SonifyAR: Context-Aware Sound Generation in Augmented Reality

Xia Su[*]
University of Washington
Seattle, Washington, USA
xiasu@cs.washington.edu

Jon E. Froehlich
University of Washington
Seattle, Washington, USA
jonf@cs.washington.edu

Eunyee Koh
Adobe Research
San Jose, California, USA
eunyee@adobe.com

Chang Xiao
Adobe Research
San Jose, California, USA
cxiao@adobe.com

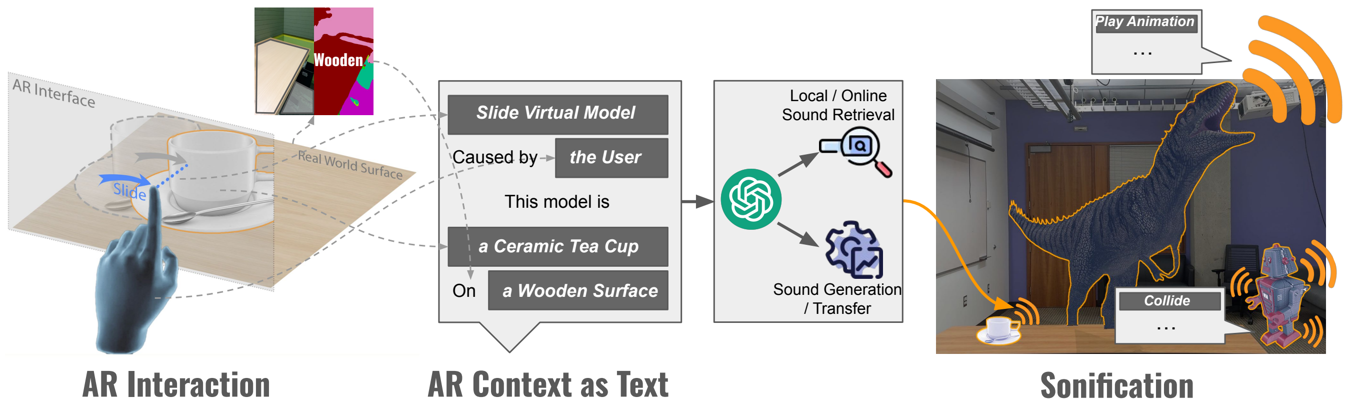


Figure 1: SonifyAR is a custom AR sound authoring pipeline that generates context-matching sounds for AR events *in situ* using generative AI. For example, imagine sliding an AR tea cup across a real-world surface such as a wood table. SonifyAR observes this user action (a slide gesture), the action source (the user), and the action target (a virtual ceramic teacup), infers scene information such as the surface material (a wood table), and uses a custom AI backend to *recommend, retrieve, generate,* or *sound-style transfer* sound effects.

ABSTRACT

Sound plays a crucial role in enhancing user experience and immersiveness in Augmented Reality (AR). However, current platforms lack support for AR sound authoring due to limited interaction types, challenges in collecting and specifying context information, and difficulty in acquiring matching sound assets. We present SonifyAR, an LLM-based AR sound authoring system that generates context-aware sound effects for AR experiences. SonifyAR expands the current design space of AR sound and implements a *Programming by Demonstration* (PbD) pipeline to automatically collect contextual information of AR events, including virtual-content-semantics and real-world context. This context information is then processed by a large language model to acquire sound effects with *Recommendation*, *Retrieval*, *Generation*, and *Transfer* methods. To evaluate the usability and performance of our system, we conducted a user study with eight participants and created five example applications, including an AR-based science experiment, and an assistive application for low-vision AR users.

[*]Major work completed during an internship at Adobe Research.

UIST '24, October 13–16, 2024, Pittsburgh, PA, USA
© 2024 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-0628-8/24/10
https://doi.org/10.1145/3654777.3676406

CCS CONCEPTS

• **Human-centered computing → Interaction design**; **Interactive systems and tools**.

KEYWORDS

Mixed Reality, Sound, Augmented Reality, Authoring Tool

ACM Reference Format:
Xia Su, Jon E. Froehlich, Eunyee Koh, and Chang Xiao. 2024. SonifyAR: Context-Aware Sound Generation in Augmented Reality. In *The 37th Annual ACM Symposium on User Interface Software and Technology (UIST '24), October 13–16, 2024, Pittsburgh, PA, USA.* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3654777.3676406

1 INTRODUCTION

Sound is a critical but often overlooked element in Augmented Reality (AR). AR-based sound can improve immersion and overall user experience [50] and support depth perception and task completion [66], search and navigation [52], and even assistance for people with low vision [49]. Despite its importance, current AR authoring platforms like *Reality Composer* [9], *Adobe Aero* [5], and *Unity Mars* [60] provide only rudimentary sound support. Specifically, we identified three critical gaps with existing AR authoring tools:

(1) **Limited real-world context.** Existing systems typically support action triggers linked to virtual objects in AR but lack support for real-world contextual information (*e.g.*, a virtual toy robot traversing diverse indoor surfaces like wood, carpet, or glass.).
(2) **Limited interaction specification.** Existing systems provide only pre-defined interaction triggers like *"Tap"* and *"Proximity Enter"*. This limits a creator's ability to specify interactions outside the provided options, especially those that involve environmental context (*e.g.,* the user "slides" virtual chalk on a real-world blackboard).
(3) **Limited sound sources.** Existing systems are limited by the sound assets available in their libraries and the scarcity of suitable sound resources online. Thus, AR authors struggle to find appropriate sounds for distinct AR events (*e.g.,* reproducing the wing flutter of a virtual dragonfly or simulating the eating sound of a virtual dinosaur).

To address these challenges, we present *SonifyAR*: a context-aware AR sound authoring system using *GPT-4* [42] and a *text2audio* diffusion model called *AudioLDM* [36]. SonifyAR makes the following key technical advancements:

First, context collection using PbD. SonifyAR adopts a *Programming by Demonstration* (PbD) [32] pipeline to simplify the specification of complex AR interactions. The PbD pipeline enables users to demonstrate AR sound interactions while the system automatically detects the action and collects context information. For example, if a creator wants to sonify the stomping of a walking robot, they can position the virtual robot on the target (physical) surface. As the robot walks, the collision between robot's feet and the surface, as well as the context information like the robot's attributes and the surface's material, is captured by SonifyAR .

Second, context as text. To utilize AR context information stemming from multiple sources (*e.g.* user action, virtual object, real-world environment) and in different formats (*e.g.* categorical, 3D shape, image), we use text as the universal medium to encompass all context information. For example, we generate an LLM prompt using the following template: "*This event is [Event Type], caused by [Source]. This event casts on [Target Object]. [Additional Information on Involved Entities]*".

Finally, LLM-based sound acquisition. We integrate a suite of four sound acquisition methods: *recommend*, *retrieve*, *generate*, and *transfer*, all controlled by the underlying LLM. For each AR sound interaction, the context information as text is fed to the LLM for processing. The LLM then provides text prompts that control the suite of four sound acquisition methods, which automatically provide matching sound assets for the AR interaction.

To author AR sound with SonifyAR, the creator initiates AR interactions and the system automatically lists sound options based on context. For example, imagine trying to make a virtual tea cup chime accordingly when being slid on a table (Figure 1). In traditional approaches, the creator would need to manually specify this sliding action and find a sound asset to match the chiming ceramic cup. But with SonifyAR, the creator could demonstrate this sliding action and the chime sounds are generated automatically for the creator to choose from.

To explore the potential of SonifyAR, we conducted two evaluations: first, a qualitative user study with eight designers to examine the usability of and reactions to the SonifyAR prototype; second, we apply SonifyAR across five key user scenarios: from AR education to improving AR headset safety. Our user study highlights the potential of SonifyAR and the generated sounds while identifying key areas of improvement such as sound quality and interface design. Our application scenarios demonstrate the breadth and potential of SonifyAR incorporating automatically acquired sounds seamlessly into various AR experiences—a practice that would otherwise take significant manual time and effort.

Our work makes three primary contributions: (1) a design space for AR sound authoring tools, which highlights existing gaps in the literature; (2) a novel context-aware AR sound authoring pipeline using generative AI, called SonifyAR, that incorporates sensed environmental cues like surface material, virtual object semantics, and user action; (3) findings from a user study with eight designers and five application examples that demonstrate SonifyAR's potential and key advancements in this space. To our knowledge, we are the first system to offer automatic, in-context sound authoring for AR.

2 RELATED WORK

We cover prior work in AR sound authoring, AI-based sound generation techniques, and context-awareness in AR.

2.1 AR Sound Authoring

Sound in AR provides many benefits from directing the user's attention and enhancing immersion [50, 66] to creating interactive time-varying experiences and increasing accessibility [52, 53]. However, as mentioned, the current landscape of AR authoring tools reveals deficiencies in how sound is supported.

Current tools highlight a variety of AR sound authoring methods. [38, 39]. Some—like *Adobe Aero* [5] and *Apple Reality Composer* [9]—are easy to use, require no coding skills, and can only attach user-provided sound with specific AR event triggers [1, 3, 5, 9]. Others—like *Unity* [59] and *Unreal Engine* [18]—are heavily code-based and require significant technical skills, but provide substantial flexibility in designing AR sound interactions, including the ability to set parameters like sound decay and code-specific conditions for activating sound assets. One common unifying approach *Programming by Specification* (PbS) methodology [38], where creators define AR sound content and play conditions during the design phase. This allows creators to precisely define the conditions for triggers and the consequent sound effects.

As mentioned in introduction, however, critical gaps exist. These gaps are partially due to the limitation of the PbS methodology, since the triggers specified in the design stage naturally lack richness in real-world context, which is crucial for AR sound realism. Also, the trade-offs between supported trigger types and the tool's usability make most tools lack coverage for many sound-producing AR events. Additionally, all sound assets need to be manually selected or created, tasking creators with sourcing suitable sound effects to match the specific triggers. The SonifyAR system aims to address these gaps.

2.2 Sound Acquisition

Although no work has explored sound generation with AR events as input, sound generation conditioned from other input modalities like images, 3D models, text, and videos has been widely explored. Various cross-modal generative models like RNN [65], GAN [16, 19, 30], VAE [14], and most recently diffusion models [28, 36, 64], are utilized to generate sound. Despite different model architectures, one common training goal is the mapping between latent representations of input modalities and output sound. Thus, sounds can be generated from prompts like images [54], videos [19, 65], and text [36, 64]. For example, *FoleyGAN* [19] conditions action sequences of input videos to generate visually aligned, realistic soundtracks. *AudioLDM* [36] utilizes the CLAP-based [63] latent space to embed input text and generate matching sound with a diffusion model. Unlike cross-modal generative methods, another promising approach is physics-based sound synthesis [15, 23, 24, 37, 45, 47, 51]. For example, the spring-mass model [45]—recently improved with deep learning [23]—can provide realistic sound simulation given a 3D model and material properties.

Besides synthesis, retrieval methods can also acquire matching sound for input conditions. For example, Oncescu *et al.* [41] trained cross-modal embedding models to support text-to-audio retrieval. Multi-modal models like VAST [10] also support text-to-audio retrieval that could potentially be used to provide sound assets for AR experiences. Recent work, called *Soundify* [34], utilizes retrieved sound assets to sonify videos. The authors claim that although sound retrieval does not ensure coverage for all input descriptions, the retrieved sound assets usually surpass the synthesis results in terms of quality. In this case, combining generation and retrieval in the sound acquisition pipeline seems to be a natural method that integrates the merits of both. We explore this possibility with SonifyAR.

Unlike other existing sound acquisition methods, AR poses unique challenges in the matching of input condition and output sound. The multi-modal nature of AR interactions, including the information of the virtual object (*e.g.* a toy robot with a walking animation), real environment (*e.g.* a living room with a wooden table), and user action (*e.g.* user taps on a virtual model), needs to be carefully processed into the sound acquisition pipeline to result in matching sound assets.

2.3 Context-awareness in AR

In AR, context-awareness is an important property that allows AR content to be dynamically adjusted according to context information, such as location, scene semantics, and human factors. Context-awareness aims to adapt an application not only to the user but also to their environment and specific needs with the goal of improved usability and immersive [29]. Previous research has explored various aspects of context-awareness in AR, highlighting its importance across different applications.

For example, the popular AR game *Pokemón Go*[40] shows game contents based on user's physical location. A more common practice in AR research is adapting virtual contents to scene semantics—essentially understanding the objects and their locations in the user's environment. Qian *et al.* [43] introduced an authoring system that helps user create adaptive AR experiences to surrounding scene semantics. Lang *et al.* [31] analyzes scene semantics to guide the positioning of virtual agents. Similarly, Liang *et al.* [33] places virtual pets into AR based on scene geometry and semantics. This realm of similar work [20, 58] usually captures and analyzes the surrounding environment and designs algorithms that determines possible interaction between virtual content and real-world scene. Lindbauer *et al.* [35] push the AR context-awareness further by including task and cognitive load as context to adjust level of detail of AR virtual interface.

Although there has been substantial discussion in the realm of AR context-awareness, we observed that prior work generally focuses on the visual modality of AR, neglecting sound. Additionally, existing work primarily concentrates on adjusting and arranging virtual content rather than generating new content. We aim to fill this gap by designing a sound generation pipeline that processes context information to produce matching sound assets.

3 THE DESIGN OF SONIFYAR

To design an effective sound authoring system that generates matching sounds for AR interaction, three main research questions need to be answered.

- What are the target AR interactions that should produce sound effects?
- How can we efficiently acquire sound assets for these AR interactions?
- How can these AR interactions be specified and represented in the authoring process?

We aim to elaborate on these questions and the respective design rationale in this section.

3.1 The Triad of User, Virtuality, and Reality

We began by examining the design space of AR interaction sounds. Drawing on Jain *et al.'s* [22] sound taxonomy in VR, we attempted a similar analysis for AR sound interaction. We first investigated several widely used code-free AR authoring tools for their AR sound capability, including *Adobe Aero* [5], *Apple Reality Composer* [9], and *Halo AR* [3]. We found that these tools support AR sound related to either *virtual content interaction* or *AR-based spatial anchors* (*e.g.*,play a sound when a user gets close to a certain position, scans a QR code, or taps on a virtual model).

The AR sound research literature [13, 17, 46], however, envisions a much broader scope. Rakkolainen *et al.* [46] classified AR audio into five classes: *ambient*, *directional*, *musical*, *speech*, and *noise.* Dam *et al.'s* [13] taxonomy of *Audio Augmented Reality* (AAR), defines AR-based sound across *Environment Connected* (sound maintaining awareness and interaction with physical environment), *Goal Directed* (sound assisting users' primary goal), and *Context Adapted*

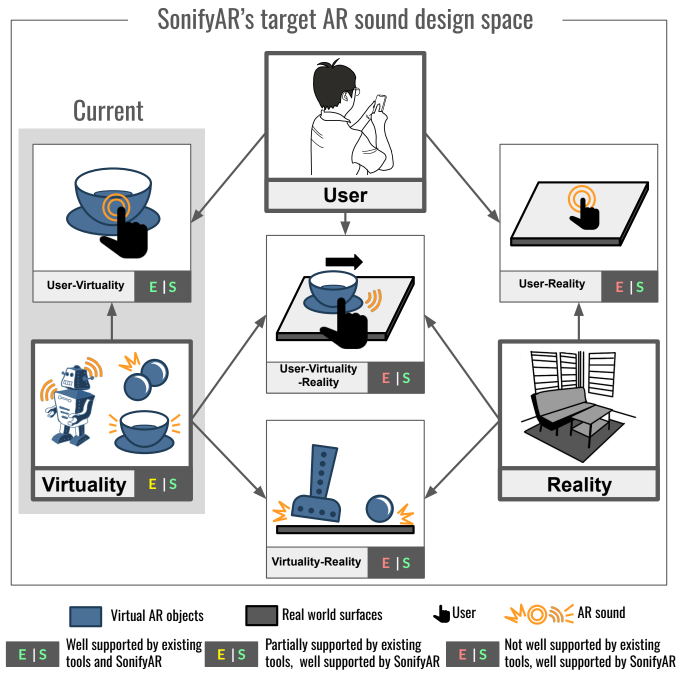

SonifyAR's target AR sound design space
Current
User
User-Virtuality
E | S
User-Reality
E | S
User-Virtuality -Reality
E | S
Virtuality
E | S
Reality
Virtuality-Reality
E | S
Virtual AR objects
Real world surfaces
User
AR sound
E | S
Well supported by existing tools and SonifyAR
E | S
Partially supported by existing tools, well supported by SonifyAR
E | S
Not well supported by existing tools, well supported by SonifyAR

Figure 2: The sound-producing opportunities in the triad of User, Virtuality and Reality.

(sound adaptive to users' immediate reality), as well as their intersection. This taxonomy emphasizes that AR sound is often rooted in contextual information from the real world environment and user intention.

Drawing on this prior work, we introduce a novel AR sound design space that helps emphasize the sonic interplay between three key factors of AR experiences: *reality*, denoting the physical environment in the real world that hosts the AR experience; *virtuality*, referring to the virtual object(s) placed in the AR experience; and *the user*, representing the individual interacting with the AR experience. Below, we expand on each element of this proposed design space and indicate current levels of support in the existing code-free AR sound authoring tools [3, 5, 9].

Virtuality (*partially supported*): *Virtuality* refers to sounds that accompany AR events involving only *virtual objects*. This could be bound to an object's change of status (*e.g.,* a notifying sound when a virtual object shows up), from an object's animated behavior (*e.g.,* the mechanical noise made by a virtual dinosaur roaring), or the interaction between multiple virtual objects (*e.g.,* two virtual balls clacking when they collide). Current authoring tools support status changes and animation playing as sound-initiating triggers, but do not support sound for interactions between virtual objects.

User-Virtuality *(well supported)*: *User-Virtuality* are the corresponding sounds that react to the users' actions with virtual objects (*e.g.,* a virtual dog barks when users tap on or gets close to it). In all of our investigated AR creation tools, these user-initiated triggers like *"Tap"* and *"Proximity Enter"* are well supported.

User-Reality *(not well supported)*: *User-Reality* refers to sounds that accompany user interactions with the physical environment via their AR devices. This sound feedback can improve users' understanding of the surrounding environment (*e.g.,* an appropriate tap sound when users tap on a real-world table surface via their phone screen). None of the AR authoring tools we investigated support user-reality actions as triggers.

Virtuality-Reality *(not well supported)*: *Virtuality-Reality* denotes sound feedback that plays a crucial role in enhancing realism when virtual objects interact with the real-world environment (*e.g.,* the crisp stomping sound of a virtual robot when it walks on a real-world glass surface). Although physics simulations between virtual models and real-world surfaces have been supported, their sound feedback remains unexplored in AR authoring tools.

User-Virtuality-Reality *(not well supported)*: *User-Virtuality-Reality* are the sounds that accompany user actions involving both virtual and real elements. For example, the material-aware sliding sounds when users apply a virtual scraper on different real-world surfaces (*e.g.,* a wooden table, painted wall, or glass window). None of our investigated authoring tools can support the specification of this complex interaction.

Others: We exclude the domains of **Reality** and **User** from the enumeration since our interest lies in events where the AR system (*e.g.* smartphone or AR headset) generate the sound. Thus, sounds naturally generated by user and reality (*e.g.,* a user physically knocking on a wooden table) are excluded from our discussion.

Based on the above analysis and as illustrated in Fig. 2, existing AR authoring tools fail to offer functional AR sound authoring support for many of the identified dimensions, especially those involving *Reality*. SonifyAR aims to address this gap.

3.2 Acquisition of AR Sound

While the above design space helps highlight the dimensions of AR sound and current authoring support, below we reflect on key challenges related to the limitations of sound acquisition methods in terms of handling these different dimensions. For example, a mechanical noise of a virtual robot (virtuality) can be easily sourced from local or online sound databases. However, more specific sounds, like a virtual steel ball hitting a physical concrete wall surface or a virtual racecar jumping into a backyard pool (virtuality-reality) requires physical world understanding and material awareness. The space of possible sound effects here is near infinite. Thus, in such cases, generating sounds using text-to-sound models is preferable. Conversely, certain highly recognizable sounds, like a dog barking, pose difficulties for current generative models, often resulting in outputs that are unsatisfactory and noisy. The complexity in user needs and trade-offs in different methods suggests that the combination of both generation and retrieval could serve as a practical solution.

3.3 Event Representation

To combine generation and retrieval methods, one challenge is how to condition this suite of different sound acquisition methods. While visual formats such as images [54] and videos [19, 65] have been widely used as inputs for sound generation models, they fall short in fully capturing AR context information, particularly user actions. To overcome this, we explore the use of text as a universal representation for sound acquisition inputs, which has the following benefits. First, text can sufficiently and precisely convey necessary context information and the specifics of sound-generating events (*e.g.,"User slides a ceramic cup on a wooden surface"*). Second, given that the AR experience operates within a hardware-software system with multi-modal sensors (*e.g.,* camera, IMU, GPS), we can leverage this system to monitor and log events within the AR experience and summarize them into descriptive text. Third, recent advancements, as demonstrated by works like *HuggingGPT* [55] which uses a LLM to control AI tasks, showcase the practicality of using LLMs as controllers to process text for sound generation.

4 SONIFYAR IMPLEMENTATION

Building on the above design space, we introduce SonifyAR, a novel PbD sound authoring framework that uses context recognition and generative AI to create personalized, context-sensitive sounds for AR interactions (Figure 3). SonifyAR consists of three primary components : (1) *Event Textualization*: Every user action is recognized as an AR event, and the context of these events is transformed into textual descriptions. (2) *Sound Acquisition*: An LLM-controlled sound acquisition process that utilize four methods to produces sound effects based on the context. (3) *User Interface*: An interface that allows users to experience the AR scene and provides the capability to view, modify, and test sound effects for AR events.

4.1 Event Textualization

In SonifyAR, we utilize a PbD authoring framework to capture potential sound-producing AR events and compile the context information into text. When users interact with the AR space, SonifyAR captures AR events that can lead to sound feedback. Here we define an AR event as a user action and its subsequent result (*e.g.,* when a user taps a virtual object to trigger its animation). For each AR event, we describe it with an event type (*e.g.,* tapping an object), action source (*e.g.,* user or virtual object), and action target (*e.g.,* virtual object or real-world plane). Information about the involved parties, such as virtual objects or real-world planes, are also included in the context information. This collected event data is then transformed into text with a template.

Event Types. To support a broad range of AR sound interactions, we implemented six event types:

(1) *Tap Real World Structure:* A user taps on a real-world structure through the AR interface.
(2) *Slide:* A user holds the virtual object and slides it on a real-world surface.
(3) *Collide:* A virtual object collides with another virtual object or with a real-world surface. We implemented a specialized collision detection mechanism as introduced in subsection 4.4
(4) *Show Up:* A virtual object shows up in the AR experience.
(5) *Tap Virtual Objects:* A user taps on a virtual object through the AR interface.
(6) *Play Animation:* A virtual object plays an animation.

Scene Context Understanding. SonifyAR utilizes ARKit [6]'s plane detection functionality [7] to assess the surrounding environment. Since most AR experiences are anchored to planes, the detected plane information serves as a crucial context. To enrich such information, we employ the *Deep Material Segmentation* model [61] to segment the scene and identify the material of planes (Figure 3), which helps produce realistic sound effects when an AR event involves this plane. Currently, we support six materials: wood, carpet, concrete, paper, metal, and glass. Surfaces that do not fall into one of these six categories are labeled as *unknown surface.*

Virtual Object Understanding Virtual object semantics are also crucial for sound generation. To ensure sound assets align with the virtual object's material and its state, we collect a text description for all the virtual objects (*e.g.,* "*This model is a toy robot made of metal.*") and their animations (*e.g.,* "*A toy robot walks.*"). These descriptions can be provided by the asset creator or, if necessary, we prompt the user to add relevant details. Any AR event involving the virtual object or its animation will incorporate these text descriptions.

Event-to-text To aggregate the multi-source context information described above, we employ a simple text template that will feed into an LLM-based backend: "*This event is [Event Type], caused by [Source]. This event casts on [Target Object]. [Additional Information on Involved Entities].*" *Event Type* is the type of event described by the aforementioned type names; *Source* refers to the subject of the event, which could be the user when the event is directly triggered by user, or virtual object when it interacts with real-world environment; *Target Object* is the object of the event (*e.g.,* the plane that gets tapped on or the animation that gets played.). Finally, *Additional Information on Involved Entities* include details that further elucidate the source object, the triggered event, and the target object. Examples include material descriptions and animation details. We leave the corresponding entity field blank when events are missing a target object or additional information.

During user interactions, the system actively monitors and logs events happening in AR space. The context information is fetched and plugged into the text template, crafting a coherent description that reflects the user's interaction within the AR environment.

4.2 Sound Acquisition

SonifyAR utilizes GPT4[42] to automatically retrieve or generate context-matching sound assets of an AR event. Inspired by *HuggingGPT* [55] and *Visual ChatGPT* [62], we utilize the LLM as a controller of multiple sound authoring methods. The LLM takes the text description of the event as input and replies with commands for multiple sound acquisition methods (Figure 5). At our current stage, we support four major sound authoring methods: local recommendation, online retrieval, *text2sound* generation, and text-guided sound transfer. All sound authoring processes listed below operate concurrently in the backend. This ensures an uninterrupted AR experience.

Local Recommendation. The LLM can recommend sound assets stored in the local database based on semantics in the event description. Similar to other AR authoring tools, we collect a set of sound effects from *Adobe Audition*'s library[4], each labeled with a descriptive filename, like "*Crash Aluminum Tray Bang*" or "*Liquid Mud Suction*". The entire catalog of sound filenames is provided to LLM. When given the event context, the LLM recommends the top five most suitable local sound effects based on their filenames. It then returns the selected sound effect in the format of `method1recommend:FILENAME`, where `FILENAME` is replaced by the actual filename. Upon receiving, SonifyAR parses the filename and

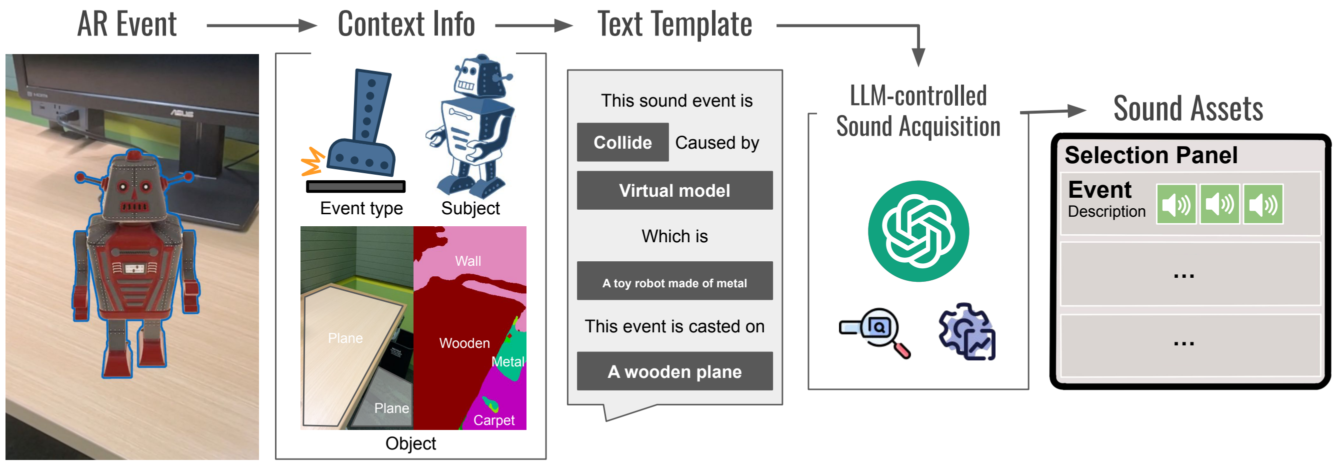


Figure 3: Overview of the pipeline of SonifyAR. Our system monitors and logs context information of AR events, which includes the event type, the subjects and objects (virtual or real-world), and the attributes of the involved elements like their materials. This information is compiled into a text template and then processed by our LLM controller to acquire sound assets. The results are subsequently presented in our selection panel.

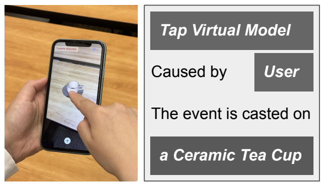


Figure 4: SonifyAR's event textualization. Left: a user tapping on a virtual cup through the SonifyAR interface; Right: the textual context information extracted by SonifyAR's internal PbD framework.

adds the top corresponding sound as one of the sound options for the event. Users can also long-hold options in the UI to reveal other top recommendation results.

Online Retrieval. We also expand our retrieval capability with an online sound asset database called *FreeSound* [2]. We use the FreeSound API, which returns an N-best list of matching sound effects based on a given query. The queries are condensed versions of full event descriptions generated by the LLM, returned in the format `method2retrieval:PROMPT`, with `PROMPT` replaced by the specific search query. The returned results are JSON strings. We select the top five matches from the entire set of sound effects returned by the API, which are then downloaded and presented to the user as sound options.

Sound Generation. Beyond recommendation and retrieval methods, we also use the text prompt to generate custom sound effects using an audio diffusion model *AudioLDM* [36]. We ask the LLM to compress the event text description into a shortened generation prompt: using the format `method3generation:PROMPT`, where the `PROMPT` would be replaced with the generation prompt. Upon receiving such a command from the LLM, SonifyAR sends the prompt

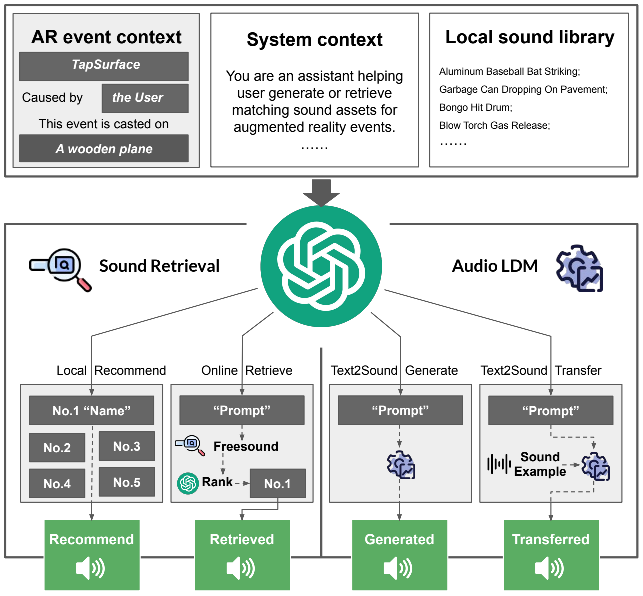


Figure 5: SonifyAR's sound acquisition pipeline.

to the AudioLDM model, requesting *text2sound* generation. These newly generated sounds are then shown in the UI as sound options.

Sound-style Transfer. Besides text2audio generation, *AudioLDM* is also capable of performing text-based sound style transfer. Specifically, for events like tapping, sliding, or colliding, instead of generating a new sound from scratch, SonifyAR uses a default sound effect and initiates a style transfer operation with a text prompt provided by the LLM (*e.g.,* transfer a general tap sound to tapping on glass). This approach allows the output sounds to match the length and rhythm of the input sounds, so that they can be well-timed with actions. Furthermore, when users wish to further modify any provided sound assets, SonifyAR offers text-based style transfer as a fine-tuning and customizing option.

4.3 User Interface

As a PbD authoring framework, the SonifyAR application invites users to explore freely with an interactive AR experience (Figure 6 left). Users can interact with the scene, performing actions like moving virtual objects or interacting with the real-world surfaces. When an AR event is detected, a text label appears on the top-left screen to inform the user (Figure 6A) and the sound acquisition component is activated to generate candidate sound effects for the detected events. By clicking the "*authoring panel*" button (Figure 6D), users can prompt an editing interface (Figure 6F) that overlays the AR scene. The UI includes all detected AR events (Figure 6G) and the corresponding generated sound assets (Figure 6H). Users can click on a sound effect to preview and double click to select and activate it. After confirming their choices, users can hide the authoring panel to resume the AR experience and test the selected sounds. The editing interface can always be re-activated to modify choices.

Additionally, users can long press a sound asset to call out a menu with a suite of exploratory options. For recommended or retrieved sounds, the menu includes an option to list all other sound assets in the top five recommendations. For all sound assets, the menu provides a "*style transfer*" and a "*generate similar sounds*" feature, enabling users to style transfer audio effects or to generate similar sounds based on the selected sound. When this feature is selected, users can type a simple text prompt to guide the sound generation process. These choices can be iterated upon or used to explore new sound variations.

4.4 Technical Implementation

SonifyAR is built with Apple's ARKit in Swift 5.7 and Xcode 14, and runs as an iOS application on iPhones. Due to the enhanced performance of plane detection on devices with LiDAR (available in iPhone's Pro lineup starting from iPhone Pro 12), we developed and tested the app on an iPhone 13 Pro Max running iOS 16. We implemented the collision detection between virtual objects and planes with ARKit's physics simulation [8]. To enable collision detection for animated virtual object parts (*e.g.* the stomping of a robot foot on the plane), we bind colliders to the joints of the virtual object's animated bones.

For the LLM-based backend, we use GPT-4.0 [42]. The text prompt is shown in Appendix A. For material segmentation, we use the *Dense Material Segmentation* (DMS) model [61] for its detailed material labeling, fast processing time, and high accuracy. For sound generation, we use the *AudioLDM* model [36] due to its state-of-the-art performance and versatility in text2sound generation and text-guided sound2sound transfer. All models are hosted on a server with an NVIDIA RTX 4080 GPU and CUDA 12.2.

5 USER STUDY

To examine the usability and authoring performance of the SonifyAR system, we conducted a usability study across eight participants, who used the SonifyAR application in a fixed experimental setting and provided ratings for the system.

5.1 Participants

We recruited eight participants (six male and two female, aged between 26 and 33) via snowball sampling at Adobe with varying experience in AR. Out of the eight participants, six had prior AR experience, and three had specific experience in AR authoring tools.

5.2 Procedure

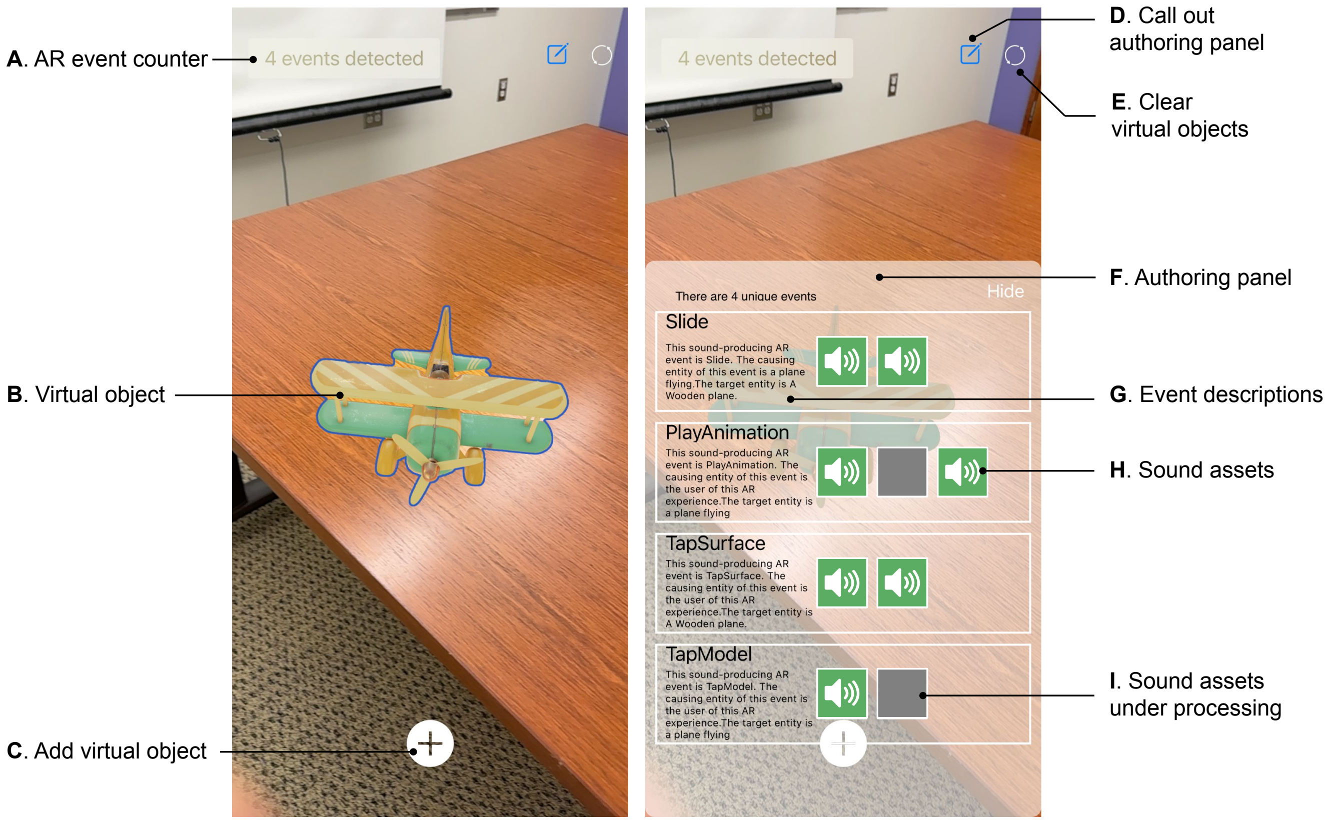
The usability study was conducted in a small meeting room and had four parts: (1) We first introduced our study goal, collected demographic and background information. (2) We then demonstrated SonifyAR with an example AR experience. During this phase, participants had the opportunity to ask any questions regarding SonifyAR's functionalities. (3) Afterwards, participants were asked to independently explore the SonifyAR app. We provided a walking robot model which could be added to indoor surfaces. This model starts walking when tapped. For consistency, we asked all participants to explore this same AR asset and use our automatic sound authoring pipeline to create sound effects with the model until they were satisfied with the results. The entire usage process was screen-recorded and the user operations were logged. (4) Finally, we sought participant feedback regarding the usability, helpfulness and technical performance of SonifyAR. Participants provided reasoning and insights to support their answers. The questions and rating results are shown in Figure 7.

5.3 Results

All participants were able to complete the AR sound authoring task using SonifyAR without difficulty. On average, the authoring process took 406 seconds (*SD*=137s), and the entire study took 35 minutes. Participants tested an average of 56 (*SD*=10.5) sound assets, with an average of 6.4 sounds (*SD*=1.7) assigned to the authoring results.

Participant feedback was also largely positive. All eight participants expressed favorable impressions of the tool. They highly agreed that the SonifyAR tool would be helpful to AR authoring process (Q1, *avg*=6 out of 7, *SD*=0.93). There was also a general willingness for using SonifyAR in their own AR creation practice (Q2, *avg*=6.3, *SD*=1.16). Participants agreed that the sound interaction involving real-world surfaces will improve the immersiveness of AR experiences (Q3, *avg*=6.4, *SD*=1.1). Additionally, they preferred the automated sound authoring process over their prior manual search experiences for sound assets (Q5, *avg*=6.1, *SD*=1.0). The only area for improvement was the quality of the generated sound (Q4, *avg*=4.75, *SD*=1.39). However, this could be enhanced with the introduction of a more advanced sound generative model, which can be easily integrated into our system. The full list of questions and the respective Likert results can be viewed in Figure 7.

Besides the ratings, participants also provided suggestions, especially on the UI and usability. For example, *"I really want more usability features that can help communicate these events"*, and *"I hope there are more associations between the AR events and the sound generation happening in the backend, like some real-time audio hints"*.



There are 4 unique events

Slide

This sound-producing AR event is Slide. The causing entity of this event is a plane flying.The target entity is A Wooden plane.

PlayAnimation

This sound-producing AR event is PlayAnimation. The causing entity of this event is the user of this AR experience.The target entity is a plane flying

TapSurface

This sound-producing AR event is TapSurface. The causing entity of this event is the user of this AR experience.The target entity is A Wooden plane.

TapModel

This sound-producing AR event is TapModel. The causing entity of this event is the user of this AR experience.The target entity is a plane flying

Figure 6: SonifyAR's authoring interface. Left: SonifyAR's phone-based AR interface. Right: SonifyAR's authoring panel.

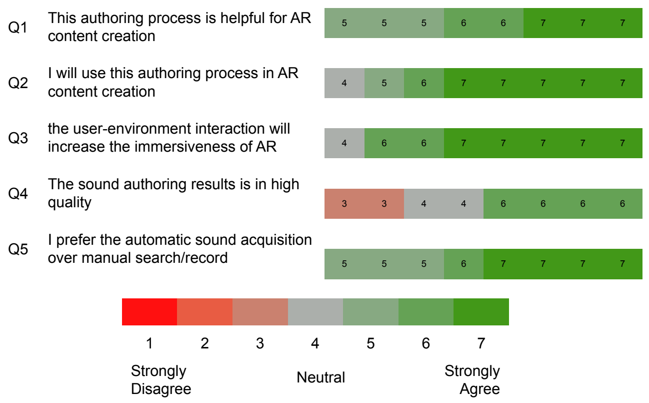


Figure 7: Likert questions and results of our usability study

6 APPLICATIONS

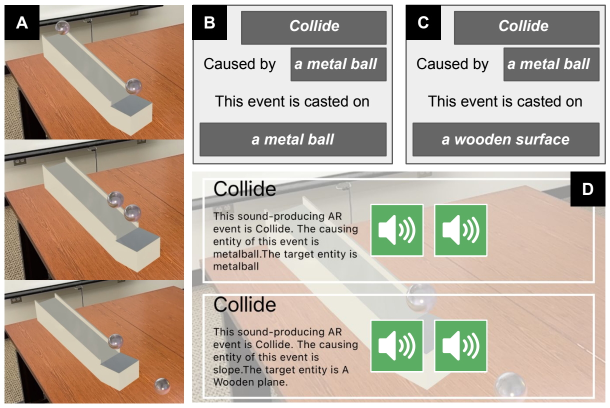
To further explore and demonstrate the potential of our approach, we authored five AR scenarios with SonifyAR. See the supplementary video for additional details and an audio-visual demonstration of the resulting SonifyAR sound effects.

6.1 Education

AR has been widely explored in STEM education (*e.g.,* [11, 25–27, 56, 57]). As existing educational applications try to simulate and visualize physical or chemical phenomenon in AR [21, 44], SonifyAR can improve the blending between the real environment and virtual content through immersive, appropriate sound effects. To showcase such a possibility, we implemented an AR physics experiment using SonifyAR. To help illustrate one of Newton's laws of motion–the conservation of momentum—we created an AR scene with a downward ramp and two metal balls: one at the top and the other at the bottom (Figure 8). Upon initiating the experiment, the top ball rolls down the slope and collides with the bottom ball. This collision causes the bottom ball to move forward and eventually fall, while the top ball comes to a stop. With SonifyAR, collision events are automatically detected and material-specific sound effects, such as the clashing of two metal balls or a metal ball dropping onto table, are seamlessly generated.

6.2 Accessibility

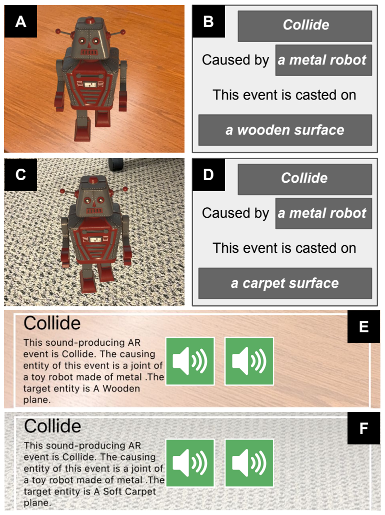
AAR (Audio Augmented Reality) has been explored as accessibility assistance for blind or low vision people [12, 48]. By sonifying real world environments and virtual contents, blind or low vision (BLV) people can better interpret visual information in both reality and virtuality. We envision SonifyAR assisting AR accessibility in



This sound-producing AR event is Collide. The causing entity of this event is metalball.The target entity is metalball

This sound-producing AR event is Collide. The causing entity of this event is slope.The target entity is A Wooden plane.

Figure 8: An AR physics experiment sonified by SonifyAR. (A) The slope and metal ball model and simulation process. (B), (C) Context information of two collisions in the simulation. (D) The acquired sound assets for the collisions.



This sound-producing AR event is Collide. The causing entity of this event is a joint of a toy robot made of metal .The target entity is A Wooden plane.

This sound-producing AR event is Collide. The causing entity of this event is a joint of a toy robot made of metal .The target entity is A Soft Carpet plane.

Figure 9: Robot walking on different surface with different sound. (A), (B) Robot walking on a wooden surface, and the corresponding context information. (C), (D) Robot walking on carpet surface, and the corresponding context information. (E), (F) The acquired sound assets.

three ways. First, by supporting AR sound authoring, SonifyAR encourages creators to add sound effects in AR experiences, which can help BLV users interpret visual content. Second, by providing context-aware AR sound in 3D audio, people with low-vision can better navigate virtual objects in the AR environment [48]. Third, by enabling user to interact with real-world surfaces via AR (*e.g.* tapping on a real world surface), user can explore the surrounding space via the AR interface.

To showcase potential accessibility benefits, we use a toy robot 3D model (Figure 9). As the robot virtually walks from one physical surface to another (*e.g.,* from a wood floor to a carpeted floor), the auto-generated sound effects change appropriately. In this way, a BLV user can better perceive the location and activity of the robot as it is walking.

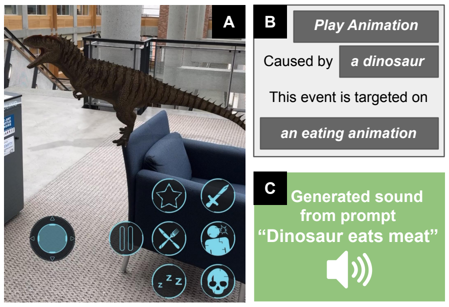

A
B
Play Animation
Caused by
a dinosaur
This event is targeted on
an eating animation
C
Generated sound from prompt "Dinosaur eats meat"

Figure 10: AR dinosaur sonified by SonifyAR. (A) AR Camera app interface with a virtual dinosaur playing eating animation in the scene. (B) The context information of the scene. (C) Text prompt and sound output from SonifyAR's pipeline.

6.3 Sonify Existing Apps

Due to the extensibility of the text template used in SonifyAR's sound acquisition pipeline, it can be easily implemented into existing AR apps. For example, if implemented at the SDK level (*e.g.* ARKit and ARCore), SonifyAR can automatically capture textual description of AR events and provide sonification using the automatic sound acquisition results.

We explore this application possibility with a Wizard of Oz (WoZ) prototype on a phone-based AR application called *ARVid* [1], which is a downloadable AR app that puts 3D animated objects into real world environment. We selected a dinosaur animation as an example (Figure 10) and use WoZ to simulate the direct integration of the SonifyAR pipeline into ARVid app. By manually feeding the context information (*e.g. "The dinosaur is eating"*) into the SonifyAR pipeline, we generated sound assets that well-match the 3D animations.

6.4 Using SonifyAR on an MR Headset

As recent AR/MR headsets become increasingly popular, manufacturers have established safety guidelines, such as setting up safety zone and clear up indoor spaces. We envision SonifyAR contributing to this topic by hinting users of the existence of real-world entities with material-aware sound effects when application windows or virtual object intersect with real world surfaces or objects.

We implement a WoZ prototype based on Apple Vision Pro's user interaction (Figure 11). When using the Vision Pro, a user can actively adjust the position of application windows by hand gestures. SonifyAR can enhance this interaction by generating AR sounds when virtual windows are placed on or intersect with indoor surfaces like walls, desks, and floors. By processing these interactions and applying sound transfers with text prompts such as *"a virtual window bumps into a wall"* and *"a virtual window collides with a carpet"*, SonifyAR aims to improve the permanence of virtual windows and increase awareness of surrounding objects as users navigate through virtual windows.

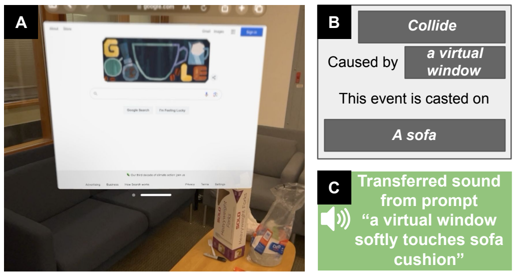

A
B
Collide
Caused by
a virtual window
This event is casted on
A sofa
C
Transferred sound from prompt "a virtual window softly touches sofa cushion"

Figure 11: Sonification of Vision Pro's window operation. (A) User place a virtual window on sofa. (B) Context information of this action. (C) Text prompt and sound output from SonifyAR's pipeline.

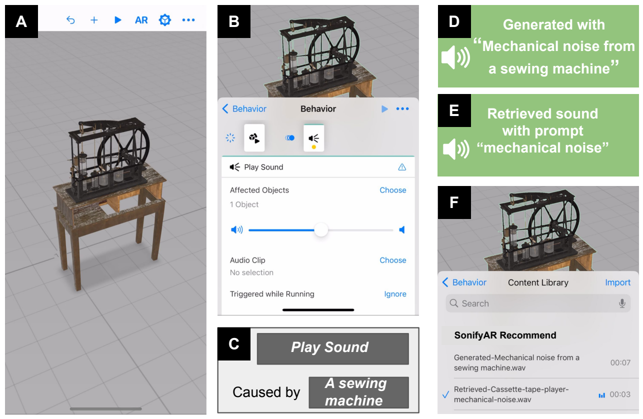

A
B
Behavior
Play Sound
Affected Objects
Choose
1 Object
Audio Clip
No selection
Triggered while Running
Ignore
C
Play Sound
Caused by
A sewing machine
D
Generated with "Mechanical noise from a sewing machine"
E
Retrieved sound with prompt "mechanical noise"
F
Content Library
Import
Search
SonifyAR Recommend
Generated-Mechanical noise from a sewing machine.wav
Retrieved-Cassette-tape-player-mechanical-noise.wav

Figure 12: WoZ example of SonifyAR being applied in Reality Composer's authoring process. (A) A machine model in Reality Composer's authoring interface. (B) Reality Composer's interface for AR sound. (C) Context information of the event in B. (D), (E) Two sound assets generated and retrieved with SonifyAR's pipeline. (F) The acquired sound being used as recommended options.

6.5 Augmenting AR Authoring Processes

Besides serving as a standalone PbD authoring experience, SonifyAR could improve existing AR authoring tools. As mainstream AR authoring tools like Adobe Aero and Apple Reality Composer already represent their AR interaction as text like *"Tap & Play Sound"* and *"Proximity & Jiggle"*, these specifications can be fed into SonifyAR's text-based sound authoring pipeline. In this case, SonifyAR works as a plugin and provides matching sound assets for AR interactions.

We implemented an additional WoZ prototype using Apple's Reality Composer and a 3D model of an animated machine (Figure 12). When authoring the *"Play Sound"* behavior of this model, context information like the model description, triggering condition, and animation description, can be fed into SonifyAR's text-based authoring pipeline. The resulting retrieved/generated sound assets appear at the top of the authoring UI as *"SonifyAR Recommended"*. See supplementary video for more details.

7 DISCUSSION AND FUTURE WORK

In this paper, we designed and implemented an AR sound authoring pipeline that automatically generates sound assets based on context information. We evaluated and identified opportunities in the AR sound interaction space, and implemented a custom LLM+Generative AI pipeline for sound acquisition. We tested the pipeline with a usability study and through a "proof-by-demonstration" across five application scenarios.

7.1 Limitations and Failure Cases

While SonifyAR was positively evaluated in our user study and demonstrated broad application potential, we observed limitations in the system adaptability and encountered some failure cases.

Firstly, since the sound acquisition pipeline relies on an LLM as both the context collector and overall controller, LLM errors can lead to failures. For example, the LLM can hallucinate about the context information and prompt sound generation model with inaccruate text input (*e.g.,* LLM provides prompt "sliding plastic on carpet" when the provided context information is the virtual model, a metal toy robot, slides on wooden floor) that lead to non-matching outputs.

Secondly, although AudioLDM [36] is a state-of-the-art text-to-audio generator and performs well in most cases, it can sometimes produce subpar outputs that are noisy or do not match prompts. Based on our experiences, such performance inconsistency can be hard to predict thus cannot be completely solved by prompt engineering.

Lastly, some AR interactions can be highly time-sensitive (*e.g.,* objects colliding, playing animations that show clear action sequences) and require sound assets to be precisely synchronized with visual content. Currently, we use two methods to synchronize sounds with AR events. First, we provide example sound assets for AR interactions of *"Colliding", "Tapping"* and *"'Sliding'*. By conducting sound style transfer using well-timed example sounds as inputs, the output sounds can be formatted to be in sync with the AR interaction. Secondly, we bind colliders to 3D model skeleton joints, ensuring that the generated sound is played only when a collision is detected between an animated model and real-world surfaces. However, these synchronization method do not apply to 3D assets without skeletons or sounds unrelated to collisions, such as dinosaur roaring and mechanical arm unfolding. Fully addressing this 3D animation-to-audio synchronization challenge is beyond our current scope.

7.2 Future Work

We plan to address the following improvements for future work.

First, our current SonifyAR app is a simplified tool that exclusively supports the authoring of AR sound and uses pre-crafted AR assets as input. For wider applicability that reflects the real-world needs of AR content creation, we recognize the need to incorporate the AR sound authoring pipeline into the entire AR content creation process. Our future goals include integrating the SonifyAR framework with established tools like Reality Composer and Adobe Aero, and also building a Unity SDK to enable usage of our system among professional AR developers. We will also expand the list of AR event types supported by SonifyAR, adding categories such as

"Spin", *"Emphasize"*, *"Expand"*, or *"Jump"*—all of which can easily be incorporated into the text template of SonifyAR.

Secondly, SonifyAR adopts a straightforward process of using a text template to compile context information collected from multiple sources, including virtual object semantics and real world semantics. Currently, we use one material segmentation model to perceive real-world context information. In future developments, we aspire to incorporate more sophisticated computer vision models or multi-modal foundational models, for deeper understanding of both virtual objects and real world environment, enhancing the context acquisition process with greater precision and broader adaptability.

Thirdly, SonifyAR has some basic error-handling features, such as the automatic removal of failed retrieval results and editing options for subpar sound assets (Section 4.3). Additionally, users can correct recognition errors, such as incorrectly recognized materials, by specifying the correct material with text. In future versions of SonifyAR, we plan to incorporate more advanced error-handling mechanisms, such as deploying a sound quality inference model to validate the output and automatically re-prompt the sound generation model when subpar sounds are generated.

Lastly, the current sound generation output of SonifyAR presents opportunities for further refinement. Recognizing the rapid advancements in AI, our system has been designed as a modular framework, ensuring that as more advanced models emerge, they can be seamlessly integrated. Every model (*e.g.*, LLM, CV, Sound Generation) within SonifyAR is replaceable, allowing SonifyAR to easily adapt and stay at the forefront of innovation in this field.

8 CONCLUSION

In this paper, we present SonifyAR, a context-aware sound authoring system designed for sonifying AR events. Our work implements a custom PbD authoring pipeline which enables sound asset acquisition using context information and AI. With SonifyAR, users demonstrate AR interactions and have AR sound assets automatically acquired. Our studies validate the usability and overall performance of our system. Our five application examples further supports the potential of our approach. This research advances literature in AR sound authoring, while also opening up new research avenues for the application of LLM and generative models in AR systems.

ACKNOWLEDGMENTS

This research was mainly conducted during an internship at Adobe Research and also supported by NSF award #1763199. We thank Dr. Cuong Nguyen for his suggestions to our research and our user study participants for their participation.

REFERENCES

[1] [n. d.]. *ARVid - Augmented Reality.* https://apps.apple.com/us/app/arvid-augmented-reality/id1276546297 Accessed on September 24, 2023.

[2] [n. d.]. *Freesound.* https://freesound.org/ Accessed on September 24, 2023.

[3] [n. d.]. *Halo AR.* https://haloar.app/ Accessed on September 24, 2023.

[4] Adobe. 2023. *Adobe Audition Sound Effects Download Page.* https://www.adobe.com/products/audition/offers/AdobeAuditionDLCSFX.html Accessed on Date of Access.

[5] Adobe. Accessed September 11, 2023. Adobe Aero. https://www.adobe.com/products/aero.html.

[6] Apple. 2023. *Apple ARKit Documentation.* https://developer.apple.com/documentation/arkit/ Accessed on Oct 9th, 2023.

[7] Apple. 2023. *ARKit - Tracking and Visualizing Planes.* https://developer.apple.com/documentation/arkit/arkit_in_ios/content_anchors/tracking_and_visualizing_planes Accessed on Oct 9th, 2023.

[8] Apple. 2023. *SceneKit - Physics Simulation.* https://developer.apple.com/documentation/scenekit/physics_simulation Accessed on Oct 9th, 2023.

[9] Apple. Accessed September 11, 2023. Reality Composer. https://apps.apple.com/us/app/reality-composer/id1462358802.

[10] Sihan Chen, Handong Li, Qunbo Wang, Zijia Zhao, Mingzhen Sun, Xinxin Zhu, and Jing Liu. 2023. VAST: A Vision-Audio-Subtitle-Text Omni-Modality Foundation Model and Dataset. arXiv:2305.18500 [cs.CV]

[11] Neil Chulpongsatorn, Mille Skovhus Lunding, Nishan Soni, and Ryo Suzuki. 2023. Augmented Math: Authoring AR-Based Explorable Explanations by Augmenting Static Math Textbooks. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology* (San Francisco, CA, USA) *(UIST '23)*. Association for Computing Machinery, New York, NY, USA, Article 92, 16 pages. https://doi.org/10.1145/3586183.3606827

[12] James M Coughlan and Joshua Miele. 2017. AR4VI: AR as an accessibility tool for people with visual impairments. In *2017 IEEE International Symposium on Mixed and Augmented Reality (ISMAR-Adjunct)*. IEEE, 288–292.

[13] Abhraneil Dam, Arsh Siddiqui, Charles Leclercq, and Myounghoon Jeon. 2024. Taxonomy and definition of audio augmented reality (AAR): A grounded theory study. *International Journal of Human-Computer Studies* 182 (2024), 103179.

[14] Prafulla Dhariwal, Heewoo Jun, Christine Payne, Jong Wook Kim, Alec Radford, and Ilya Sutskever. 2020. Jukebox: A generative model for music. *arXiv preprint arXiv:2005.00341* (2020).

[15] Rodrigo Diaz, Ben Hayes, Charalampos Saitis, György Fazekas, and Mark Sandler. 2022. Rigid-Body Sound Synthesis with Differentiable Modal Resonators. http://arxiv.org/abs/2210.15306 arXiv:2210.15306 [cs, eess].

[16] Jesse Engel, Kumar Krishna Agrawal, Shuo Chen, Ishaan Gulrajani, Chris Donahue, and Adam Roberts. 2019. Gansynth: Adversarial neural audio synthesis. *arXiv preprint arXiv:1902.08710* (2019).

[17] Mohd Ihsan Alimi Mohd Filus and Dayang Rohaya Awang Rambli. 2012. Using non-speech sound as acoustic modality in Augmented Reality environment. In *2012 International Symposium on Computer Applications and Industrial Electronics (ISCAIE)*. IEEE, 79–82.

[18] Epic Games. 2023. *Unreal Engine.* https://www.unrealengine.com/en-US Accessed on Oct 9th, 2023.

[19] Sanchita Ghose and John J Prevost. 2022. Foleygan: Visually guided generative adversarial network-based synchronous sound generation in silent videos. *IEEE Transactions on Multimedia* (2022).

[20] Lei Han, Tian Zheng, Yinheng Zhu, Lan Xu, and Lu Fang. 2020. Live semantic 3d perception for immersive augmented reality. *IEEE transactions on visualization and computer graphics* 26, 5 (2020), 2012–2022.

[21] Ferli Septi Irwansyah, YM Yusuf, Ida Farida, and Muhammad Ali Ramdhani. 2018. Augmented reality (AR) technology on the android operating system in chemistry learning. In *IOP conference series: Materials science and engineering*, Vol. 288. IOP Publishing, 012068.

[22] Dhruv Jain, Sasa Junuzovic, Eyal Ofek, Mike Sinclair, John R. Porter, Chris Yoon, Swetha Machanavajhala, and Meredith Ringel Morris. 2021. A Taxonomy of Sounds in Virtual Reality. In *Proceedings of the 2021 International Conference on Multimodal Interaction*. ACM, Montréal QC Canada, 80–91. https://doi.org/10.1145/3462244.3479946

[23] Xutong Jin, Sheng Li, Tianshu Qu, Dinesh Manocha, and Guoping Wang. 2020. Deep-Modal: Real-Time Impact Sound Synthesis for Arbitrary Shapes. In *Proceedings of the 28th ACM International Conference on Multimedia (MM '20)*. Association for Computing Machinery, New York, NY, USA, 1171–1179. https://doi.org/10.1145/3394171.3413572

[24] Xutong Jin, Sheng Li, Guoping Wang, and Dinesh Manocha. 2022. NeuralSound: learning-based modal sound synthesis with acoustic transfer. *ACM Transactions on Graphics* 41, 4 (July 2022), 1–15. https://doi.org/10.1145/3528223.3530184

[25] Seokbin Kang, Leyla Norooz, Elizabeth Bonsignore, Virginia Byrne, Tamara Clegg, and Jon E. Froehlich. 2019. PrototypAR: Prototyping and Simulating Complex Systems with Paper Craft and Augmented Reality. In *Proceedings of the 18th ACM International Conference on Interaction Design and Children* (Boise, ID, USA) *(IDC '19)*. Association for Computing Machinery, New York, NY, USA, 253–266. https://doi.org/10.1145/3311927.3323135

[26] Seokbin Kang, Leyla Norooz, Vanessa Oguamanam, Angelisa C. Plane, Tamara L. Clegg, and Jon E. Froehlich. 2016. SharedPhys: Live Physiological Sensing, Whole-Body Interaction, and Large-Screen Visualizations to Support Shared Inquiry Experiences. In *Proceedings of the The 15th International Conference on Interaction Design and Children* (Manchester, United Kingdom) *(IDC '16)*. Association for Computing Machinery, New York, NY, USA, 275–287. https://doi.org/10.1145/2930674.2930710

[27] Seokbin Kang, Ekta Shokeen, Virginia L. Byrne, Leyla Norooz, Elizabeth Bonsignore, Caro Williams-Pierce, and Jon E. Froehlich. 2020. ARMath: Augmenting Everyday Life with Math Learning. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–15. https://doi.org/10.1145/3313831.3376252

[28] Zhifeng Kong, Wei Ping, Jiaji Huang, Kexin Zhao, and Bryan Catanzaro. 2020. Diffwave: A versatile diffusion model for audio synthesis. *arXiv preprint arXiv:2009.09761* (2020).

[29] Sarah Krings, Enes Yigitbas, Ivan Jovanovikj, Stefan Sauer, and Gregor Engels. 2020. Development framework for context-aware augmented reality applications. In *Companion Proceedings of the 12th ACM SIGCHI Symposium on Engineering Interactive Computing Systems (EICS '20 Companion)*. Association for Computing Machinery, New York, NY, USA, 1–6. https://doi.org/10.1145/3393672.3398640

[30] Kundan Kumar, Rithesh Kumar, Thibault De Boissiere, Lucas Gestin, Wei Zhen Teoh, Jose Sotelo, Alexandre De Brebisson, Yoshua Bengio, and Aaron C Courville. 2019. Melgan: Generative adversarial networks for conditional waveform synthesis. *Advances in neural information processing systems* 32 (2019).

[31] Yining Lang, Wei Liang, and Lap-Fai Yu. 2019. Virtual agent positioning driven by scene semantics in mixed reality. In *2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. IEEE, 767–775.

[32] Tessa A. Lau and Daniel S. Weld. 1998. Programming by Demonstration: An Inductive Learning Formulation. In *Proceedings of the 4th International Conference on Intelligent User Interfaces* (Los Angeles, California, USA) *(IUI '99)*. Association for Computing Machinery, New York, NY, USA, 145–152. https://doi.org/10.1145/291080.291104

[33] Wei Liang, Xinzhe Yu, Rawan Alghofaili, Yining Lang, and Lap-Fai Yu. 2021. Scene-aware behavior synthesis for virtual pets in mixed reality. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. 1–12.

[34] David Chuan-En Lin, Anastasis Germanidis, Cristóbal Valenzuela, Yining Shi, and Nikolas Martelaro. 2023. Soundify: Matching sound effects to video. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*. 1–13.

[35] David Lindlbauer, Anna Maria Feit, and Otmar Hilliges. 2019. Context-aware online adaptation of mixed reality interfaces. In *Proceedings of the 32nd annual ACM symposium on user interface software and technology*. 147–160.

[36] Haohe Liu, Zehua Chen, Yi Yuan, Xinhao Mei, Xubo Liu, Danilo Mandic, Wenwu Wang, and Mark D Plumbley. 2023. Audioldm: Text-to-audio generation with latent diffusion models. *arXiv preprint arXiv:2301.12503* (2023).

[37] Shiguang Liu and Dinesh Manocha. 2021. Sound Synthesis, Propagation, and Rendering: A Survey. http://arxiv.org/abs/2011.05538 arXiv:2011.05538 [cs].

[38] Kyzyl Monteiro, Ritik Vatsal, Neil Chulpongsatorn, Aman Parnami, and Ryo Suzuki. 2023. Teachable Reality: Prototyping Tangible Augmented Reality with Everyday Objects by Leveraging Interactive Machine Teaching. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*. 1–15.

[39] Michael Nebeling and Maximilian Speicher. 2018. The Trouble with Augmented Reality/Virtual Reality Authoring Tools. In *2018 IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct)*. IEEE, Munich, Germany, 333–337. https://doi.org/10.1109/ISMAR-Adjunct.2018.00098

[40] Inc. Niantic. 2024. Pokémon GO. https://pokemongolive.com/?hl=en. Accessed: 2024-07-22.

[41] A.-M. Oncescu, A.S. Koepke, J. Henriques, and Albanie-S. Akata, Z. 2021. Audio Retrieval with Natural Language Queries. In *INTERSPEECH*.

[42] OpenAI. 2024. GPT-4 Technical Report. arXiv:2303.08774 [cs.CL]

[43] Xun Qian, Fengming He, Xiyun Hu, Tianyi Wang, Ananya Ipsita, and Karthik Ramani. 2022. Scalar: Authoring semantically adaptive augmented reality experiences in virtual reality. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*. 1–18.

[44] Iulian Radu and Bertrand Schneider. 2019. What can we learn from augmented reality (AR)? Benefits and drawbacks of AR for inquiry-based learning of physics. In *Proceedings of the 2019 CHI conference on human factors in computing systems*. 1–12.

[45] Nikunj Raghuvanshi and Ming C. Lin. 2006. Interactive sound synthesis for large scale environments. In *Proceedings of the 2006 symposium on Interactive 3D graphics and games (I3D '06)*. Association for Computing Machinery, New York, NY, USA, 101–108. https://doi.org/10.1145/1111411.1111429

[46] Ismo Rakkolainen, Ahmed Farooq, Jari Kangas, Jaakko Hakulinen, Jussi Rantala, Markku Turunen, and Roope Raisamo. 2021. Technologies for multimodal interaction in extended reality—a scoping review. *Multimodal Technologies and Interaction* 5, 12 (2021), 81.

[47] Zhimin Ren, Hengchin Yeh, and Ming C. Lin. 2013. Example-guided physically based modal sound synthesis. *ACM Transactions on Graphics* 32, 1 (Feb. 2013), 1:1–1:16. https://doi.org/10.1145/2421636.2421637

[48] Flavio Ribeiro, Dinei Florencio, Philip A Chou, and Zhengyou Zhang. 2012. Auditory augmented reality: Object sonification for the visually impaired. In *2012 IEEE 14th international workshop on multimedia signal processing (MMSP)*. IEEE, 319–324.

[49] Flávio Ribeiro, Dinei Florêncio, Philip A. Chou, and Zhengyou Zhang. 2012. Auditory augmented reality: Object sonification for the visually impaired. In *2012 IEEE 14th International Workshop on Multimedia Signal Processing (MMSP)*. 319–324. https://doi.org/10.1109/MMSP.2012.6343462

[50] Agnieszka Roginska and Paul Geluso. 2017. *Immersive Sound*. Focal Press.

[51] Hessam Roodaki, Navid Navab, Abouzar Eslami, Christopher Stapleton, and Nassir Navab. 2017. SonifEye: Sonification of Visual Information Using Physical Modeling Sound Synthesis. *IEEE Transactions on Visualization and Computer Graphics* 23, 11 (Nov. 2017), 2366–2371. https://doi.org/10.1109/TVCG.2017.2734327 Conference Name: IEEE Transactions on Visualization and Computer Graphics.

[52] Dariusz Rumiński. 2015. An experimental study of spatial sound usefulness in searching and navigating through AR environments. *Virtual Reality* 19, 3-4 (2015), 223–233.

[53] Stefania Serafin, Michele Geronazzo, Cumhur Erkut, Niels C. Nilsson, and Rolf Nordahl. 2018. Sonic Interactions in Virtual Reality: State of the Art, Current Challenges, and Future Directions. *IEEE Computer Graphics and Applications* 38, 2 (March 2018), 31–43. https://doi.org/10.1109/MCG.2018.193142628 Conference Name: IEEE Computer Graphics and Applications.

[54] Roy Sheffer and Yossi Adi. 2023. I Hear Your True Colors: Image Guided Audio Generation. In *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 1–5. https://doi.org/10.1109/ICASSP49357.2023.10096023

[55] Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu, and Yueting Zhuang. 2023. Hugginggpt: Solving ai tasks with chatgpt and its friends in huggingface. *arXiv preprint arXiv:2303.17580* (2023).

[56] Mustafa Sırakaya and Didem Alsancak Sırakaya. 2022. Augmented reality in STEM education: A systematic review. *Interactive Learning Environments* 30, 8 (2022), 1556–1569.

[57] Ryo Suzuki, Rubaiat Habib Kazi, Li-yi Wei, Stephen DiVerdi, Wilmot Li, and Daniel Leithinger. 2020. RealitySketch: Embedding Responsive Graphics and Visualizations in AR through Dynamic Sketching. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology* (Virtual Event, USA) *(UIST '20)*. Association for Computing Machinery, New York, NY, USA, 166–181. https://doi.org/10.1145/3379337.3415892

[58] Tomu Tahara, Takashi Seno, Gaku Narita, and Tomoya Ishikawa. 2020. Retargetable AR: Context-aware Augmented Reality in Indoor Scenes based on 3D Scene Graph. In *2020 IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct)*. 249–255. https://doi.org/10.1109/ISMAR-Adjunct51615.2020.00072

[59] Unity Technologies. 2023. *Unity*. https://unity.com/ Accessed on Oct 9th, 2023.

[60] Unity Technologies. 2023. Getting Started with Unity MARS. https://unity.com/products/mars/get-started. Accessed: 2024-04-02.

[61] Paul Upchurch and Ransen Niu. 2022. A Dense Material Segmentation Dataset for Indoor and Outdoor Scene Parsing. http://arxiv.org/abs/2207.10614 arXiv:2207.10614 [cs].

[62] Chenfei Wu, Shengming Yin, Weizhen Qi, Xiaodong Wang, Zecheng Tang, and Nan Duan. 2023. Visual chatgpt: Talking, drawing and editing with visual foundation models. *arXiv preprint arXiv:2303.04671* (2023).

[63] Yusong Wu, Ke Chen, Tianyu Zhang, Yuchen Hui, Taylor Berg-Kirkpatrick, and Shlomo Dubnov. 2023. Large-scale contrastive language-audio pretraining with feature fusion and keyword-to-caption augmentation. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 1–5.

[64] Dongchao Yang, Jianwei Yu, Helin Wang, Wen Wang, Chao Weng, Yuexian Zou, and Dong Yu. 2023. Diffsound: Discrete diffusion model for text-to-sound generation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing* (2023).

[65] Yipin Zhou, Zhaowen Wang, Chen Fang, Trung Bui, and Tamara L. Berg. 2018. Visual to Sound: Generating Natural Sound for Videos in the Wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[66] ZhiYing Zhou, Adrian David Cheok, Yan Qiu, and Xubo Yang. 2007. The Role of 3-D Sound in Human Reaction and Performance in Augmented Reality Environments. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans* 37, 2 (March 2007), 262–272. https://doi.org/10.1109/TSMCA.2006.886376 Conference Name: IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans.

A TASK PROMPTS

Here we provide text prompts for tuning a GPT4.0 model into our sound acquisition process controller.

System Context:
You are an assistant helping user generate or retrieve matching sound assets for augmented reality events. You will be provided with the description of the event, including: who caused this event, what is this event, and what is the subject of this event, which can be null for some cases. Note that the description of the causer and subject of event can sometimes be long and you need to extract the important part of it to create concise replies. For example, when being told that the causing entity of the event is “metalball of a plastic slope with balls running on it” you should know that the only mattering keyword here is the metalball.
You have three methods to provide sound assets. Method 1 is recommending from a predefined list of sound assets, each with a name describing the content. You will recommend the best matching name based on event description. The full list of sound asset will be provided at the end of this context; Method 2 is an online sound retrieval API, for which you should generate a simplified text prompt to search with, preferably within 4 words; Method 3 is a diffusion model that takes text prompt and generate sound assets. This method has three functions: generating new sound file with text prompt, generating similar sound file based on an existing sound, and transferring an existing sound with a text prompt.
For each conversation, you will first provide a response about all three sound sources. User may ask following up questions in reply, then you just reply based on specific feedback. When providing results, you should always follow a strict format starting and end with #. For method 1, you should reply the best matching sound asset in #method1:FILENAME# format. For method 2, you should reply in format of #method2:PROMPT#, please replace the PROMPT with your generated prompt. For method 3, when generating new sound, you should reply with #method3generation:PROMPT#, please replace the PROMPT with your generated prompt; when generating similar sound of a existing sound, reply with #method3similar#, when transferring an existing sound, reply with #method3transfer:PROMPT#, please replace the PROMPT with your generated prompt. Note that the hashtags need to be kept at both start and end of the result, the FILENAME need to be replaced by the exact name provided in the asset list, with no exception. The PROMPT should be replaced with the prompt you generate for these cases. Don’t explain why you have these prompts, just stricktly follow the formats.
Here I list names of sound assets to recommend from, each filename represents its content:
1. Knock surface
2. Sliding sound
3. Crash Bulb Break
<... 32 assets in total...>

We also prompt GPT4.0 to better parse the context information of specific AR events. We show one example event in the following prompt:

Conversation Prompt:
Please give me the results as hinted by context. For this specific event, the description of the AR event is:
The sound-producing AR event is *TapSurface*. The causing entity of this event is *a metal robot*. The target entity is *a wooden surface*. You should reply top5 results with method 1, ensuring that the filename provided is exactly from the asset list. You should also give one result with method 3 transfer, with a reasonable prompt that could transfer an existing tapping sound to a more material-aware sound. For example, for a tapping on a wooden surface, you could use tapping wood as the prompt. Reply with the format of #method3transfer:PROMPT#, replacing the PROMPT with your generated prompt. You can add a ’high quality’ tag in the prompt to help improve the sound quality.