SurfaceCast

Ubiquitous, Cross-Device Surface Sharing

Echtler, Florian; Maierhöfer, Vitus; Hansen, Nicolai Brodersen; Wimmer, Raphael

Published in:
Proceedings of the ACM on Human-Computer Interaction

DOI (link to publication from Publisher):
10.1145/3626475

Publication date:
2023

Document Version
Publisher's PDF, also known as Version of record

SurfaceCast: Ubiquitous, Cross-Device Surface Sharing

FLORIAN ECHTLER, Aalborg University, Denmark
VITUS MAIERHÖFER, University of Regensburg, Germany
NICOLAI BRODERSEN HANSEN, Aalborg University, Denmark
RAPHAEL WIMMER, University of Regensburg, Germany

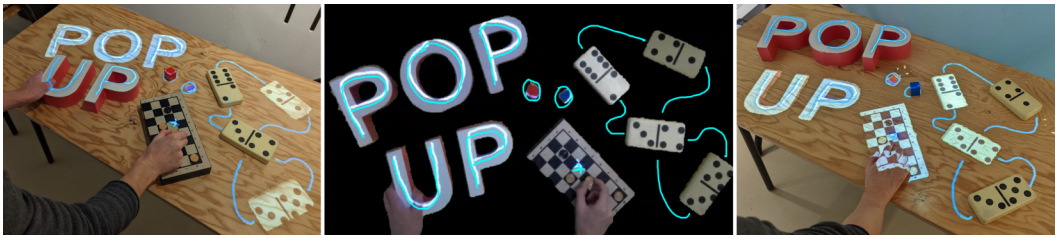

Fig. 1. Three locations, remotely connected via SurfaceCast. Hands and objects on the left table (e.g. the letters "UP") show up as projections on the right table, and vice versa (e.g. "POP" on the right), to enable natural interaction at a distance. The web-based client in the middle image shows merged content from both physical tables and allows for drawing virtual annotations which are visible in both other locations as projections.

Real-time online interaction is the norm today. Tabletops and other dedicated interactive surface devices with direct input and tangible interaction can enhance remote collaboration, and open up new interaction scenarios based on mixed physical/virtual components. However, they are only available to a small subset of users, as they usually require identical bespoke hardware for every participant, are complex to setup, and need custom scenario-specific applications.

We present SurfaceCast, a software toolkit designed to merge multiple distributed, heterogeneous end-user devices into a single, shared mixed-reality surface. Supported devices include regular desktop and laptop computers, tablets, and mixed-reality headsets, as well as projector-camera setups and dedicated interactive tabletop systems. This device-agnostic approach provides a fundamental building block for exploration of a far wider range of usage scenarios than previously feasible, including future clients using our provided API.

In this paper, we discuss the software architecture of SurfaceCast, present a formative user study and a quantitative performance analysis of our framework, and introduce five example application scenarios which we enhance through the multi-user and multi-device features of the framework. Our results show that the hardware- and content-agnostic architecture of SurfaceCast can run on a wide variety of devices with sufficient performance and fidelity for real-time interaction.

CCS Concepts: • **Human-centered computing → Collaborative and social computing devices**; *Collaborative content creation*; *Interactive whiteboards*; *Ubiquitous and mobile computing systems and tools.*

Authors' addresses: Florian Echtler, floech@cs.aau.dk, Aalborg University, Aalborg, Denmark; Vitus Maierhöfer, vitus.maierhoefer@ur.de, University of Regensburg, Regensburg, Germany; Nicolai Brodersen Hansen, nbha@cs.aau.dk, Aalborg University, Aalborg, Denmark; Raphael Wimmer, raphael.wimmer@ur.de, University of Regensburg, Regensburg, Germany.

2573-0142/2023/12-ART439
https://doi.org/10.1145/3626475

Additional Key Words and Phrases: software; toolkit; video; streaming; interactive surface; collaboration; tangible; WebRTC; framework; tabletops

ACM Reference Format:
Florian Echtler, Vitus Maierhöfer, Nicolai Brodersen Hansen, and Raphael Wimmer. 2023. SurfaceCast: Ubiquitous, Cross-Device Surface Sharing. *Proc. ACM Hum.-Comput. Interact.* 7, ISS, Article 439 (December 2023), 23 pages. https://doi.org/10.1145/3626475

1 INTRODUCTION

Interactive surfaces are facilitators for a variety of remote collaboration scenarios, such as hybrid whiteboards, shared tabletops, or mixed physical/virtual environments. However, nearly all of these use cases require participants to have access to the exact same specialized hardware, such as interactive tabletop devices, mixed-reality headsets, or large-scale wall screens. These systems are mostly closed, i.e. it is difficult for third parties to build upon or extend them. In addition, the required equipment is usually highly scenario-specific, expensive, and often not widely available - which is especially problematic for distributed participants. Consequently, users without access to such configurations have so far been limited to purely digital collaborative solutions, such as Miro (for whiteboard work), Teams/Skype/Zoom (for meetings), or specialized software such as Tabletop Simulator for playing games.

To overcome these challenges, we developed SurfaceCast, a software toolkit designed to enable easy creation of shared mixed-reality surfaces for multiple distributed participants using a wide variety of hardware devices. SurfaceCast aims to approximate the experience of collaborating at a single physical surface, thereby providing a building block for a multitude of interaction scenarios with any devices currently available to participants (such as tablets, laptops, ad-hoc projected surfaces, or consumer-grade mixed-reality headsets). Central components are a universal web-based API, a device-independent frontend, and a robust and performant streaming/mixing backend. SurfaceCast is based on a content-agnostic architecture to mix and distribute data from the shared physical and/or virtual surfaces, i.e. unmodified visual content is shown to each remote participant to approximate a co-located shared surface as closely as possible (within the constraints of each specific device class). While both commercial (e.g. Twilio) and open-source platforms (e.g. Kurento or BigBlueButton) exist for multi-client media streaming, none of these systems support layering and particularly alpha-blending of streams. We opt for a standalone implementation (as opposed to extending an existing media server) to keep the installation effort minimal, and to allow easy extension of the system by other researchers without having to adapt to a far larger code base.

Our core contribution is a *software toolkit* to build shared interactive surface scenarios with multiple distributed parties across heterogeneous devices, i.e. an *artefact* [55], using existing technology to provide novel functionality [11]. As our main contribution is the framework itself as a building block for new interactive systems and not the scenarios themselves, we evaluate our contribution based on a) *demonstrations* of replicated examples augmented with new features, and b) *technical performance* [31]. A formative user study in which participants solved a collaborative puzzle task highlights the potential of the concept and properties of the implementation.

2 RELATED WORK

Our toolkit builds on prior work from four subcategories, which we group into *Shared Point-to-Point Spaces* (connecting two interactive surfaces in disjoint locations), *Multi-Party Shared Spaces* (having three or more persons interact in a shared mixed-reality environment), *Analysis of Remote Collaboration* (socio-technical perspectives on remote interaction), and *Usage Scenarios* (employing one or more of the previously discussed technologies for a specific use case).

2.1 Shared Point-to-Point Spaces

Various systems for sharing surface content between *two* locations have been proposed. Below, we present samples from earlier research which follow this approach.

For example, Unver et al. [51] developed a one-to-one desk sharing application for the HP Sprout, a combined desktop computer and top-down projector. IllumiShare by Junuzovic et al. [27] uses a modified desk lamp with projector and camera to share the desktop space with a second device. Similarly, LuminAR by Lindner et al. [32] uses an actuated desk lamp with projector and camera for local augmentations.

Earlier work by Wilson (PlayAnywhere, PlayTogether) [53, 54] uses a custom build with a short-throw projector and infrared camera to mirror a chess game between two locations. MirageTable by Benko et al. [4] uses a point cloud rendered from the correct point of view of the local user to create the illusion of a shared 3D task space. Similarly, *3D Helping Hands* by Teccia et al. [50] merges data from two separate locations with depth cameras into a single point cloud which is then displayed at both sites.

With MeetAlive by Fender et al. [9], an instrumented room with multiple projectors and depth cameras virtually extends the participants' personal computer desktop into the room around it, accounting for different perspectives and projection surfaces. Re-Locations by Fink et al. [10] follows a similar approach, but splits the shared room up into two disjoint locations. Iwai et al. [23] present a two-table setup that pays particular attention to providing a perspectivically correct view of the participant and items on the other side. More recently, Apple DeskView [1] (part of Continuity Camera) uses the wide-angle camera in current iPhones to act as a webcam, capturing both the face of the user as well as part of the table surface in front of them, and streams a rectified view of the surface to the remote party. In addition, Grønbæk et al. [12] present RealityBlender, a toolkit to (partially) align two remote mixed-reality spaces.

Although a wide variety of systems have been presented that enable sharing between *two* surfaces or spaces, all of them employ one specific and usually custom type of hardware that has to be present at both locations, and cannot easily be extended to three or more parties. In contrast, SurfaceCast allows to merge three or more locations using heterogeneous equipment into a single shared mixed-reality space.

2.2 Multi-Party Shared Spaces

In this section, we will now look at earlier works which focus on supporting multiple (i.e., three or more) remote participants in a shared environment.

Early work by Regenbrecht et al. presents Carpeno [40], a hybrid VR/tabletop environment integrating multiple local and remote laptops on a shared tabletop space that extends into VR, already introducing many concepts that appeared again in later systems. Sun and Regenbrecht [48] also investigated how to setup 3-party videoconferencing to allow better spatial awareness between participants, similarly to earlier work by Sellen et al. [44] which used dedicated camera-display units for each remote participant. Tang et al. [49] present VideoArms, a mixed whiteboard-tabletop environment which focuses on delivering a plausible image of the participants' hands and arms to the remote location.

Saffo et al. used VRChat, a shared multi-party VR environment, to replicate earlier tabletop interface studies in virtual reality [42]. Similarly, Grünefeld et al. present VRception [13], a VR toolkit to simulate and prototype various extended reality systems in a purely virtual environment. Yang et al. [57] use a mixed VR and desktop environment to support collaborative sensemaking among multiple users.

The Inter-Reality Interactive Surface by Pietroszek [37] uses an asymmetrical setup with an interactive whiteboard in one location which is accessed by a remote participant via VR glasses. Handberg et al. present SharedSpaces Mingle [15], a shared media space where multiple front-facing videos of participants are mixed together in front of a common backdrop, additionally allowing for drawings and custom background images controlled by the users.

Jasche et al. [24] present BeamLite, a shared MR space that can be used at the same time by co-located and remote participants using AR or VR headsets. Speicher et al. [46] introduce XD-AR, a toolkit to unify various types of mixed-reality devices (e.g. HoloLens, Tango devices, RoomAlive setup) into a single shared, co-located 3D space. Hershkovitz et al. later extend this work into a similar toolkit called XSpace [18].

While a variety of multi-participant shared media spaces already exist, they all are either restricted to traditional 2D face-to-face video conferencing, or require specialized hardware to participate. In contrast, SurfaceCast is able to make use of a wide variety of commodity hardware, fusing multiple remote devices into a single shared space.

2.3 Analysis of Remote Tabletop Interaction

In this section, we will focus on works which analyze the social components of remotely shared task spaces, to verify our system's focus on approximating a co-located tabletop environment without extensive digital augmentation.

Liu et al. [33] analyzed the motivational differences between real and digital card games, and found that players of the real-world variant are focused more on the social aspects of the game, while the digital version is more streamlined and less social. This echoes earlier findings by Hauber et al. [17], who observed people completing collaborative tasks using various levels of remote-collaboration technology. While task performance was slightly increased for the purely digital "remote desktop" approach, the feeling of social presence was more pronounced in the spatially-aligned scenario.

LaLone [30] studied the effects of "too much" automation on boardgames, and found that people value at least parts of the manual experience over a digitized approach. This is echoed by Rogerson et al. [41], who focused on the "materiality" of the game components. Likewise, Darzentas et al. [5] have explored how, in the miniature wargame "Warhammer 40.000", the actual tangible objects play a key role, and how technology should enhance, rather than replace these.

These findings support our approach of approximating a regular co-located tabletop experience without further augmentation, regardless of the content, and of allowing each participant to still interact with tangible physical components as far as possible.

2.4 Usage Scenarios

Lastly, we will review research which uses remote shared spaces for specific application scenarios, such as geospatial data, remote dining, tabletop games, or worker assistance and training.

An obvious application scenario for shared surfaces are digital whiteboards, SCRUM boards, geospatial data visualizations, and similar vertical interactive spaces. For example, Avellino et al. introduce CamRay [2] which embeds an array of cameras into a large tiled wall display to provide face-to-face video communication between multiple persons at different locations. CollaBoard by Kuechler and Kunz [29], Zillner et al. with 3D-Board [61], and Higuchi et al. with ImmerseBoard [19] follow a similar approach, where a whole-body representation of the remote person is captured and rendered at the perspectively correct location for the local user.

A surprisingly widely studied scenario, perhaps due to the rising prevalence of long-distance relations, is remote dining using connected tabletops [3, 8, 52]. In a related context, Yarosh et al.

presented the ShareTable [58, 59], a two-party shared tabletop that was specifically designed and deployed for remote parent-child interaction, similar to Family Portals by Judge et al. [26] which connected three families through a shared whiteboard and video space.

Various groups have also investigated the effects of remote collaboration specifically for tabletop games, e.g. for complex role-playing games conducted using standard videoconferencing software [60], for remote roleplaying with dedicated support software [47], for a hybrid AR-tangible chess game [14], for a hybrid physical-digital version of the game "Wavelength" [35], or for a simple Ludo game in an asymmetric scenario [36]. So, while digital tools of high quality exist for even complex tabletop gaming scenarios, we, like e.g. Rogerson et al. [41] and Darzentas et al. [5], consider playing with physical objects to be core to a exciting tabletop gaming experience.

Radu et al. propose the Virtual Makerspace [38], a shared MR environment in which remote users with VR equipment can participate in activities in a makerspace equipped with 3D cameras. In a related context, Ludwig et al. introduce shARe-it [34], a solution for ad-hoc remote troubleshooting in a fablab using augmented reality goggles. Similarly, Rasmussen et al. [39] present a worker-assistance system which enables the calibration and use of multiple cameras in the workspace, to enhance environmental awareness by the remote helper in 3D space.

Once again, we see that the wealth of application scenarios is usually addressed through bespoke hardware and software that is difficult to replicate or transfer to a related context, and as such is difficult to use as building blocks for further work in concrete use scenarios.

2.5 Summary

As the review of related work shows, there is currently no content-agnostic platform available that can merge multiple distributed, heterogeneous clients into a single, shared mixed-reality space. Existing solutions for multi-client media streaming, such as BigBlueButton, Twilio, or Kurento, do not support the crucial feature of layering and alpha-blending streams, as well as partial mixing.

Our central contribution in comparison to related work, therefore, is a software abstraction that subsumes multiple disjoint facets of previous works into a unified toolkit. This makes it much easier and thus practically feasible for the first time to implement a multitude of usage scenarios, without being tied to a specific type of device (e.g. Kinect [50]; HP Sprout [51]) or class of content (e.g. tabletop games [47]; map data [2]).

In the following section, we describe the fundamental architecture and features of SurfaceCast, our open-source toolkit which enables this functionality.

3 SURFACECAST

The central design goal of SurfaceCast is to enable multiple spatially distributed persons, using heterogeneous devices, to access a shared surface space. To carry over some of the experience of collaborating together at a physical surface, even across multiple different device classes in one session, we always provide both a video stream of participants' faces (front view) and a stream of the shared tabletop (surface view). To enable people with a wide variety of devices to join, we started with a fundamental design decision: SurfaceCast only streams 2D representations of the surface contents. While this approach invariably reduces the overall fidelity by removing depth cues (see also Section 5.1), it considerably broadens the range of supported devices, as users do not need any type of complex 3D-capable sensor or display (depth camera, HMD, autostereoscopic screen, shutter glasses, etc.).

Following the taxonomy for collaborative augmented reality by Sereno et al. [45], our approach is *hybrid* in the space dimension (i.e. supports both remote and co-located users), *synchronous* in the time dimension, *symmetric* with respect to the users' roles (i.e. all users see the same interface),

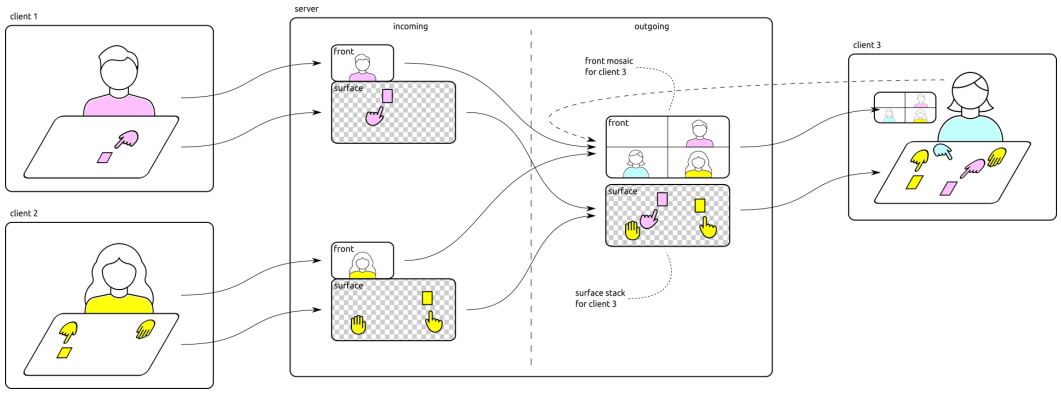

Fig. 2. Partial view of SurfaceCast server architecture with 3 clients. Each client sends two video streams to the server, one *front* stream showing the user and one *surface* stream showing physical objects on the client's surface. Other parts of the surface stream are marked as transparent, so that the server can layer them on top of each other and send them back to the other clients. The client's own front view is also included in the global front view mosaic (dashed line). However, to avoid video loops, the surface stream for client 3 does not include the client's own surface objects (turquoise).

and decidedly *asymmetric* regarding the technology employed by each individual user (although symmetric setups are still possible).

3.1 System Architecture

SurfaceCast implements a client-server architecture, with multiple clients sending separate front and surface video streams to a central mixing server, which recombines the incoming streams and returns them to the clients.

Each SurfaceCast client sends three streams to the server: the *surface stream* showing the physical contents of the local surface, the *front stream* showing the user(s), and the *audio stream* for voice communication (see also Figure 3). In parallel, the server sends three streams back to each client: the surface view that consists of the composited surface streams of all *other* clients (excluding the receiving client itself to avoid video loops); the front view showing a mosaic of all incoming front streams, including each user's own "self view", and the audio stream which is mixed from all other incoming audio streams (see Figure 2 for a partial view). The combined surface streams of the other clients are layered on top of each other; we refer to this component as the *surface stack*.

Note that we require the outgoing surface stream and the incoming surface view to be a) rectified and b) spatially aligned to each other on each individual client. While this is trivial to achieve for some cases (e.g. the web client, or most models of dedicated interactive tabletops), it requires a separate calibration step for others such as a projector/camera-based setup. The overall concept is inspired by SurfaceStreams [6], but uses a standards-compliant WebRTC bundle, carrying two H.264 video streams and one Opus audio stream in both directions. This approach allows us to take advantage of the error correction, recovery, and synchronization features of WebRTC, and enables use of a much wider variety of client devices, such as plain webbrowsers, WebXR platforms, or also standalone clients.

An important challenge for this approach is to avoid video loops in which the projected content gets picked up by the system again, thereby leading to undesired artefacts or "whiteout". To address this issue, any SurfaceCast client is required to exclude the *incoming* surface view from its *outgoing* surface stream. This can be achieved through various means, such as software-based background

Table 1. SurfaceCast API. All clients (both HTML5-based and standalone) communicate with the server using these channels; any future clients are expected to adhere to the same formats. Note that the given resolutions are just default values and can be scaled up if a higher level of detail is desired.

Channel	Protocol	Stream encoding	Default parameters
Flags/SDP messages	WebSocket	JSON	UTF-8
HTML/JS content[a]	HTTPS	plaintext	UTF-8
Surface stream	WebRTC	H.264[b]	outgoing: 1280x720, green chroma-key incoming: 1280x720
Front stream	WebRTC	H.264[b]	outgoing: 640x360 incoming: 1280x720
Audio stream	WebRTC	Opus	both directions: mono, 48 kHz, 64 kBit/s

[a] only relevant for browser-based clients
[b] Constrained Baseline Profile, Level 3.1+, 15 FPS, 1.5 MBit/s

subtraction, by using a depth camera to filter out the projection surface itself, or through an infrared sensor that does not pick up the visible projection (see also Section 3.2). In any case, the final outgoing surface stream is expected to only show the physical objects locally present on the surface, with all other parts of the display surface itself marked as transparent.[1]

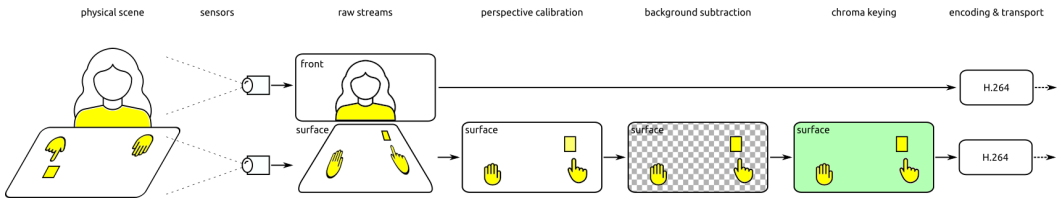

Fig. 3. Video processing pipeline inside a camera-based SurfaceCast client.

The capture module of SurfaceCast also contains a four-point homography calibration component that can be used to extract and rectify a suitable sub-area of the tabletop surface through a homography transformation. Using the **own** option flag sent to the server (see also Section 3.3), the client can request that its own surface stream is included in the surface output, which is required for some scenarios so that the user can see their own actions in the composite stream.

As the common video codecs required for WebRTC (H.264 and VP8) do not natively support alpha channels, a workaround for delivering the transparency information is required. The three possible options are a) sending an additional video stream containing the alpha channel, b) sending a double-height video stream with the alpha channel inserted below the original video, and c) blanking the transparent parts of the video with a predefined chroma-key color. Due to performance considerations, we select option c) (chroma-key on 100% bright green), as option a) would require an additional pair of video encoders and decoders, while option b) would still need to encode twice as many pixels per second when compared to option c).

[1]Note that some parts of the projection might still get included if they are themselves projected onto physical objects.

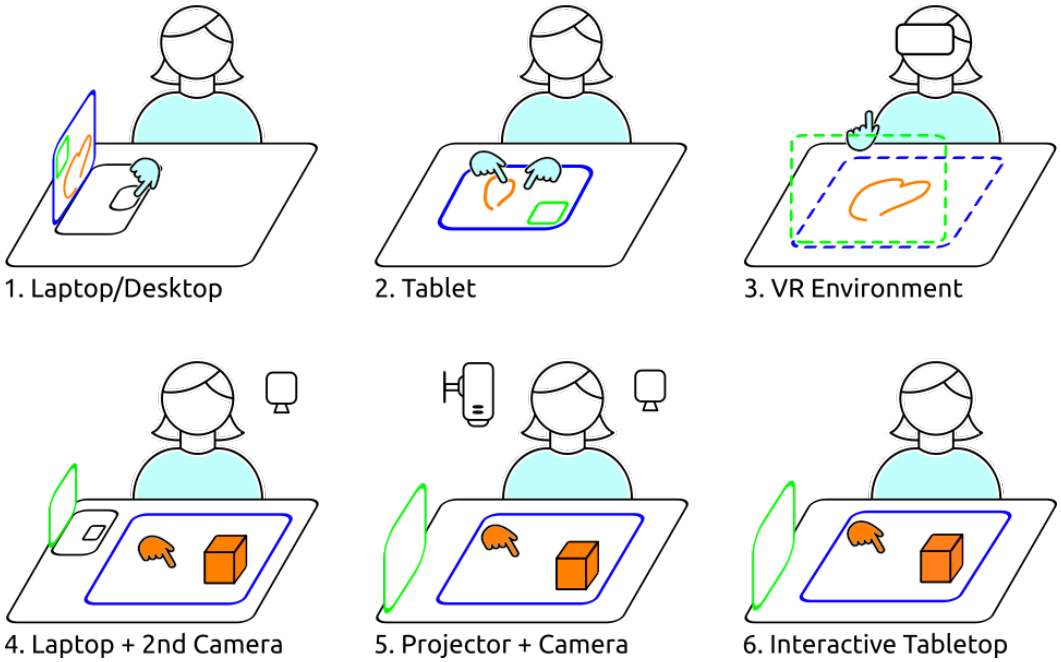

Fig. 4. Possible configurations of SurfaceCast clients. The blue outlines represent the interactive surface area, while the green outlines represent the front view display. In configurations 1 – 3, virtual annotations are supported in the outgoing surface stream, while the other configurations allow for physical tangible objects on the surface (both colored orange).

For evaluation purposes, the SurfaceCast server optionally allows to record all combined streams (front view mosaic, surface view stack, audio stream) to an MKV file which can then later be reviewed using standard video player software such as VLC. Additionally, all front streams can be tagged with an arbitrary text label for easier identification of a specific client.

Both server and standalone client are written in Python 3 and use the GStreamer library bindings for computationally intensive media processing tasks such as audio and video capture, compression, mixing etc. The HTML5 client is written in standard JavaScript with minimal external dependencies (WebRTC adapter.js 8.2.2) and has been verified to work on Chrome, Firefox, and Safari. The WebXR client is based on an extension of the HTML5 client using A-Frame 1.4.1 and has been verified to work on the Meta Quest 1/2 and Microsoft HoloLens 2 headsets using their built-in browsers, as well as using the third-party Wolvic VR browser. All code is available in our open-source repository – see https://github.com/floe/surfacecast for additional details.

3.2 Hardware Configurations

In this section, we describe the various possible hardware configurations for clients that can participate in a SurfaceCast session. These range from a regular desktop or laptop computer, optionally extended with a second top-down camera, over tablets, AR & VR head-mounted displays, to top-down projected tabletop interfaces or dedicated interactive tabletop hardware. Generally, any device with at least one display, one user-facing webcam, and one virtual or physical surface input can join a session. An illustration of the various potential configurations is shown in Figure 4.

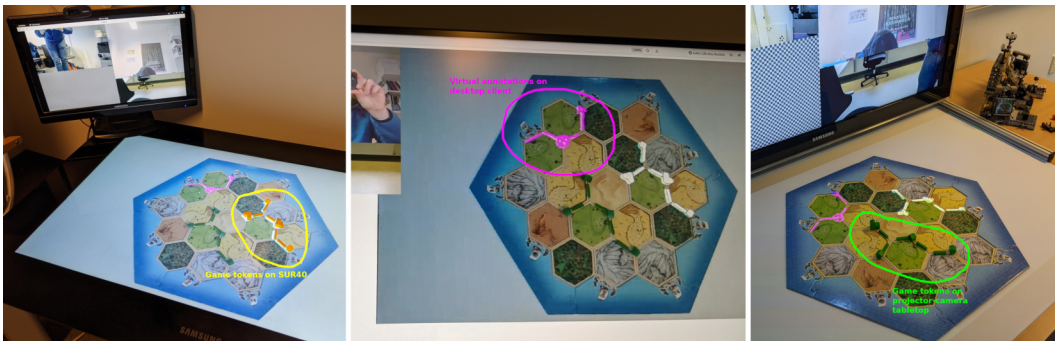

Fig. 5. SurfaceCast running on three clients while playing a board game (Settlers of Catan). Left: a SUR40 tabletop with physical yellow tokens; center: a desktop client using virtual magenta-coloured annotations; right: a SurfaceStation device with physical dark-green tokens and physical game board. For each client, the local components are circled in the corresponding color. The "front mosaic" is visible in the background screens on the left and right clients.

3.2.1 Standalone Laptop/Desktop. The minimum hardware configuration for joining a SurfaceCast session is a regular laptop or desktop computer with a web browser. The HTML5-based SurfaceCast client will open two video streams, one showing the incoming front view of the other persons in the session and one showing the surface view (see also Figure 9 left). For the outgoing streams, the front view is provided by a regular user-facing webcam, while the outgoing surface view is provided through a virtual overlay that the user can paint on and erase via left and right mouse button, respectively. In addition, user-customizeable image "stickers" can be placed on the virtual surface, and moved, scaled, rotated, and deleted using the cursor and a per-sticker delete button. These are intended to simulate tangible objects of any kind (game tokens, post-it notes, decorative items) to create a closer approximation between the physical surface and the browser-based client. Note that a session which contains only laptop/desktop/tablet clients is possible, which would actually be very similar to a standard video call with a shared virtual whiteboard. For this scenario, it is also possible to insert a screencast of any other software on the client as background in the surface view to mimic a physical surface which will then be streamed to the other clients.

Tablet. The tablet configuration is very similar to the desktop configuration, with the difference that the surface view is now shown in fullscreen mode and the front view is a movable overlay on top. Front stream output is provided by the tablet's user-facing "selfie" camera, while the surface stream output is again created through a virtual overlay. Like on the desktop client, this overlay can be annotated with touch-based drawing and virtual stickers, which can be transformed using common multi-touch gestures.

3.2.2 Laptop/Desktop + 2nd Camera. For a slightly more immersive experience, the minimum laptop/desktop setup can be extended with a second surface-facing webcam. The surface view will now consist of a livestream of the table surface, capturing the user's hands and any additional objects placed on the surface. The surface output will still be provided via a video window on the plain screen, but users are easily able to coordinate their on-surface movements with the composite livestream. Alternatively, if the screen is placed in between the user and the surface, a "virtual window" view can be created which shows the combined local content along with the remote objects.

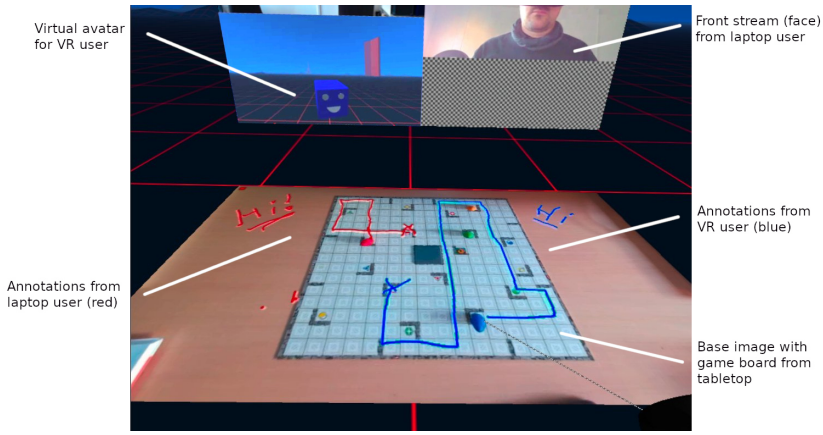

Fig. 6. SurfaceCast VR client, running in a Oculus Quest 1 headset, with two virtual screens (front/surface) shown. A game scenario with both physical items (background), local annotations (red) and remote annotations (blue) is shown. The virtual avatar of the VR user is visible in the bottom left of the front screen, while the front view of the laptop user is visible in the top-right webcam stream.

For this type of client, background subtraction can optionally be performed by providing a uniformly-coloured white background in the camera view, e.g. a large sheet of paper, which will then be replaced with the chroma-key color (bright green) in the outgoing surface stream.

3.2.3 Standalone HMD. Assuming that a WebXR-capable head-mounted display is available (either a VR headset such as the Meta Quest 1/2, or an AR headset such as the Microsoft HoloLens), an alternative configuration uses two virtual screens for output, and the headset's builtin sensors for input. The head pose of the user is transferred to a simple predefined box avatar rendered off-screen and transmitted as front stream, while the tracked positions of the user's hands or controllers (depending on headset capabilities) are rendered to a second off-screen surface and transmitted as surface stream. In addition, the user can also draw on the virtual surface using the VR controllers like in the web client, through raycasting onto the virtual tabletop.

3.2.4 SurfaceStation. The *SurfaceStation* is a self-contained video conferencing & SurfaceCast client, which consists of a 40" upright screen combined with a top-down projector, both mounted to a portable frame that can be placed on any regular table. A regular webcam captures the user(s), while a downwards-facing depth camera (currently supported: Intel Realsense or Kinect Azure) captures physical objects present on the surface while filtering out the projection surface itself through RANSAC-based plane detection. The same four-point calibration module as before is used to map the captured, outgoing stream precisely to the projected surface area. By default, the depth filter will capture any object that is at least 3 mm above the detected plane of the table surface, which is sufficient to include fingers, hands, and most objects in the outgoing stream while removing the projected surface itself. The height can be adjusted through hotkeys at runtime.

3.2.5 Interactive Tabletop. A dedicated interactive tabletop that uses a camera-like sensor, such as the original reacTable [25], the Samsung SUR40 [7], or the more recent MultiTaction MT554, can also be used to join a SurfaceCast session (see Figure 7). A regular screen and webcam are used to display and capture the front stream, respectively, while the built-in sensor provides the outgoing surface stream and the tabletop display is used for the incoming surface stream. In this scenario, background subtraction happens implicitly, as the infrared image sensor in the tabletop does not

capture the visible screen or projection content. If the device is able to identify and track specific tokens (e.g. reacTIVision markers or similar) and transmit them using TUIO [28], then objects with these tokens on their underside can be replaced by customizable images in the outgoing surface stream, thereby allowing to differentiate between multiple items that do not have an easily distinguishable shape.

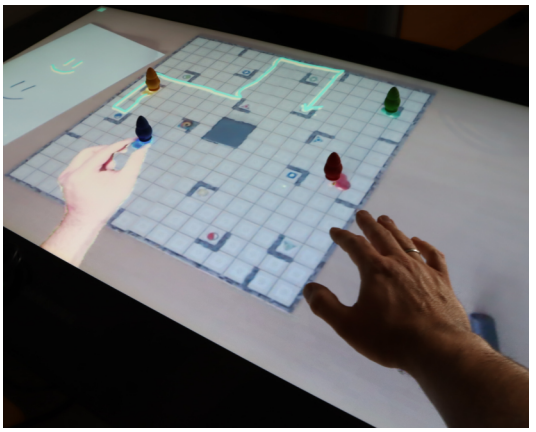

Fig. 7. Joining a SurfaceCast game session with a Samsung SUR40 tabletop. Left hand is from remote player using a Surfacestation, right hand is from local player. Painted path is from 3rd player on a laptop (front screen & front camera not shown).

3.3 Client Setup

Setting up a SurfaceCast client involves two main steps: a) calibrating the camera (only applicable to camera-based clients, see Section 3.1), and b) configuring the surface stack, which contains the individual alpha-masked surface streams from the other clients.

To adjust the server-side handling of the surface stack to the specific hardware setup, each client can send option flags to the server through the WebRTC data channel. Normally, each client will receive the combined, alpha-blended surface streams of all *other* clients, ordered by which client connected first. However, there are some scenarios/setups which may require other configurations for the stack.

Setting the **`main`** flag will cause the surface stream of this specific client to be inserted at the "bottom" of the stack for all clients, i.e. content from all other clients will always be layered on top of this one. This option is needed, e.g., if a client is intended to show a base image that all other clients will refer to, such as a game board to play on, or a floorplan for discussion. For example, this could be a desktop client streaming an external application window, or a projector-camera setup with a physical printout, which should be visible on all other clients.

In contrast, setting the **`own`** flag will insert the client's own surface stream back into its own surface stack. While this is not usually necessary and will degrade the video quality by causing feedback effects for most setups, some clients that use different channels for surface input and output (e.g. a desktop machine with second camera) can use this option to improve user experience by making the virtual surface stream visible to the client itself again.

4 EVALUATION

4.1 System Demonstration

We primarily conduct an evaluation by demonstration [31], in which we re-create scenarios that have previously been presented in the literature, and extend them using the capabilities of SurfaceCast where applicable, e.g. by including three participants instead of only two, or by using heterogeneous client devices. This allows us to verify the feasibility of our approach in a realistic context, while at the same time also testing the cross-device and multi-user features. In addition, we also conduct a small-scale user study to verify the general usability of a system built using SurfaceCast. Finally, we provide performance metrics in terms of server load, client load, required network bandwidth, frame rate, and latency.

4.1.1 Board Game. A variety of popular board games are playable using our setup, provided that a physical copy of the game is available to every involved participant. Obviously, games where tokens are drawn from a common pile or exchanged among players will be difficult to replicate while keeping the original rules intact, but this limitation does not affect many traditional games such as Ludo, Halma, Tangram, or chess (provided that the pieces can be easily distinguished in the top-down view, e.g. when using the Bauhaus chess set [16]). Even many recent, more complex games such as Settlers of Catan or Carcassonne are playable as well. In most cases, one of the players with a camera-based client would setup the actual game board on their table, which is then streamed to the other clients. Each of the other players can then use their own physical game tokens on top of the projected game board if they also have a camera-based client, which will then be streamed back to the other tables. If only a virtual client is available (web/XR), then annotations and/or stickers can be drawn on the virtual surface view which will again be visible to the other players in their projections.

We showcase two games: the widely known "eurogame" *Settlers of Catan* as well as the puzzle game *Richochet Robots*, where players have to find the shortest path to a randomly selected goal for one of the robot tokens placed on the board. In our examples, we connect one SurfaceStation client with physical game board and tokens, one SUR40 tabletop device with virtual gameboard and physical tokens, and a desktop or VR client. The player on desktop/VR can draw their tokens on the game board as virtual annotations, while the other players can directly move the physical tokens on their own side. This increases the feeling of immersion through tangible interaction on the tabletop side, while still allowing the desktop/VR player to participate without having access to a dedicated tabletop device. For illustration, we refer to the pictures in Section 3.2 which all depict the board game scenario, e.g. figures 5 and 7.

Fig. 8. Remote dining using SurfaceCast, running on two SurfaceStation clients.

4.1.2 Remote Social Dining. As discussed above, there has been considerable research into enabling remotely shared social dining [3, 8, 52], a topic that perhaps gained in relevance again during and after the COVID-19 pandemic. But also under "normal" conditions, e.g. long-distance relationships could benefit from a more personal way to share a meal, or remote friends who are interested in cooking can share their creations with each other in a natural setting. We recreated this scenario using two SurfaceStation devices as shown in Figure 8. Each participant has their own food in front of them, while seeing the projection of the others' food and interaction.

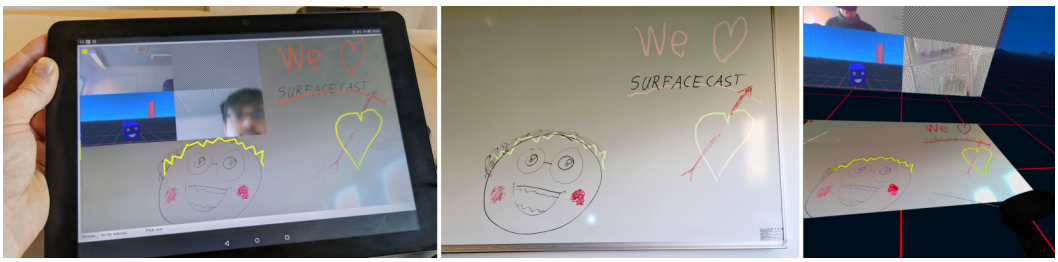

Fig. 9. Shared whiteboard scenario. Center: physical whiteboard with projected annotations and local drawings; left: web client running on Android tablet (yellow annotations); right: VR client running in Oculus Quest (red annotations)

4.1.3 Shared Whiteboard. For this scenario, we choose a brainstorming session at a whiteboard, e.g. in a distributed research collaboration or for a remote team in a software company. We use a regular upright whiteboard that is also projected on with a standard projector, connected to a laptop with an additional webcam. The projection area and webcam image are aligned to each other using the standalone SurfaceCast client for surface stream input/output, while the laptop screen and built-in webcam provide front stream input/output. Two external users join via their tablet and VR headset, respectively, and can draw annotations on the physical whiteboard, to be discussed with the co-located participants. A unique drawing color is assigned to each remote participant (see also Figure 9).

Fig. 10. Piano teaching scenario. The projected hand of the teacher is visible on the right, while the student emulates the pose with their own hand on the left.

4.1.4 Piano Teaching. This scenario was inspired by MirrorFugue [56], in which an experienced pianist's hands are projected onto the piano of their student, to provide a direct example of correct hand and finger placement (see Figure 10). Vice versa, the teacher is also able to see the student's hands so that they can give real-time feedback. The default hardware setup for this scenario would consist of a camera and projector mounted above a regular piano, although the teacher could use any camera-based sensor such as the SUR40 as well. This example illustrates how even usage scenarios that are not immediately related to interactive surfaces can still be easily configured using SurfaceCast without needing to rely on custom streaming code.

4.1.5 Collaborative Play. This scenario uses one projected tabletop and one tablet client to enable collaborative remote building and play with Lego and other toys. One possible context for this scenario are distributed families, where e.g. the grandparents can play with their grandchildren remotely, with the children having the actual Lego model and the adults interacting through their projected hands, drawings, and/or virtual stickers.

In an informal evaluation with an 8-year-old, the child quickly accepted the projection as part of their play and built structures around projected figures, asked the adults on the other client to "move that over there", hid objects from the camera's view, and integrated the remote persons into their interaction.

Fig. 11. Collaborative play between spatially separate locations, using a projected tabletop (left column) and an iPad (right column). Top row: the users have improvised a Tic-Tac-Toe game using physical Lego bricks and virtual stickers. Bottom row: the iPad user is building a virtual road, which the child on the physical side uses as a drawing template.

A second, public deployment of this scenario was installed in two separate locations at the Maker Faire in Aarhus, and was used by 11 groups of people, with 27 participants in total (four groups of children aged 9-14, and seven families with children aged 6-12). Alongside a selection of Lego

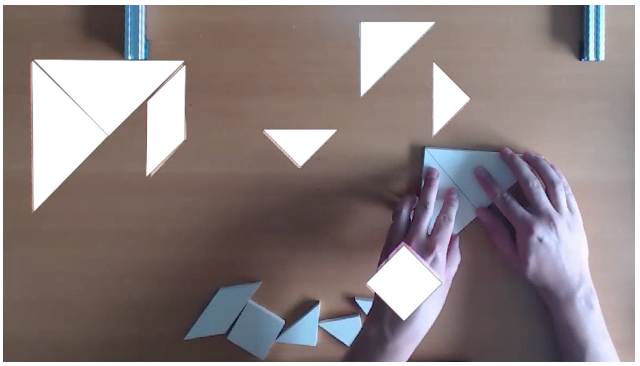

Fig. 12. Collaborative Tangram puzzle. The bright white tiles are projected from the tablet user's device, while the off-white tiles are physically present on the table and are manipulated by the local user.

bricks and figures, we also provided paper and drawing pens on the "physical" side. People were quick to adapt the setup to their own play ideas, such as collaborative drawing, building virtual tracks for Lego figures, or playing Tic-Tac-Toe (see also Figure 11).

A modification of this scenario could include remote teaching and learning, e.g. with several children in remote locations, that can share a common virtual table for a classroom-like experience.

4.2 User Study

To verify the fundamental usability of our system with regular users (i.e., persons who are not themselves developers of the system), and to better understand the different collaboration modes, we conducted a between-subject user study with six pairs of users (12 in total). We intentionally kept the study scale small, as our goal was not to compare performance metrics or gain detailed insights into differences between the individual modes, but rather to verify that a system built with SurfaceCast can be used in a real-world context, and to inform directions for future studies.

Our users had to complete two tasks together on one of three different setups: a) a single unaugmented table, b) two projected tabletops, and c) a combination of one projected tabletop and one tablet. Two tables with projector-camera systems were set up in adjacent rooms within our lab. As all traffic was transferred via a SurfaceCast server at an external hosting company, end-to-end latency was comparable to typical usage scenarios.

For the study context, we chose collaborative tasks based on the traditional Chinese "Tangram" puzzle (see also Figure 12). The participants received one set of Tangram pieces each (13 cm at longest edge), and two abstract outlines which they had to recreate using the Tangram pieces. The first task could be solved individually, although participants were encouraged to arrive at a solution together, while the second task could only be solved when both participants combined their Tangram pieces. The task was deliberately kept simple so that the influence of the communication setup on users' behavior stood out well, and the test always started with the individual task as a "tutorial", followed by the collaborative task.

Regarding the study conditions, we opted for a between-subject design to avoid learning effects and keep the duration of each individual test manageable. Each test session lasted approximately 35 minutes. Ten of the participants were selected from local computer science students through opportunity sampling, while the other two participants were an arts student and an apprentice. All

users received chocolate and/or user study course credits for their participation. Their average age was 23 years (5 female/7 male).

Inspired by Scott et al. [43], we observed turn-taking behaviour, verbal interaction, and physical interaction such as pointing. All interactions were recorded on video. Additionally, participants answered questions from the Social Presence module of the Games Experience Questionnaire (GEQ) [22] and a few questions about their general experience with the system, similar to [36].

We did not find any statistically significant differences between the three conditions for any of the GEQ items on social presence (t-tests with Bonferroni correction, $p < .05$). The average social presence scores for the individual conditions were: 4.29 (σ 1.96) for the plain table, 4.24 (σ 1.94) for the dual-table setup, and 4.37 (σ 1.95) for the tablet-table condition. On the one hand, this is not surprising given the small number of participants. On the other hand, this suggests that we should not expect to see large differences between the three conditions, even with larger numbers of participants. Consequently, both SurfaceCast variants seem to support a level of social presence which is similar to that of the plain shared table.

Nevertheless, our observations and participants' further responses indicate a few potential differences that might be worth exploring in more detail.

When solving the tasks on the single shared table, the participants' focus was completely on the game. Communication flowed naturally. As expected, none of the participants mentioned technical problems. Both groups collaborating on the single table setup were observed joking and laughing - a behaviour we did not observe with the two SurfaceCast setups.

In the second condition with the two identical interactive tabletops, we observed that participants used the front stream video call for communicating with each other and also for quickly glancing at the other one in order to check whether they understood what was being said. One participant explicitly lauded the low latency of the system, while another one mentioned high latency as an issue (see also Discussion). A third participant was dissatisfied with the quality of the video stream. Participants demonstrated to each other how pieces could fit together.

In the third condition, Table & Tablet, we observed participants mentioning audio problems, which could later be attributed to a low-quality tablet microphone. Generally, the person working at the table talked more than the person at the tablet. Unlike in the other two conditions, we did not observe participants demonstrating to each other how pieces could fit together.

Overall, all three variants seem to work satisfactorily for solving a collaborative task. There are indications that the combination of tablet and interactive tabletop causes more distraction from the main task than a shared table or a combination of interactive tabletops. This is an aspect we will investigate further in future studies.

4.3 System Performance

We conclude our evaluation by discussing the performance of SurfaceCast across various metrics (frame rate & latency, server & client performance, network bandwidth, and robustness), which are important for usability and user experience in any application scenarios building on top of our framework (tests performed with Python 3.10 & GStreamer 1.20.3).

Frame Rate & Latency. We aim for a constant framerate of 15 FPS and a latency below 500 ms for the whole system, which can be considered a lower bound for an acceptable video conferencing experience.

To measure these metrics, we used a client sending a timecoded synthetic surface stream, and recorded both the outgoing and the incoming stream in parallel on the client side. The difference

between the timecodes equals to the full round-trip latency of the system. Note that this approach does not measure additional latency introduced by individual display and/or camera subsystems, but only the streaming path of our system. For realistic conditions, we used a server located in an external data center, with an approximate one-way network latency of 20 ms.

Based on 832 samples recorded over a 60-second period, we calculate a framerate of 13.9 FPS and an average round-trip latency of 389 ms with a standard deviation of 41 ms for this setup, which is well within the acceptable range for other video conferencing software.

Server & Client Performance. Our main development server uses a Intel Core i7-2600 CPU with 4 cores/8 threads at 1.6 GHz base frequency, and is equipped with 16 GB RAM. On this platform, four simultaneous test clients with server-side stream recording (all streams are 720p at 15 FPS) result in approximately 58% of sustained CPU load at the base frequency, and a moderate 4.8% (~ 800 MB) RAM usage.

To evaluate support for higher-resolution usecases, we also tested the same scenario with all incoming and outgoing streams now at full HD resolution (1080p at 15 FPS). In this case, the server reached nearly 100% CPU load, but was still able to deliver a continuous stream to all clients.

In addition, we verified the performance of our client software across multiple hardware platforms: a standard office machine with an Intel i5-4570T CPU and a high-end laptop with an Intel i7-1065G7 CPU running the SurfaceStation configuration with a Kinect Azure camera; a SUR40 device with an AMD Athlon X2 245e CPU and a Raspberry Pi 4 with a Broadcom BCM2711 SoC (overclocked to 1800 MHz with active cooling) running the dual-camera configuration; an Oculus Quest 1 with a Snapdragon 835 SoC, running the WebXR client on the Wolvic browser; and at the very low end, an Android tablet with an Intel Atom x5-Z8300 SoC, running the web-based client on Firefox. All configurations are capable of consistently delivering the standard 720p at 15 FPS base streams, although the lower-end systems (SUR40, RasPi, Quest, tablet) reach a CPU usage between 75% and 93%.

Network Bandwidth. We currently use a fixed bitrate of 1.5 MBit/s for each outgoing H.264 stream encoder. Therefore, a single client will require roughly 3 MBit/s of both incoming and outgoing network bandwidth. The Opus-encoded audio stream adds only negligible overhead at 64 kBit/s. As part of future work, we will consider using adaptive video bitrates depending on network conditions and content complexity. Nevertheless, our current settings provide an acceptable tradeoff between quality and bandwidth, so that most mobile or broadband connections can easily support 1 - 2 clients.

Robustness. To allow for continuously operated SurfaceCast installations, robustness is also an important factor. As a long-term test of the whole system, we setup three spatially distributed clients (two in Germany, one in Denmark), and had them stream continuously for 8 hours. The system was able to run without issues during the entire time period, also coping with occasional WiFi disconnections and network dropouts. To evaluate the server's robustness, we automated a client to repeatedly connect to a single running instance of the server, stream content for a random duration between 5 and 35 seconds, and disconnect again. Our server component performed 50 iterations of this test without any crashes, hangs, or measurable memory leaks.

5 DISCUSSION

Our goal was to provide a framework that can work across a wide range of devices and scenarios, merging multiple distributed, heterogeneous clients into a unified shared surface with sufficient performance for realtime interaction between users. We have achieved this goal as evidenced by

our implementation of five test scenarios inspired by the literature, by our user study, and by our performance tests using six different client configurations.

By providing a building block for interactive surface systems that does not depend on a specific type of hardware and can make use of a variety of readily available devices, we enable future research and development of new scenarios in the context of distributed collaboration and interaction.

5.1 Limitations

For any scenario which does not exclusively use virtual input and output for the surface component (e.g. laptop, VR headset), the surface display area should have the same physical size in all locations. The lowest common denominator for our test systems is the Samsung SUR40 with a 40" screen. Consequently, the SurfaceStation is adjusted to use a 40" surface projection as well. However, if size mismatches between locations are acceptable (e.g. for a whiteboard scenario), then this is not a fundamental requirement.

As consumer-grade cameras and projectors usually exhibit some degree of radial image distortion, projected content near the image borders may not always line up accurately with related real-world objects. The homography-based calibration cannot capture this type of distortion and would have to be replaced by a full projector-camera calibration (see also below).

As we currently do not implement adaptive compression control, the system is still somewhat sensitive to network dropouts, and video quality is limited by the fixed, relatively low encoder bandwidth of 1.5 MBps. While this is sufficient for most scenarios, some participants in our user test still suggested improving video quality and/or latency, which we hope to address through implementing dynamic bandwidth usage in the future.

Our intentional limitation to 2D content creates issues when attempting to play more complex board games, such as e.g. Warhammer 40K or Battletech. The intricate miniatures lose a noticeable amount of detail in the top-down 2D view that makes it sometimes difficult to differentiate between individual pieces, and terrain height information is not captured. Switching to 4K streams could partially address this problem while noticeably increasing the system's performance requirements. In addition, games that inherently require 3D movement, e.g. Jenga, are unsupported with the presented setup.

Flat content directly on the surface (e.g. drawings, paper) is currently challenging to integrate, especially when using a client with a depth camera. SurfaceCast provides a background colour filter, but this does not offer the same performance as filtering based on physical depth, which is applicable to hands, game tokens, books etc. In addition, the chroma-key approach (both for filtering the background, and for marking transparency in the transmitted video) sometimes causes visual artefacts at object borders. A grayscale transparency map as discussed above could solve this issue at the expense of some performance.

5.2 Future Work

For some scenarios, we anticipate that content-sensitive digital support can still add value to the interaction, even though we intentionally did not consider this approach for the present work. We plan to extend the SurfaceCast architecture to allow generic computer vision plugins both on the client and on the server to enable content-sensitive interaction, e.g. a text recognition plugin for whiteboards, hand tracking for virtual annotations, or boardgame-specific plugins for easier scoring. Having a plugin to segment objects on the table surface could help with automatically creating stickers matching the context, similar to ThingShare [20]. This approach will also enable an auto-calibration plugin that displays, e.g., a chessboard pattern on the surface display and uses this to calibrate the display-camera system, including radial distortion parameters. In addition, we

will also prototype a configuration GUI which allows e.g. interactive background subtraction and re-ordering of the individual video layers in the surface stack.

Our system does not have any fundamental technical limitation that prevents more than 4 simultaneous clients, but especially the surface view might start to get confusing due to multiple overlapping projections of hands, arms, and physical items. Similarly, it is an open question whether users might over time get irritated by the fact that all remote participants are oriented in the same way around the virtual table environment, i.e. sit in the same virtual spot as the local user. We will investigate how to rotate/modify the individual videostreams before compositing, in order to simulate individual, spatially separate "seats" around a virtual table like in Hydras [44].

In addition, it is difficult to quantify whether we have achieved our initial goal of approximating a regular shared table as closely as possible. This question could potentially be answered through the extended social co-presence questionnaire as in [36]. We plan to conduct a separate qualitative evaluation to investigate these issues, using real-world co-design sessions similar to [21] or full-length game sessions with 4 distributed participants.

6 CONCLUSION

We have presented SurfaceCast, a device- and content-agnostic software framework to merge multiple distributed devices into a single, shared surface. This approach opens up a far greater range of use cases, due to the integration of any available devices instead of specialized hardware. We recreated several existing usage scenarios using SurfaceCast, extending their functionality with heterogeneous devices and multiple clients (i.e. three or more). Our results show that SurfaceCast is able to provide a common platform and fundamental building block for a wide variety of current and future interactive surface scenarios, irrespective of client hardware.

REPRODUCTION NOTE

All source code for this work is available publicly at https://github.com/floe/surfacecast.

ACKNOWLEDGMENTS

This work was funded partly by the German Federal Ministry of Education and Research (BMBF) through grant 16SV8288 within the VIGITIA project, and partly by the German Research Foundation (DFG) through grant EC437/1-1 within the DeUCoE project.

We would like to thank Juliana Zgryzek and Andreas Schmidt from University of Regensburg for their help in conducting the lab-based user study, and Yongxin Zhang, Charlotte M. Guldbæk, and Christian F. D. Jensen from Aalborg University for their help with the public deployment of the system.

REFERENCES

[1] Apple, Inc. 2022. *Use Desk View on your Mac.* Retrieved Dec. 2, 2022 from https://support.apple.com/guide/mac-help/use-iphone-with-desk-view-mchl06927be8/mac

[2] Ignacio Avellino, Cédric Fleury, Wendy E. Mackay, and Michel Beaudouin-Lafon. 2017. CamRay: Camera Arrays Support Remote Collaboration on Wall-Sized Displays. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. ACM, New York, NY, USA, 6718–6729. https://doi.org/10.1145/3025453.3025604

[3] Pollie Barden, Rob Comber, David Green, Daniel Jackson, Cassim Ladha, Tom Bartindale, Nick Bryan-Kinns, Tony Stockman, and Patrick Olivier. 2012. Telematic Dinner Party: Designing for Togetherness through Play and Performance. In *Proceedings of the Designing Interactive Systems Conference* (New York, NY, USA) *(DIS '12)*. ACM, 38–47. https://doi.org/10.1145/2317956.2317964

[4] Hrvoje Benko, Ricardo Jota, and Andrew Wilson. 2012. MirageTable: Freehand Interaction on a Projected Augmented Reality Tabletop. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (New York, NY, USA) *(CHI '12)*. ACM, 199–208. https://doi.org/10.1145/2207676.2207704

[5] Dimitrios Paris Darzentas, Michael A. Brown, Martin Flintham, and Steve Benford. 2015. The Data Driven Lives of Wargaming Miniatures. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI '15)*. Association for Computing Machinery, New York, NY, USA, 2427–2436. https://doi.org/10.1145/2702123.2702377

[6] Florian Echtler. 2018. SurfaceStreams: A Content-Agnostic Streaming Toolkit for Interactive Surfaces. In *The 31st Annual ACM Symposium on User Interface Software and Technology Adjunct Proceedings* (New York, NY, USA) *(UIST '18 Adjunct)*. ACM, 10–12. https://doi.org/10.1145/3266037.3266085

[7] Florian Echtler and Martin Kaltenbrunner. 2016. SUR40 Linux: Reanimating an Obsolete Tangible Interaction Platform. In *Proceedings of the 2016 ACM on Interactive Surfaces and Spaces* (New York, NY, USA) *(ISS '16)*. ACM, 343–348. https://doi.org/10.1145/2992154.2996778

[8] Florian Echtler and Raphael Wimmer. 2014. The Interactive Dining Table, or Pass the Weather Widget, Please. In *Proceedings of the Ninth ACM International Conference on Interactive Tabletops and Surfaces* (New York, NY, USA) *(ITS '14)*. ACM, 419–422. https://doi.org/10.1145/2669485.2669525

[9] Andreas Rene Fender, Hrvoje Benko, and Andy Wilson. 2017. MeetAlive: Room-Scale Omni-Directional Display System for Multi-User Content and Control Sharing. In *Proceedings of the 2017 ACM International Conference on Interactive Surfaces and Spaces (ISS '17)*. ACM, New York, NY, USA, 106–115. https://doi.org/10.1145/3132272.3134117

[10] Daniel Immanuel Fink, Johannes Zagermann, Harald Reiterer, and Hans-Christian Jetter. 2022. Re-Locations: Augmenting Personal and Shared Workspaces to Support Remote Collaboration in Incongruent Spaces. *Proc. ACM Hum.-Comput. Interact.* 6, ISS, Article 556 (nov 2022), 30 pages. https://doi.org/10.1145/3567709

[11] James Fogarty. 2017. Code and contribution in interactive systems research. In *Proceedings of the 2017 HCITools CHI Workshop: Strategies and Best Practices for Designing, Evaluating and Sharing Technical HCI Toolkits*.

[12] Jens Emil Sloth Grønbæk, Ken Pfeuffer, Eduardo Velloso, Morten Astrup, Melanie Isabel Sønderkær Pedersen, Martin Kjær, Germán Leiva, and Hans Gellersen. 2023. Partially Blended Realities: Aligning Dissimilar Spaces for Distributed Mixed Reality Meetings. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems* (Hamburg, Germany) *(CHI '23)*. Association for Computing Machinery, New York, NY, USA, Article 456, 16 pages. https://doi.org/10.1145/3544548.3581515

[13] Uwe Gruenefeld, Jonas Auda, Florian Mathis, Stefan Schneegass, Mohamed Khamis, Jan Gugenheimer, and Sven Mayer. 2022. VRception: Rapid Prototyping of Cross-Reality Systems in Virtual Reality. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems* (New Orleans, LA, USA) *(CHI '22)*. Association for Computing Machinery, New York, NY, USA, Article 611, 15 pages. https://doi.org/10.1145/3491102.3501821

[14] Sebastian Günther, Florian Müller, Martin Schmitz, Jan Riemann, Niloofar Dezfuli, Markus Funk, Dominik Schön, and Max Mühlhäuser. 2018. CheckMate: Exploring a Tangible Augmented Reality Interface for Remote Interaction. In *Extended Abstracts of the 2018 CHI Conference on Human Factors in Computing Systems* (Montreal QC, Canada) *(CHI EA '18)*. Association for Computing Machinery, New York, NY, USA, 1–6. https://doi.org/10.1145/3170427.3188647

[15] Leif Handberg, Charlie Gullstrom, Joke Kort, and Jimmy Nyström. 2016. SharedSpaces Mingle. In *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems* (New York, NY, USA) *(CHI EA '16)*. ACM, 269–272. https://doi.org/10.1145/2851581.2889469

[16] Josef Hartwig. 1924. "Bauhaus" Chess Set. https://www.moma.org/collection/works/4240

[17] Jörg Hauber, Holger Regenbrecht, Mark Billinghurst, and Andy Cockburn. 2006. Spatiality in Videoconferencing: Trade-Offs between Efficiency and Social Presence. In *Proceedings of the 2006 20th Anniversary Conference on Computer Supported Cooperative Work* (Banff, Alberta, Canada) *(CSCW '06)*. Association for Computing Machinery, New York, NY, USA, 413–422. https://doi.org/10.1145/1180875.1180937

[18] Jaylin Herskovitz, Yi Fei Cheng, Anhong Guo, Alanson P. Sample, and Michael Nebeling. 2022. XSpace: An Augmented Reality Toolkit for Enabling Spatially-Aware Distributed Collaboration. *Proc. ACM Hum.-Comput. Interact.* 6, ISS, Article 568 (nov 2022), 26 pages. https://doi.org/10.1145/3567721

[19] Keita Higuchi, Yinpeng Chen, Philip A. Chou, Zhengyou Zhang, and Zicheng Liu. 2015. ImmerseBoard: Immersive Telepresence Experience Using a Digital Whiteboard. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI '15)*. ACM, New York, NY, USA, 2383–2392. https://doi.org/10.1145/2702123.2702160

[20] Erzhen Hu, Jens Emil Sloth Grønbæk, Wen Ying, Ruofei Du, and Seongkook Heo. 2023. ThingShare: Ad-Hoc Digital Copies of Physical Objects for Sharing Things in Video Meetings. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems* (Hamburg, Germany) *(CHI '23)*. Association for Computing Machinery, New York, NY, USA, Article 365, 22 pages. https://doi.org/10.1145/3544548.3581148

[21] Casey Lee Hunt, Kaiwen Sun, Zahra Dhuliawala, Fumi Tsukiyama, Iva Matkovic, Zachary Schwemler, Anastasia Wolf, Zihao Zhang, Allison Druin, Amanda Huynh, Daniel Leithinger, and Jason Yip. 2023. Designing Together, Miles Apart: A Longitudinal Tabletop Telepresence Adventure in Online Co-Design with Children. In *Proceedings of the 22nd Annual ACM Interaction Design and Children Conference* (Chicago, IL, USA) *(IDC '23)*. Association for Computing Machinery, New York, NY, USA, 52–67. https://doi.org/10.1145/3585088.3589359

[22] W.A. IJsselsteijn, Y.A.W. de Kort, K. Poels, A. Jurgelionis, and F. Bellotti. 2007. Characterising and Measuring User Experiences in Digital Games. In *Proceedings of the International Conference on Advances in Computer Entertainment Technology (ACE 2007)*. Salzburg. http://repository.tue.nl/661449

[23] Daisuke Iwai, Ryo Matsukage, Sota Aoyama, Tsuyoshi Kikukawa, and Kosuke Sato. 2018. Geometrically Consistent Projection-Based Tabletop Sharing for Remote Collaboration. *IEEE Access* 6 (2018), 6293–6302. https://doi.org/10.1109/ACCESS.2017.2781699

[24] Florian Jasche, Jasmin Kirchhübel, Thomas Ludwig, and Peter Tolmie. 2021. BeamLite: Diminishing Ecological Fractures of Remote Collaboration through Mixed Reality Environments. In *C&T '21: Proceedings of the 10th International Conference on Communities & Technologies - Wicked Problems in the Age of Tech* (Seattle, WA, USA) *(C&T '21)*. Association for Computing Machinery, New York, NY, USA, 200–211. https://doi.org/10.1145/3461564.3461566

[25] Sergi Jordà, Martin Kaltenbrunner, Günter Geiger, and Marcos Alonso. 2006. The reacTable: A Tangible Tabletop Musical Instrument and Collaborative Workbench. In *ACM SIGGRAPH 2006 Sketches* (New York, NY, USA) *(SIGGRAPH '06)*. ACM. https://doi.org/10.1145/1179849.1179963

[26] Tejinder K. Judge, Carman Neustaedter, Steve Harrison, and Andrew Blose. 2011. Family Portals: Connecting Families Through a Multifamily Media Space. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11)*. ACM, New York, NY, USA, 1205–1214. https://doi.org/10.1145/1978942.1979122

[27] Sasa Junuzovic, Kori Inkpen, Tom Blank, and Anoop Gupta. 2012. IllumiShare: Sharing Any Surface. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '12)*. ACM, New York, NY, USA, 1919–1928. https://doi.org/10.1145/2207676.2208333

[28] 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, Article 8 (June 2018), 35 pages. https://doi.org/10.1145/3229090

[29] Martin Kuechler and Andreas M. Kunz. 2010. Collaboard: A Remote Collaboration Groupware Device Featuring an Embodiment-Enriched Shared Workspace. In *Proceedings of the 2010 ACM International Conference on Supporting Group Work* (Sanibel Island, Florida, USA) *(GROUP '10)*. Association for Computing Machinery, New York, NY, USA, 211–214. https://doi.org/10.1145/1880071.1880107

[30] Nicolas LaLone. 2021. Gameplay as Network: Understanding the Consequences of Automation on Play and Use. In *HCI in Games: Experience Design and Game Mechanics*, Xiaowen Fang (Ed.). Springer International Publishing, Cham, 293–313.

[31] David Ledo, Steven Houben, Jo Vermeulen, Nicolai Marquardt, Lora Oehlberg, and Saul Greenberg. 2018. Evaluation Strategies for HCI Toolkit Research. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (Montreal QC, Canada) *(CHI '18)*. Association for Computing Machinery, New York, NY, USA, 1–17. https://doi.org/10.1145/3173574.3173610

[32] 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* (New York, NY, USA) *(UIST '10)*. ACM, 395–396. https://doi.org/10.1145/1866218.1866237

[33] Sha Liu, Junren Wang, Yang Long, and Yu Tan. 2021. Motivational and Behavioral Differences Between Traditional and Digital Tabletop Games. In *Advances in Usability, User Experience, Wearable and Assistive Technology*, Tareq Z. Ahram and Christianne S. Falcão (Eds.). Springer International Publishing, Cham, 351–359.

[34] Thomas Ludwig, Oliver Stickel, Peter Tolmie, and Malte Sellmer. 2021. shARe-IT: Ad hoc Remote Troubleshooting through Augmented Reality. *Computer Supported Cooperative Work (CSCW)* 30, 1 (01 Feb 2021), 119–167. https://doi.org/10.1007/s10606-021-09393-5

[35] Chelsea Mills, Denise Y. Geiskkovitch, Carman Neustaedter, William Odom, and Benett Axtell. 2023. Remote Wavelength: Design and Evaluation of a System for Social Connectedness Through Distributed Tabletop Gameplay. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems* (Hamburg, Germany) *(CHI '23)*. Association for Computing Machinery, New York, NY, USA, Article 680, 19 pages. https://doi.org/10.1145/3544548.3581142

[36] Jasmin Odenwald, Sven Bertel, and Florian Echtler. 2020. Tabletop Teleporter: Evaluating the Immersiveness of Remote Board Gaming. In *Proceedings of the 9TH ACM International Symposium on Pervasive Displays* (Manchester, United Kingdom) *(PerDis '20)*. Association for Computing Machinery, New York, NY, USA, 79–86. https://doi.org/10.1145/3393712.3395337

[37] Krzysztof Pietroszek. 2019. IRIS: Inter-Reality Interactive Surface. In *25th ACM Symposium on Virtual Reality Software and Technology* (Parramatta, NSW, Australia) *(VRST '19)*. Association for Computing Machinery, New York, NY, USA, Article 77, 2 pages. https://doi.org/10.1145/3359996.3364731

[38] Iulian Radu, Tugce Joy, and Bertrand Schneider. 2021. Virtual Makerspaces: Merging AR/VR/MR to Enable Remote Collaborations in Physical Maker Activities. In *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems* (Yokohama, Japan) *(CHI EA '21)*. Association for Computing Machinery, New York, NY, USA, Article 202, 5 pages. https://doi.org/10.1145/3411763.3451561

[39] Troels Rasmussen, Tiare Feuchtner, Weidong Huang, and Kaj Grønbæk. 2022. Supporting workspace awareness in remote assistance through a flexible multi-camera system and Augmented Reality awareness cues. *Journal of Visual Communication and Image Representation* 89 (2022), 103655. https://doi.org/10.1016/j.jvcir.2022.103655

[40] Holger Regenbrecht, Michael Haller, Joerg Hauber, and Mark Billinghurst. 2006. Carpeno: Interfacing Remote Collaborative Virtual Environments with Table-top Interaction. 10, 2 (2006), 95–107. https://doi.org/10.1007/s10055-006-0045-3

[41] Melissa J. Rogerson, Martin Gibbs, and Wally Smith. 2016. "I Love All the Bits": The Materiality of Boardgames. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 3956–3969. https://doi.org/10.1145/2858036.2858433

[42] David Saffo, Sara Di Bartolomeo, Caglar Yildirim, and Cody Dunne. 2021. Remote and Collaborative Virtual Reality Experiments via Social VR Platforms. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems* (Yokohama, Japan) *(CHI '21)*. Association for Computing Machinery, New York, NY, USA, Article 523, 15 pages. https://doi.org/10.1145/3411764.3445426

[43] Stacey D. Scott, Garth B.D. Shoemaker, and Kori M. Inkpen. 2000. Towards Seamless Support of Natural Collaborative Interactions. In *Proceedings of the Graphics Interface 2000 Conference, May 15-17, 2000, Montr'eal, Qu'ebec, Canada*. 103–110. http://graphicsinterface.org/wp-content/uploads/gi2000-15.pdf

[44] Abigail Sellen, Bill Buxton, and John Arnott. 1992. Using Spatial Cues to Improve Videoconferencing. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Monterey, California, USA) *(CHI '92)*. Association for Computing Machinery, New York, NY, USA, 651–652. https://doi.org/10.1145/142750.143070

[45] Mickael Sereno, Xiyao Wang, Lonni Besancon, Michael J Mcguffin, and Tobias Isenberg. 2020. Collaborative Work in Augmented Reality: A Survey. *IEEE Transactions on Visualization and Computer Graphics* (2020), 1–1. https://doi.org/10.1109/TVCG.2020.3032761

[46] Maximilian Speicher, Brian D. Hall, Ao Yu, Bowen Zhang, Haihua Zhang, Janet Nebeling, and Michael Nebeling. 2018. XD-AR: Challenges and Opportunities in Cross-Device Augmented Reality Application Development. *Proc. ACM Hum.-Comput. Interact.* 2, EICS, Article 7 (jun 2018), 24 pages. https://doi.org/10.1145/3229089

[47] Sebastian Stickert, Hagen Hiller, and Florian Echtler. 2018. Companion - A Software Toolkit for Digitally Aided Pen-and-Paper Tabletop Roleplaying. In *The 31st Annual ACM Symposium on User Interface Software and Technology Adjunct Proceedings* (New York, NY, USA) *(UIST '18 Adjunct)*. ACM, 48–50. https://doi.org/10.1145/3266037.3266097

[48] Jian Sun and Holger Regenbrecht. 2007. Implementing Three-party Desktop Videoconferencing. In *Proceedings of the 19th Australasian Conference on Computer-Human Interaction: Entertaining User Interfaces* (New York, NY, USA) *(OZCHI '07)*. ACM, 95–102. https://doi.org/10.1145/1324892.1324910

[49] Anthony Tang, Carman Neustaedter, and Saul Greenberg. 2007. VideoArms: Embodiments for Mixed Presence Groupware. In *People and Computers XX — Engage*, Nick Bryan-Kinns, Ann Blanford, Paul Curzon, and Laurence Nigay (Eds.). Springer London, London, 85–102.

[50] Franco Tecchia, Leila Alem, and Weidong Huang. 2012. 3D Helping Hands: A Gesture Based MR System for Remote Collaboration. In *Proceedings of the 11th ACM SIGGRAPH International Conference on Virtual-Reality Continuum and Its Applications in Industry* (Singapore, Singapore) *(VRCAI '12)*. Association for Computing Machinery, New York, NY, USA, 323–328. https://doi.org/10.1145/2407516.2407590

[51] Baris Unver, Sarah A. McRoberts, Sabirat Rubya, Haiwei Ma, Zuoyi Zhang, and Svetlana Yarosh. 2016. ShareTable Application for HP Sprout. In *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems (CHI EA '16)*. ACM, New York, NY, USA, 3784–3787. https://doi.org/10.1145/2851581.2890252

[52] Jun Wei, Xuan Wang, Roshan Lalintha Peiris, Yongsoon Choi, Xavier Roman Martinez, Remi Tache, Jeffrey Tzu Kwan Valino Koh, Veronica Halupka, and Adrian David Cheok. 2011. CoDine: An Interactive Multi-sensory System for Remote Dining. In *Proceedings of the 13th International Conference on Ubiquitous Computing* (New York, NY, USA) *(UbiComp '11)*. ACM, 21–30. https://doi.org/10.1145/2030112.2030116

[53] Andrew Wilson and Daniel Robbins. 2007. PlayTogether: Playing Games across Multiple Interactive Tabletops. In *IUI Workshop on Tangible Play: Research and Design for Tangible and Tabletop Games (IUI '07)*. http://research.microsoft.com/en-us/um/people/awilson/publications/WilsonIUI2007/WilsonIUI2007.html

[54] Andrew D. Wilson. 2005. PlayAnywhere: A Compact Interactive Tabletop Projection-Vision System. In *Proceedings of the 18th Annual ACM Symposium on User Interface Software and Technology* (Seattle, WA, USA) *(UIST '05)*. Association for Computing Machinery, New York, NY, USA, 83–92. https://doi.org/10.1145/1095034.1095047

[55] Jacob O. Wobbrock and Julie A. Kientz. 2016. Research Contributions in Human-Computer Interaction. *Interactions* 23, 3 (apr 2016), 38–44. https://doi.org/10.1145/2907069

[56] Xiao Xiao, Paula Aguilera, Jonathan Williams, and Hiroshi Ishii. 2013. MirrorFugue III: Conjuring the Recorded Pianist. In *CHI '13 Extended Abstracts on Human Factors in Computing Systems* (Paris, France) *(CHI EA '13)*. Association for Computing Machinery, New York, NY, USA, 2891–2892. https://doi.org/10.1145/2468356.2479564

[57] Ying Yang, Tim Dwyer, Michael Wybrow, Benjamin Lee, Maxime Cordeil, Mark Billinghurst, and Bruce H. Thomas. 2022. Towards Immersive Collaborative Sensemaking. *Proc. ACM Hum.-Comput. Interact.* 6, ISS, Article 588 (nov 2022), 25 pages. https://doi.org/10.1145/3567741

[58] Svetlana Yarosh, Stephen Cuzzort, Hendrik Müller, and Gregory D. Abowd. 2009. Developing a Media Space for Remote Synchronous Parent-child Interaction. In *Proceedings of the 8th International Conference on Interaction Design and Children* (New York, NY, USA) *(IDC '09)*. ACM, 97–105. https://doi.org/10.1145/1551788.1551806

[59] Svetlana Yarosh, Anthony Tang, Sanika Mokashi, and Gregory D. Abowd. 2013. "Almost Touching": Parent-child Remote Communication Using the Sharetable System. In *Proceedings of the 2013 Conference on Computer Supported Cooperative Work* (New York, NY, USA) *(CSCW '13)*. ACM, 181–192. https://doi.org/10.1145/2441776.2441798

[60] Ye Yuan, Jan Cao, Ruotong Wang, and Svetlana Yarosh. 2021. Tabletop Games in the Age of Remote Collaboration: Design Opportunities for a Socially Connected Game Experience. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems* (Yokohama, Japan) *(CHI '21)*. Association for Computing Machinery, New York, NY, USA, Article 436, 14 pages. https://doi.org/10.1145/3411764.3445512

[61] Jakob Zillner, Christoph Rhemann, Shahram Izadi, and Michael Haller. 2014. 3D-Board: A Whole-Body Remote Collaborative Whiteboard. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology (UIST '14)*. ACM, New York, NY, USA, 471–479. https://doi.org/10.1145/2642918.2647393

Received 2023-07-01; accepted 2023-09-22