Agents that Listen: High-Throughput Reinforcement Learning with Multiple Sensory Systems

Shashank Hegde
University of Southern California
Los Angeles, United States
khegde@usc.edu

Anssi Kanervisto
University of Eastern Finland
Joensuu, Finland
anssk@uef.fi

Aleksei Petrenko
University of Southern California
Los Angeles, United States
petrenko@usc.edu

***Abstract*—Humans and other intelligent animals evolved highly sophisticated perception systems that combine multiple sensory modalities. On the other hand, state-of-the-art artificial agents rely mostly on visual inputs or structured low-dimensional observations provided by instrumented environments. Learning to act based on combined visual and auditory inputs is still a new topic of research that has not been explored beyond simple scenarios. To facilitate progress in this area we introduce a new version of VizDoom simulator to create a highly efficient learning environment that provides raw audio observations. We study the performance of different model architectures in a series of tasks that require the agent to recognize sounds and execute instructions given in natural language. Finally, we train our agent to play the full game of Doom and find that it can consistently defeat a traditional vision-based adversary.**

We are currently in the process of merging the augmented simulator with the main ViZDoom code repository. Video demonstrations and experiment code can be found at `https://sites.google.com/view/sound-rl`.

***Index Terms*—reinforcement learning, machine learning, video games, artificial intelligence, sound**

I. Introduction

Reinforcement learning (RL) algorithms have reached tremendous success in the field of embodied intelligence, including human-level control in Atari games [1], [2] and in first-person games [3], [4], and super-human control in competitive games [5], [6]. These state-of-the-art learning methods allow artificial agents to discover efficient policies that map high-dimensional unstructured observations to actions. While the general framework of deep RL enables learning from arbitrary sources of data, so far the majority of research in embodied AI focused on learning only from visual input (for example, see all the previous citations). We argue that another important sensor modality, sound, is largely overlooked.

Sound represents a highly salient signal rich with information about the environment. Sound cues correspond to discrete events such as contacts and collisions which might be difficult to identify from visual data alone. Stereo sound encodes important spatial information that can reveal objects and events outside of the agent's field of view. Finally, sound could be used to establish a natural communication channel between agents in the form of speech and hearing, which is one of the distinguishing features of higher forms of intelligence.

978-1-6654-3886-5/21/$31.00 ©2021 IEEE

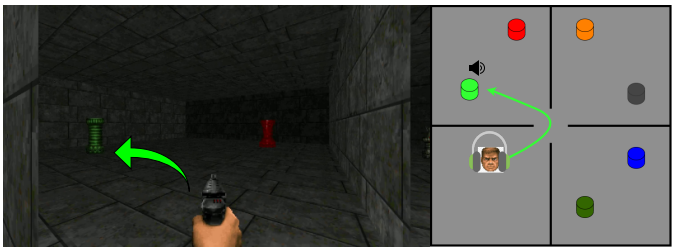


Fig. 1. Trained agent follows visual and sound cues to reach the target object in a ViZDoom environment.

In computer games, especially in the first-person shooter (FPS) genre, the ability to perceive and understand game sounds is one of the essential skills. This is particularly important in tactical duel scenarios in games like Quake or Doom: in order to gain an advantage skilled players listen to their opponent's actions to understand where they are on the game level and what resources they possess.

Reinforcement learning on combined auditory and visual inputs is complicated by the lack of infrastructure. The existing learning environments either do not support sound, or do not allow high-throughput parallel simulation necessary for large-scale experiments. We attempt to improve the situation by releasing an augmented version of the popular ViZDoom environment [7] where in-game stereo sound is available to the agents. Our implementation is decoupled from dedicated sound hardware typically used for audio rendering, and thus allows faster-than-realtime parallel simulation. We proceed to train agents in our environment in a series of increasingly complex scenarios designed to test various aspects of sound perception.

II. Related work

A number of prior projects explored RL with audio observations. Gaina and Stephenson [8] augmented General Video Game AI framework to support sound, focusing on 2D sprite-based games. Chen et al. introduced SoundSpaces [9], a version of Habitat environment which focuses on audio-visual navigation in photorealistic scenes. SoundSpaces was further used in [10] to investigate the problem of separating sound sources from background noise. Park et al. [11] introduced

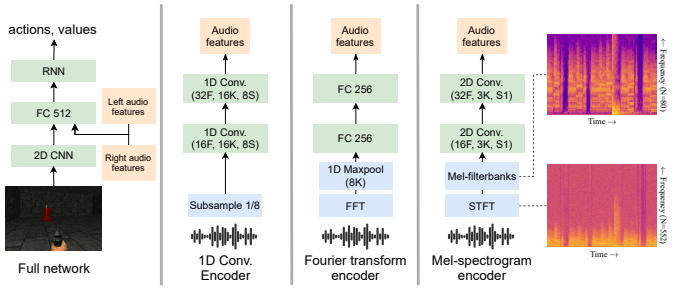


Fig. 2. Illustration of the network architecture and audio encoders used (see Appendix of [4] for complete details). Here K stands for kernel size, F for number of filters, S for stride, FC for fully connected layers and STFT for short-term Fourier transform. All convolutional layers are followed by max-pool layers with kernel size two.

a general-purpose simulation platform based on Unity engine with both auditory and visual observations.

While ViZDoom [7] supports in-game stereo sounds, the default audio subsystem is not designed for faster-than-realtime experience collection, and thus can only be used in relatively basic scenarios [12]. To our best knowledge, the version of ViZDoom presented in this work is the first simulation platform that enables accelerated embodied simulation with sounds at tens of thousands of actions per second, enabling large-scale training. Our experiments with the Doom duel scenario represent one of the first deployments of an agent with auditory and visual perception in a full first-person computer game.

III. ViZDoom environment with audio

We generate the audio observations for the agents through the OpenAL[1] sound subsystem supported by ViZDoom. OpenAL implementation offers many modern features, such as 3D sounds, reverberation, Doppler shift, and dampening of the sounds based on the agent's gaze direction with respect to the source.

Normally the sound engine is designed for human perception and plays back the sound samples in real time, prohibiting fast simulation. We circumvent this issue by using OpenAL Soft[2] with the *ALC_SOFT_loopback* extension which completely decouples the in-game sound from the device audio and enables software rendering of sounds on the CPU. This allows us to generate both visual and auditory observations at a maximum rate, enabling the environment simulation in the lock-step fashion typical for a RL setup.

In addition to that, an *ALC_EXT_thread_local_context* extension allows us to spawn a large number of game instances generating sound samples simultaneously. We leverage that in our experiments by starting hundreds of concurrent processes to achieve high training throughput with an asynchronous RL framework [4].

By directly accessing OpenAL sound buffers we expose raw audio observations through ViZDoom API. To give the agents access to all available sound data, we implement configurable audio frame-stacking, independent of the ViZDoom frame-skipping parameters. By default, if the agent chooses its action in the environment once every N simulation steps, we provide the audio observation containing the sounds for the previous N steps. The length of this window can be increased if needed, for example to facilitate training of feed-forward policies.

Another configuration parameter we expose is the audio sampling rate. A larger sampling frequency is analogous to a higher screen resolution, it enables more detailed observations at a cost of increased computation time. In this work we used a fixed sampling rate of 22050 Hz, which provides fast rendering and high sound quality.

[1] https://openal.org/

[2] https://github.com/kcat/openal-soft

IV. Audio encoder architectures

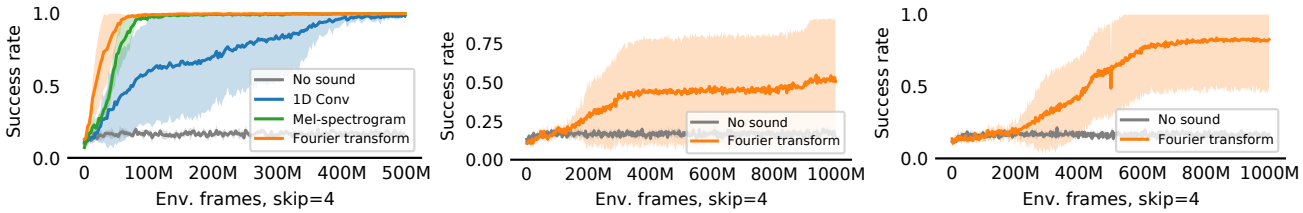
Our focus is on finding a general approach for processing sound with neural network-based policies. We seek models that are powerful and general enough to solve different, complex tasks, yet compact enough to facilitate fast learning. Using deep learning to process raw image pixels has been successful in RL [1], [3], however processing raw audio samples usually takes very large models to do efficiently [13], and to this day many state-of-the-art audio systems rely on some form of feature engineering (see Garcia et al. [14] for an example). These features are applicable to different tasks, with varying levels of performance depending on the task at hand.

For this reason, we propose three different encoders, which we compare in our experiments. The task of the audio encoder is to generate a compact representation of the raw sound data. This representation is then concatenated with the features from the image processing network. The resulting vector of features is fed to the rest of the network to generate actions and value estimates (see Fig. 2).

The raw audio input is a vector $s \in \mathbb{R}^n$, $s_i \in [-1, 1]$ containing n normalized audio samples. ViZDoom runs at a fixed 35 frames per (realtime) second, so for each simulation step this input contains audio corresponding to 29ms of gameplay. With the fixed 22050Hz sampling rate and standard 4-frameskip, our audio observation consists of 2520 samples, or 114ms of audio. We process both left and right audio channels separately and concatenate channel features into a single output vector. Fig. 2 illustrates the high-level structure of the encoders.

a) 1D Conv.: We downsize the audio by taking every 8$^{\text{th}}$ sample and then feed the samples through two 1D convolutional layers. While this removes high-frequency components (anything above $\approx$ 3000Hz), most of the information lies below this frequency threshold. This downsampling allows us to reduce computational complexity. The convolutional encoder can be considered a naive baseline approach

b) Fourier transform: We transform the audio buffer to frequency domain using fast Fourier transform (FFT) and take the natural logarithm of the magnitudes $s_{\text{FFT}} = \log \text{FFT}(s) \in \mathbb{R}^{n/2}$, downsample with a 1D max-pool layer and then feed it through a two-layer, fully connected network. This discards the



(a) Music Recognition (b) Sound Instruction (c) Sound Instruction Once

Fig. 3. Results on the main testing scenarios. Fig. 3a shows the comparison of different encoders in a sound source finding task. Figs. 3b and 3c show the performance of the FFT (Fourier transform) encoder on the *Instruction* and *Instruction Once* environments respectively. For each experiment we report mean and standard deviation of five independent training runs.

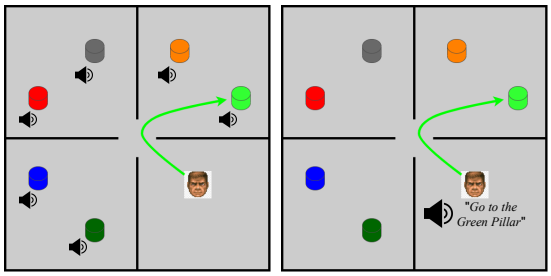


Fig. 4. Illustrations of the *Music Recognition* (left) and *Sound Instruction* scenarios (right). Locations of target objects and the player starting position are sampled randomly in each episode.

temporal information inside the 114ms of audio, but enables robust performance and a simple network architecture.

c) Mel-spectrogram: Motivated by the success of the mel-spectrogram approach in speech processing [14], we transform samples into frequency domain spectrogram with short-term Fourier transform (SFTF). STFT works by sliding a window over the audio samples, computing FFT on that window and then moving the window by a given hop. Depending on the window size and the hop length, the resulting spectrogram can have a large number of feature vectors (depending on the length of the audio) and frequency bins, with most high-frequency components containing only a minimal amount of useful information. Motivated by studies of human audio perception, mel-frequency scale emphasises higher resolution at lower frequencies, usually computed by using triangular overlapping windows [15]. This comes with the additional benefit of reducing the size of the spectrogram. We compute the spectrogram using parameters from [14], with a window size of 25ms, 10ms hop size and 80 mel-frequency components. See Fig. 2 for an example of the spectrograms produced with these hyperparameters. The resulting spectrogram is processed by two 2D convolutional layers.

V. Experimental Setup

We train our agents using an asynchronous RL framework Sample Factory. We follow ViZDoom experiments in the original paper [4] and use the same algorithm, hyperparameters, and model architectures. In particular, we use the asynchronous proximal policy optimization (PPO) algorithm with clip loss [16] and V-trace off-policy correction [17]. Our model consists of a three-layer, convolutional network to process the RGB image, a chosen audio encoder, a gated recurrent unit layer [18], and a fully-connected layer to produce action probabilities and value estimates. We ran all our experiments on a single 36-core server with four Nvidia RTX 2080Ti GPUs.

A. Environment scenarios

In order to test the audio encoders and the agent's overall problem-solving abilities we designed three different scenarios based on the map layout depicted in the Figs. 1 and 4, where six visually distinct pillars are placed in four different rooms. The ordering of pillars and the agent starting position is randomized in each episode. Our fourth and final scenario is a self-play duel in a full game of Doom.

a) Music Recognition: Each pillar plays a different music track in a loop throughout the episode. One pillar is randomly chosen to play an unique target track. The agent is given a $+1$ reward upon touching the pillar that plays the target track. Touching other pillars terminates the episode with a 0 reward. The attenuation value of the sound sources is high, therefore the agent has to move close to a pillar to hear the sound. Thus the agent's strategy shall be to move from pillar to pillar and listen, until it finds the pillar playing the target track.

b) Sound Instruction: During the episode the agent repeatedly hears a command in spoken English, which instructs it to go to a particular object. The agent is rewarded for touching the correct object and receives zero reward otherwise. Unlike the previous scenario where the decision to move close to an object was purely based on the sound, here the agent has to use both visual and auditory input to complete the task.

c) Sound Instruction Once: A more complex version of the *Sound Instruction* environment where the instruction is only given once at the beginning of the episode. This scenario tests a combination of multimodal perception and the ability to memorize instructions.

d) Duel: Finally, we train our agents in a 1v1 self-play matchup in the full game of Doom, following a setup similar to [4] except with full access to in-game sounds. We evaluate the agent against a separately trained agent that is not equipped with the sound encoder, and we hypothesise that the agent with access to sound can outperform the deaf agent.

TABLE I
RESULTS OF 1V1 MATCHES BETWEEN OUR AGENT THAT HAS ACCESS TO THE SOUND AND A VISION-ONLY AGENT. "SOUND (DIS.)" IS THE MAIN AGENT WITH SOUND INPUTS DISABLED DURING THE EVALUATION.

Match	Wins	Losses	Draws
Sound vs No sound	53	31	16
Sound vs Sound (dis.)	74	17	9

B. Training settings

We test all audio encoders in the *Music Recognition* scenario, where training converges within 5×10^8 environment steps. We then choose the best-performing encoder for other experiments, where we train for 10^9 steps in *Sound Instruction* scenarios and for 2×10^9 steps in *Duel* scenario. We fixed the image resolution to 128x72 and set the frameskip to 4 for all environments except Duel which was run with 2-frameskip.

VI. RESULTS

After the initial testing on the *Music Recognition* scenario we found that the Fourier transform encoder was the most efficient (Fig. 3). We continued to test the FFT encoder on *Sound Instruction* and *Sound Instruction Once* scenarios. In the majority of the training runs the agent was able to reach optimal performance in each of these scenarios. The high variance in the results suggests that 10^9 steps of training are still not sufficient for all seeds to converge. We also found that the agent showed slightly better final performance on the supposedly harder task *Sound Instruction Once*. Although it is likely this is just a statistical anomaly given the high variance and low number of independent runs (limited by the computation budget), we leave full explanation of this surprising result for future work.

In the *Music Recognition* scenario, we saw the agent achieve the expected behaviour, where it uses the stereo sound to navigate to the correct pillar. The agent explores the map listening to different music and moves closer to the source when the target music track is recognised. We also saw the agent move towards pillars backwards, showing that visual input is often superfluous in this task. In the *Sound Instruction Once* scenario, we noticed that the agent goes to the center of the map early to await the instructions, which helps minimize the average time to complete the task. While the agent is waiting it keeps turning around memorizing the locations of the objects. Once the instruction starts the agent would quickly turn and approach the target object. This behaviour shows the agent's ability to combine auditory and visual cues to quickly explore its surroundings and map the sound instruction to the appropriate action. Besides, we noticed the agent's ability to memorize the instructions for the entire episode, courtesy of the recurrent model architecture.

Table I shows the benefit of having access to sound information in the *Duel* scenario. Here we trained two sets of agents using population-based training and self-play, with a small population of 4 policies. The main population ("Sound") used the FFT encoder and had access to both auditory and visual observations. Another set of agents ("No sound") used only image observations. After training for 2×10^9 steps we chose the best agents from both populations and ran two series of one hundred 4-minute matches between them. In the first series of games we compared "Sound" and "No sound" versions of the agents. Our main agent won in more games, demonstrating the advantage of the enhanced sensorium. In the second series of games we tested our "Sound" agent against a version of itself with its auditory observations replaced with silence ("Sound (dis.)"). The agent with disabled hearing played significantly worse, demonstrating the strong reliance of our agent on sound cues.

When analysing the behavior of the main agent in the duel environment we noticed the reduced usage of loud ammunition. We believe this allows the agent to conceal its position from the opponent which facilitates surprise attacks. The agent also uses its spatial sound perception to discover the location of the enemy by listening to the opponent's gunfire.

VII. TRAINING THROUGHPUT

To measure the total computation cost added by rendering and processing sound in our experiments we tracked the average training throughput. In single-agent experiments we collected experience using 72 parallel workers, each worker sampling 8 environments sequentially for a total of $72 \times 8 = 576$ parallel environments per experiment. We ran 4 such experiments at a time on a 36-core machine with 4 GPUs to maximize the hardware utilization. We observed training throughput of 1.5×10^5 game frames per second per experiment with disabled sounds and 1.2×10^5 when sound is enabled.

We did not notice a significant difference in performance in Duel scenario. Here we trained a population of 4 policies at a combined framerate of 6.7×10^4 both with and without the sound. The training performance in multi-agent VizDoom envs is bottlenecked by slow network-based communication between game instances in the multi-agent setup, and thus addition of sound rendering workload does not have a significant effect.

VIII. CONCLUSIONS AND FUTURE WORK

In this work we introduced an immersive environment based on ViZDoom that provides access to both auditory and visual observations while maintaining high simulation throughput. We introduced new scenarios that test the agent's ability to hear and identify sounds, as well as combine sound with visual cues. Our results indicate that transforming the audio samples into frequency domain with FFT is sufficient for fast and effective RL training when combined with a recurrent neural architecture. This is evident from the results of our experiments with sound separation and instruction execution, as well as results on a full game where the agents with augmented sensorium prevail. We hope that access to the efficient environment that simulates auditory experience will enable

large-scale experiments and can facilitate further research in this area.

Being a preliminary work, there are still a myriad of open questions and limitations to address. We only used one RL algorithm in our experiments and only three different audio encoders, without excessive hyperparameter and/or architecture tuning. The scenarios used in our experiments could also be extended: we used a limited bank of sounds in the experiments, and to assess how well the agent learned to "understand sound" instead of overfitting to specific cues, we need a larger bank of sounds to pick from. This can be done by adding more natural sounds and by augmenting the existing ones with random noise and other transformations to prevent the neural network from memorizing the exact samples.

While it is evident from our experiments that the agents benefit from the addition of auditory observations, it is not clear how exactly the agents utilize the sound cues. For our agents trained in the Duel scenario the behavior of the hearing agents can be studied in-depth, i.e. to find out in what ways the agent utilizes the sound and how well it can localize its opponent in 3D. We leave this interesting research direction for future work.

Finally, while this work used a recurrent neural architecture for temporal modelling ("memory" for the agent), it is unclear whether agents can understand long audio sequences. One should assess this with, for example, longer-lasting audio cues or longer, more dynamic commands (e.g. not only "go to X", but also "do not go to X", etc.) Preferably, these experiments should be combined with other cognitive tasks like in DMLab30 [17] to evaluate the agent's ability to really understand sound information. This could be compared to a pipelined baseline approach, where audio is preprocessed through a speech recognition system to assess the agent's ability to learn language understanding with pure end-to-end RL.

Acknowledgments

We thank the reviewers of this paper for very insightful comments and ideas for the future work, which we included in the previous section.

References

[1] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, p. 529, 2015.

[2] A. P. Badia, B. Piot, S. Kapturowski, P. Sprechmann, A. Vitvitskyi, Z. D. Guo, and C. Blundell, "Agent57: Outperforming the atari human benchmark," in *ICML*, pp. 507–517, 2020.

[3] M. Wydmuch, M. Kempka, and W. Jaśkowski, "Vizdoom competitions: Playing doom from pixels," *IEEE Transactions on Games*, vol. 11, no. 3, pp. 248–259, 2018.

[4] A. Petrenko, Z. Huang, T. Kumar, G. Sukhatme, and V. Koltun, "Sample factory: Egocentric 3d control from pixels at 100000 fps with asynchronous reinforcement learning," in *ICML*, pp. 7652–7662, 2020.

[5] O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. H. Choi, R. Powell, T. Ewalds, P. Georgiev, *et al.*, "Grandmaster level in starcraft ii using multi-agent reinforcement learning," *Nature*, pp. 1–5, 2019.

[6] OpenAI et al., "Dota 2 with large scale deep reinforcement learning," *arXiv:1807.01281*, 2019.

[7] M. Kempka, M. Wydmuch, G. Runc, J. Toczek, and W. Jaśkowski, "Vizdoom: A doom-based ai research platform for visual reinforcement learning," in *CIG*, pp. 1–8, 2016.

[8] R. D. Gaina and M. Stephenson, ""Did you hear that?" learning to play video games from audio cues," in *COG*, pp. 1–4, 2019.

[9] C. Chen, U. Jain, C. Schissler, S. V. A. Gari, Z. Al-Halah, V. K. Ithapu, P. Robinson, and K. Grauman, "Soundspaces: Audio-visual navigation in 3D environments," in *ECCV*, pp. 17–36, 2020.

[10] S. Majumder, Z. Al-Halah, and K. Grauman, "Move2hear: Active audio-visual source separation," *arXiv preprint arXiv:2105.07142*, 2021.

[11] K. Park, H. Oh, and Y. Lee, "Veca: A toolkit for building virtual environments to train and test human-like agents," *arXiv preprint arXiv:2105.00762*, 2021.

[12] A. Woubie, A. Kanervisto, J. Karttunen, and V. Hautamaki, "Do autonomous agents benefit from hearing?," *arXiv preprint arXiv:1905.04192*, 2019.

[13] A. v. d. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "Wavenet: A generative model for raw audio," *arXiv:1609.03499*, 2016.

[14] D. Garcia-Romero, G. Sell, and A. Mccree, "Magneto: X-vector magnitude estimation network plus offset for improved speaker recognition," in *Proc. Odyssey 2020 The Speaker and Language Recognition Workshop*, pp. 1–8, 2020.

[15] S. Davis and P. Mermelstein, "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 28, no. 4, pp. 357–366, 1980.

[16] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv:1707.06347*, 2017.

[17] L. Espeholt, H. Soyer, R. Munos, K. Simonyan, V. Mnih, T. Ward, Y. Doron, V. Firoiu, T. Harley, I. Dunning, S. Legg, and K. Kavukcuoglu, "IMPALA: Scalable distributed deep-rl with importance weighted actor-learner architectures," in *ICML*, pp. 1407–1416, 2018.

[18] K. Cho, B. van Merrienboer, Ç. Gülçehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," in *EMNLP*, pp. 1724–1734, 2014.