on Interactive Tabletops and Surfaces, ITS 2010*. Association for Computing Machinery, New York, NY, USA, 69–72. https://doi.org/10.1145/1936652.1936665

[220] Andrew D. Wilson and Hrvoje Benko. 2010. Combining multiple depth cameras and projectors for interactions on, above, and between surfaces. In *23rd ACM Symposium on User Interface Software and Technology*. Association for Computing Machinery, New York, NY, USA, 273–282. https://doi.org/10.1145/1866029.1866073

[221] Andrew D Wilson and Hrvoje Benko. 2016. Projected augmented reality with the RoomAlive toolkit. In *Proceedings of the 2016 ACM International Conference on Interactive Surfaces and Spaces (Wilson2016)*. Association for Computing Machinery, New York, NY, USA, 517–520. https://doi.org/10.1145/2992154.2996362

[222] Andrew D Wilson and Hrvoje Benko. 2016. Projected Augmented Reality with the RoomAlive Toolkit. In *Proceedings of the 2016 ACM International Conference on Interactive Surfaces and Spaces (ISS '16)*. Association for Computing Machinery, New York, NY, USA, 517–520. https://doi.org/10.1145/2992154.2996362

[223] Andrew D. Wilson, Hrvoje Benko, Shahram Izadi, and Otmar Hilliges. 2012. Steerable augmented reality with the Beamatron. In *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology*. Association for Computing Machinery, New York, NY, USA, 413–422. https://doi.org/10.1145/2380116.2380169

[224] Andrew D Wilson and Raman Sarin. 2007. BlueTable: Connecting Wireless Mobile Devices on Interactive Surfaces Using Vision-Based Handshaking. In *Proceedings of Graphics Interface 2007 (GI '07)*. Association for Computing Machinery, New York, NY, USA, 119–125. https://doi.org/10.1145/1268517.1268539

[225] Benjamin Wingert, Isabel Schöllhorn, and Matthias Bues. 2017. Prodesk: An interactive ubiquitous desktop surface. In *Proceedings of the 2017 ACM International Conference on Interactive Surfaces and Spaces*. Association for Computing Machinery, New York, NY, USA, 366–371. https://doi.org/10.1145/3132272.3135078

[226] Christian Winkler, Julian Seifert, David Dobbelstein, and Enrico Rukzio. 2014. Pervasive information through constant personal projection: The ambient mobile pervasive display (AMP-D). In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, New York, NY, USA, 4117–4126. https://doi.org/10.1145/2556288.2557365

[227] Qin Wu, Jiayuan Wang, Sirui Wang, Tong Su, and Chenmei Yu. 2019. MagicPAPER: Tabletop interactive projection device based on tangible interaction. In *Proceedings of the 2018 ACM International Conference on Interactive Surfaces and Spaces*. Association for Computing Machinery, Los Angeles, CA, USA, 2. https://doi.org/10.1145/3306214.3338575

[228] Robert Xiao, Chris Harrison, and Scott E. Hudson. 2013. WorldKit: Rapid and easy creation of Ad-hoc interactive applications on everyday surfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, New York, NY, USA, 879–888. https://doi.org/10.1145/2470654.2466113

[229] Robert Xiao, Scott Hudson, and Chris Harrison. 2017. Supporting Responsive Cohabitation Between Virtual Interfaces and Physical Objects on Everyday Surfaces. *Proceedings of the ACM on Human-Computer Interaction* 1, EICS (jun 2017), 17. https://doi.org/10.1145/3095814

[230] Xuhai Xu, Chun Yu, Yuntao Wang, and Yuanchun Shi. 2020. Recognizing unintentional touch on interactive tabletop. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 4, 1 (mar 2020), 24. https://doi.org/10.1145/3381011

[231] S Yamazaki, M Mochimaru, and T Kanade. 2011. Simultaneous self-calibration of a projector and a camera using structured light. In *CVPR 2011 WORKSHOPS*. IEEE, Colorado Springs, CO, USA, 60–67. https://doi.org/10.1109/CVPRW.2011.5981781

[232] L Yang, J Normand, and G Moreau. 2015. Local Geometric Consensus: A General Purpose Point Pattern-Based Tracking Algorithm. *IEEE Transactions on Visualization and Computer Graphics* 21, 11 (nov 2015), 1299–1308. https://doi.org/10.1109/TVCG.2015.2459897

[233] L Yang, J Normand, and G Moreau. 2016. Practical and Precise Projector-Camera Calibration. In *2016 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. IEEE, Merida, Mexico, 63–70. https://doi.org/10.1109/ISMAR.2016.22

[234] Jieping Ye. 2007. Least Squares Linear Discriminant Analysis. In *Proceedings of the 24th International Conference on Machine Learning*. Association for Computing Machinery, New York, NY, USA, 1087–1093. https://doi.org/10.1145/1273496.1273633

[235] Kaito Yoshino and Saeko Matsuura. 2020. Requirements Traceability Management Support Tool for UML Models. In *Proceedings of the 2020 9th International Conference on Software and Computer Applications (ICSCA 2020)*. Association for Computing Machinery, New York, NY, USA, 163–166. https://doi.org/10.1145/3384544.3384586

[236] Kaichao You, Yong Liu, Jianmin Wang, and Mingsheng Long. 2021. LogME: Practical Assessment of Pre-trained Models for Transfer Learning. In *Proceedings of the 38th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 139)*, Marina Meila and Tong Zhang (Eds.). PMLR, 12133–12143. https://proceedings.mlr.press/v139/you21b.html

[237] Zhengyou Zhang. 2000. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22, 11 (nov 2000), 1330–1334. https://doi.org/10.1109/34.888718

[238] Zhengyou Zhang. 2008. A Flexible New Technique for Camera Calibration. http://research.microsoft.com/~zhanghttp://research.microsoft.com/~zhang

[239] Jiehan Zhou, M Rautiainen, M Ylianttila, M Foulonneau, and P Blandin. 2008. Metamodeling for Community Coordinated Multimedia and Experience on Metamodel-Driven Content Annotation Service Prototype. In *IEEE Congress on Services Part II*. IEEE, Beijing, China, 88–95. https://doi.org/10.1109/SERVICES-2.2008.31

# A OPEN-SOURCE REPOSITORIES

## A.1 Tools

https://github.com/edisonslightbulbs/Bluetooth-RFCOMM-Linux
https://github.com/edisonslightbulbs/Bluetooth-RFCOMM-Android
https://github.com/edisonslightbulbs/YOLOv5-cpp
https://github.com/edisonslightbulbs/Point
https://github.com/edisonslightbulbs/SVD
https://github.com/edisonslightbulbs/viewer
https://github.com/edisonslightbulbs/segment
https://github.com/edisonslightbulbs/outliers
https://github.com/edisonslightbulbs/procam-calibration
https://github.com/edisonslightbulbs/edge-detection

## A.2 Tutorials

https://github.com/edisonslightbulbs/K4a
https://github.com/edisonslightbulbs/CV-K4a

## A.3 Algorithms

https://github.com/edisonslightbulbs/KMEANS
https://github.com/edisonslightbulbs/KNN
https://github.com/edisonslightbulbs/DBSCAN
https://github.com/edisonslightbulbs/EPF
https://github.com/edisonslightbulbs/JCT
https://github.com/edisonslightbulbs/ECE

## A.4 Android APKs

https://github.com/edisonslightbulbs/APK-traceless
https://github.com/edisonslightbulbs/APK-gravity-sensing
https://github.com/edisonslightbulbs/APK-orientation-sensing
https://github.com/edisonslightbulbs/APK-gyroscope-sensing
https://github.com/edisonslightbulbs/APK-accelerometer-sensing
https://github.com/edisonslightbulbs/APK-notifications

## A.5 Systems, toolkits, and APIs

https://github.com/edisonslightbulbs/Traceless
https://github.com/edisonslightbulbs/3DINTACToolkit

## A.6 Object detection model training

https://github.com/edisonslightbulbs/PTOD-model-training
https://github.com/edisonslightbulbs/TFOD-model-training