“Did You Hear That?”
Learning to Play Video Games from Audio Cues

Raluca D. Gaina
Game AI Research Group
Queen Mary University of London, UK
r.d.gaina@qmul.ac.uk

Matthew Stephenson
Department of Data Science and Knowledge Engineering
Maastricht University, NL
matthew.stephenson@maastrichtuniversity.nl

Abstract—Game-playing AI research has focused for a long time on learning to play video games from visual input or symbolic information. However, humans benefit from a wider array of sensors which we utilise in order to navigate the world around us. In particular, sounds and music are key to how many of us perceive the world and influence the decisions we make. In this paper, we present initial experiments on game-playing agents learning to play video games solely from audio cues. We expand the Video Game Description Language to allow for audio specification, and the General Video Game AI framework to provide new audio games and an API for learning agents to make use of audio observations. We analyse the games and the audio game design process, include initial results with simple Q-Learning agents, and encourage further research in this area.

I. INTRODUCTION

Sound and music have long been an important aspect of video game development and play [1]. Not only can audio greatly influence our engagement and emotional investment in a game [2], but it can also provide important environmental information or gameplay cues [3]. Sounds within games can be used to alert the player to a nearby hazard (especially when in darkness), inform them they collected an item, or provide clues for solving certain puzzles. This additional sensory output is different from traditionally visual information, and allows for many new gameplay possibilities.

Certain games rely heavily on audio to create an immersive atmosphere, particularly horror games [4], and would likely lose much of their impact without it. Other games can sometimes require the player to listen to and understand certain sounds to progress, such as the numerous music and audio based puzzles in point-and-click adventure games (Myst, The Secret of Monkey Island, Machinarium, etc.). Even when not essential for the player to proceed, many games use sound to inform the player about useful non-visual information, such as alarm systems in many stealth games (Thief, Far Cry, Alien: Isolation, etc.) or enemy positions in first person shooters (Overwatch, Call of Duty, Battlefield, etc.) Without the ability to process audio input, we would likely be unable to play many of these games effectively. Some expert human players are also able to play games exclusively based on audio input even where on the surface visuals would appear essential, such as visually impaired players competing in fighting game tournaments [5] or speedrunners attempting to finish games such as Mario, Zelda or Punch Out while blindfolded.

While such examples are specific to the domain of video games, detecting and understanding certain sounds can be vitally important in a variety of other real-world scenarios. In such cases, current machine learning approaches that do not consider audio input as a factor in their decisions, could be seriously hindered. One topical example could be for a self-driving car [6]. Ambulances, fire engines and other emergency response vehicles typically use loud sirens and bright lights to alert traffic to pull over. It is often the case when driving that you can hear the sound produced by these vehicles long before you can see them. Most human drivers would be able to detect an approaching emergency response vehicle by sound and react appropriately. A self-driving car relying only on visual input would only react once it saw the vehicle itself. It is very possible to imagine many similar scenarios where the ability to listen to and interpret audio would be an important skill to possess. Returning to video games, any situation where there is an audible sound for something that is not yet visible on screen could be of benefit to an agent.

The remainder of this paper is structured as follows. Section II gives a brief background on the analysis of audio in games and the framework used in this work. Section III describes the expansion of the General Video Game AI framework to include audio cues. Section IV discusses initial experiments, while Section V concludes with future work.

II. BACKGROUND

A. Audio analysis in games

AI has previously been used in several ways to analyse the effects and meaning of game audio. TagATune is a game that involves annotating sounds and music [7]. The collection of categorised audio clips that an application like this can produce might allow AI to process and learn the likely effects of different output sounds (e.g. predict whether a particular sound indicates that the player is being healed or damaged). This idea of understanding the intended meaning of different sounds is also related to the field of AI-based audio stenography [8], which focuses on interpreting noisy or corrupt audio messages. Several AI approaches for providing a categorical understanding of spoken dialogue systems have also been proposed [9], including the use of deep reinforcement learning [10]. This has
several applications within video games, such as understanding human speech and generating suitable response options for NPCs. The topics of procedural audio generation [11] and sound synthesis are also highly relevant, as sounds produced by certain actions often depend on their consequences. For example, collecting a coin would likely yield an entirely different sound to that of killing an enemy. An interesting connection can be done to the work by Lopes et al. [12] and, more generally, the area of soundscape generation for games: where the Sonacia systems decides what sounds should be played in a procedurally generated level, our AI could learn to interpret them and react accordingly.

B. General Video Game AI Framework

We chose to develop an audio game-playing API within an existing framework in order to make use of existing agents (and visual-based games) as a starting point for the project. The General Video Game AI framework (GVGAI) [13] is a Java framework which contains a large (and continuously expanding) set of games written in the Video Game Description Language (VGDL), which propose varied challenges for game-playing agents, from navigation to puzzles to fast-reactionary problems. In addition to the large collection of games, we consider an important benefit the fact that new games can easily be created in VGD, as well as varying existing ones to obtain a potentially infinite supply of games. The Learning Track in the GVGAI competition proposes the challenge of developing general learning agents based on either visuals (an image of the game state can be provided) or symbolic information. In Section III we describe the expansion of the framework by adding the possibility of including audio signals in VGD, as well as processing these appropriately in the Java framework and sending correct observations to the agents.

III. AUDIO GAMES IN GVGAI

We integrated audio with VGDL in order to provide sound properties for games in two of the definition sets:

• **SpriteSet**: Each of the sprites can have audio files associated with them in the format `audio=move:filename1;use:filename2`. Audio signals are integrated into 3 functionalities:
  - Sprite movement: audio plays on each sprite move.
  - USE-type1 sprite action: audio plays whenever the sprite applies a USE-type action and it can be applied to spawn points as well as avatars, e.g. "play chime sound each time a new enemy is spawned".
  - Beacon sprite: audio plays at every game tick (volume based on the proximity to the player's avatar).

Several or none of the options can be defined per sprite.

• **InteractionSet**: Each of the interactions defined in GVGAI can have audio files associated with them in the format `audio=filename`. The SoundManager class is called to play the defined sound (if any) at the beginning of each interaction, i.e. when 2 sprites that have a defined effect overlap. We have also added a new interaction option, `playSound`, which only plays the given sound, without any other effects taking place.

The SoundManager features several easy-to-use methods for sound management, including playing, pausing or restarting sounds. It uses a pre-defined path and file extension (thus only the name of the file within the path, without the extension, should be included in VGDL). In the current version, only `.wav` files are supported.

In addition to the VGDL integration, a new AI agent API and game-running options have been introduced as well. Audio-only game-players should extend the AudioPlayer class and implement the required methods. They receive `AudioStateObservation` objects, which restrict observations to sound only: the agents are only provided with an array of `AudioObservation` objects, ordered by the proximity to the avatar (closest observations will be first). Each `AudioObservation` object includes details of its proximity to the avatar, as well as information about the `.wav` file associated with the observation for quick processing: bytes, fingerprint, normalized amplitudes and spectogram. The `.wav` file itself is also supplied, to allow for use of custom libraries in audio processing.

The codebase is publicly available on GitHub2.

A. Audio Games Set

This section describes the 3 games we have included with the initial version of the framework.

**ALIENS**. The player controls a spaceship which can move left and right, as well as shoot the incoming aliens, with the goal of killing all aliens. Bases can protect the player from incoming alien bombs, but can also be destroyed by player missiles. This is an audio adaptation of the original game in the GVGAI framework. Audio signals play each time the avatar shoots, aliens drop bombs, bombs or missiles kill bases or aliens, and when the avatar hits the edge of the play area.

**LABYRINTH**. A simple maze navigation game, where each level comprises of paths in-between impassable walls. This game is simplified from the GVGAI version with the removal of traps. Audio signals play each time the avatar bumps into walls, as well as every tick for the exit, a beacon sprite.

**BLOODSHED**. A fighting game in which the avatar can move left or right and wave its sword in the direction it is currently facing. This game is adapted from the 2015 Samtupy Productions audio game “Bloodshed, release the pain”3. Audio signals play each time the avatar bumps into the edge of the play area, when they wave their sword, when they hit an enemy fighter or when the player is hit, with different sounds depending on the direction of the attack (left or right).

IV. EXPERIMENTS AND DISCUSSION

This section describes some simple proof-of-concept experiments and analysis of the framework and games included. To

---

1. In GVGAI, **USE** actions are used for avatars as the fifth legal action in some games, which can have various effects, from spawning new sprites of different types to jumping or activating some avatar-specific properties.

2. [https://github.com/rdgain/GVGAI-Sound](https://github.com/rdgain/GVGAI-Sound)

3. [https://www.audiogames.net/](https://www.audiogames.net/)
this extent, we have adapted the sample Q-Learning method from the GVGAI Learning track [13] to work with the new API. In particular, we now describe a state \( S_t \) in terms of audio observations received by the agent, \( O_{t0}, O_{t1}, \ldots O_{tn} \) corresponding to any audio signals triggered at game tick \( t \). The reward function \( Q(S_t, a_t) \) is then updated depending on the reward received according to Equation 1.

\[
Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r_{t+1} + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t)] \tag{1}
\]

As the game score is not available as part of the observation space, we replace the reward perceived by the agent with a feedback for the player's attempted actions in order for the game to remain playable. Adding too little input would make the game impossible to play, as there would not be sufficient feedback for the player to figure out how the game works, or for them to be able to map out the game world.

A. Designing Audio Games

In this context, we can further highlight a parallel made in previous literature [14], between designing games and orchestrating large musical pieces to which several different instruments contribute. Here we can see each sprite or interaction as a possible instrument which need to play the right notes at the right time. Overlaying too much input would overload the human processing capacity and such an audio game would become unplayable. Adding too little input would also make the game impossible to play, as there would not be sufficient feedback for the player’s attempted actions in order for the player to figure out how the game works, or for them to be able to map out the game world.
would learn to consistently find the exit. Figure 2 shows this agent’s knowledge base size progression in the game Labyrinth (training on individual levels in green, and on all levels in red over 100 runs each). We can observe that in some levels the agent is able to move around more and learn more about the environment, while it appears beneficial to use more than 1 level for developing the knowledge. Using better learning algorithms and better exploration policies would help this agent in learning more efficiently to increase its performance.

V. CONCLUSIONS

In this paper we describe work in progress regarding an interesting direction of game-playing AI research: learning to play video games from audio cues only. We highlight that current state-of-the-art techniques rely either on visuals or symbolic information to interpret their environment, whereas humans benefit from the processing of many other types of sensor inputs. Sounds and music are key elements in games, which not only affect player experience, but gameplay itself in certain scenarios. We also introduce an extension of the General Video Game AI framework to support audio in games and audio observations. Simple Q-Learning agents were suggested to have promise in such environments.

There are many research directions opened by this work. A question that might arise is why not simply include observations of events or sprites behaviour? In partial observability scenarios (i.e. Starcraft), important events may take place outside of the player’s vision range which are often signalled through sound effects. These sounds can then be analysed in more detail in order to create an appropriate response: while often intuitive for humans, machines can make use of sentiment analysis research [15], for example, to identify what specific sounds might mean. Another interesting line of future work regarding audio analysis would be looking into exactly how removing certain sounds affects the agent’s performance, in the context of having an agent proficient enough to be able to solve the games given enough information. The knowledge and abilities learned by such high-skilled audio game-playing agents can be used together with other methods to maximise sensor usage for superior performance.

Audio design in games also raises some important challenges when it comes to inclusivity and accessibility [16]. People who may be partially or completely blind rely exclusively on audio, as well as some minor haptic feedback, to play a large number of video games effectively [17]. Including audio as well as visual information within a game can make completing it much more plausible for visually impaired players. Additionally, individuals with hearing difficulties would find it hard to play games that are heavily reliant on sound [18].

Intelligent agents can help to evaluate games for individuals with disabilities: if an agent is able to successfully play a game using only audio or visual input, then this could help validate the game for the corresponding player demographics.

REFERENCES