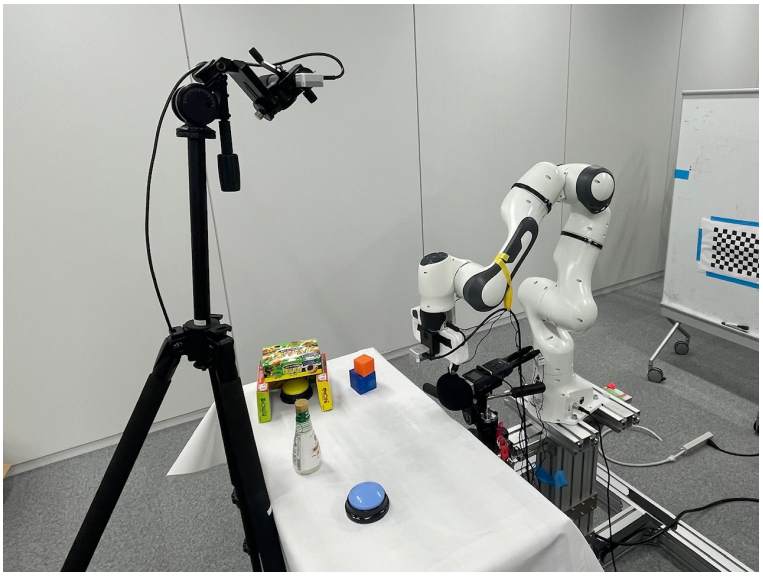
Engineering Assistant Internship

AIST AIRC Computer Vision Research Team



Gaspard Delpiano–Manfrini

NON-CONFIDENTIAL

Abstract

This internship report presents the work done on the implementation of sound in a Vision Language Action based control policy, an end-to-end framework for sound and vision based robotics manipulation, in the Computer Vision Research Team at the Artificial Intelligence Research Center (AIRC), part of Japan's National Institute of Advanced Industrial Science and Technology (AIST) in Tsukuba, Japan. A sound VLA model was developed based on the architecture and weights of SmolVLA. The base architecture was expended upon by adding a Mel Spectrogramme encoder and training a sound projector.
The work also includes a physics and sound simulation environment leveraged for custom dataset generation as well as in the reinforcement learning and inference testing pipeline.
Although finalising the training and testing of the model on Benchmarks has not been completed due to a lack of time, this work is still ongoing.

Keywords: Vision-Language-Action (VLA), robot learning, auditory manipulation, multimodal learning

Résumé

Ce rapport de stage présente les travaux menés sur l'intégration du son dans une politique de contrôle robotique basée sur le modèle « Vision Language Action » (VLA), un framework end-to-end pour la manipulation robotique basée sur la vision et le son, au sein de l'équipe de recherche en vision par ordinateur du Artificial Intelligence Research Center (AIRC), rattaché à l'Institut national; Advanced Industrial Science and Technology (AIST) à Tsukuba, au Japon. Un modèle VLA sonore a été développé à partir de l'architecture et des poids de SmolVLA. L'architecture de base a été étendue par l'ajout d'un encodeur utilisant le spectrogramme Mel et l'entraînement d'un projecteur sonore.
Ce travail comprend également un environnement de simulation physique et sonore utilisé pour la génération d'ensembles de données ainsi que dans le pipeline d'apprentissage par renforcement et de tests d'inférence.
Bien que la finalisation de l'entraînement et des tests du modèle sur des benchmarks n'ait pas été achevée faute de temps, ces travaux se poursuivent.

Mots-clés : Vision-Langage-Action (VLA), apprentissage robotique, manipulation auditive, apprentissage multimodal

Acknowledgements

I would like to thank the Computer Vision Lab for the warm welcome they gave me during my four-month internship. I would especially like to thank Dr. Qiu Yue, my internship advisor, for guiding me and being a great help throughout the internship. I would also like to thank Raphaël Druon, who helped me on numerous occasions and Thibault Lucas, with whom I had the pleasure of working closely throughout the entire internship. Finally, I would like to thank all my colleagues in the office —whether they are interns or researchers— with whom I have had the opportunity to engage in enriching discussions.

Contents

1 Introduction **1**
- 1.1 Laboratory presentation . . . 1
- 1.2 The Objectives . . . 1
- 1.3 Understanding machine learning . . . 2

2 Preliminary research **4**
- 2.1 Relevant papers and state of the art . . . 4
- 2.2 SmolVLA trial . . . 7
  - 2.2.1 Architecture and Inference Process . . . 7
  - 2.2.2 SO100 Setup description . . . 9
  - 2.2.3 Task description and training . . . 10
  - 2.2.4 Results . . . 10

3 Simulation **11**
- 3.1 Data needs . . . 11
- 3.2 RLBench . . . 12
- 3.3 Sound architecture . . . 12
  - 3.3.1 Collision detection . . . 13
  - 3.3.2 Sound library . . . 14
  - 3.3.3 Sound modulation . . . 14
  - 3.3.4 Manipulation tacks . . . 14
  - 3.3.5 Sound output for the model . . . 17
- 3.4 Reinforcement learning . . . 17

4 Physical Setup **17**
- 4.1 Franka Panda setup description . . . 17
- 4.2 Teleoperation . . . 18
- 4.3 Data-set recording . . . 19

5 Audio SmolVLA **19**
- 5.1 Implementation . . . 20
- 5.2 Training strategies . . . 20
- 5.3 Results . . . 21
  - 5.3.1 Pipeline Testing . . . 22
  - 5.3.2 Model shortcomings . . . 22
  - 5.3.3 Physical setup testing . . . 23
  - 5.3.4 Sound limitations . . . 23
  - 5.3.5 Software limitations and future . . . 23

6 Conclusion **24**

1 Introduction

I wanted to use the second year Internship as an opportunity to discover what research and working in a laboratory means with a hands on experience. Japan was an ideal place, as it has a long history of leading research in robotics and computer vision. Discovering a new culture far from what i am accustomed to was also a great experience, teaching me what working abroad can be like and the challenges it brings with it.
To counteract Japan's population ageing, the country has been leaning into the development of new technologies such as in robotics and computer vision. With the advent of LLM and the quest for more robust more general, smarter robot control algorithms, the VLA paradigm has become a prime field of research. At its core these algorithms use MLLM (Multimodal Large Language Model) to output directly robot controls from mainly textual and visual inputs. This framework enables the development of very general context dependant policies. As part of the Computer Vision Research Team at AIRC my goal was to take part in the development and training of a VLA model capable of understanding a new modality not natively included in VLA, sound.

The result of this internship was a joint effort between Thibault Lucas and my self, as such this report will focus mainly on my contribution while still giving a complete overview of the work conducted.

1.1 Laboratory presentation

The National Institute of Advanced Industrial Science and Technology (AIST) is one of Japan's largest public research organizations. Its research activities focus on developing advanced technologies and contributing to their practical implementation in industry and society. It regroups many different laboratory specialising in a diversity of fields such biotechnology, chemistry, electronics, geology and metrology.
Within AIST, the Artificial Intelligence Research Center (AIRC) is dedicated to research and development in artificial intelligence. The center aims to advance AI technologies while facilitating their integration into real-world applications and society. Its activities cover a broad range of research areas, including machine learning, pattern recognition, natural language processing and real-world sensing. The Computer Vision Research Team, the team where I had the chance to work, conducts research focused on acquiring information from real-world environments—such as 3D measurement and 3D modeling—and on using machine learning based on moving images and shape data to achieve objectives such as the recognition of people, objects and anomaly.
During my internship, I was under the supervision of Dr. Qiu Yue, a specialist in the fields of deep learning, 3D vision, natural language processing and has more recently greatly contributed on the research surrounding MLLM.

1.2 The Objectives

The objective of this internship was to develop and train a VLA model that would not only by able to understand the vision and language modalities but also include audio

information. As this field is recent, environment able to simulate sounds alongside a physical robotics simulation don't exist or are not made publicly available. Moreover, as the field is still quite recent, standard easy to use, wide spread tools don't really exist although some progress is being made with the developmental of platforms such as Hugging Face. Because of these constraints, the objective became as much about developing a new model as it also was about creating a robust infrastructure and pipeline for robotics VLA training.
My work ended up including:

- Creating a sound and physics simulation for robotic learning used in:
  - the creation of an imitation learning data-set based on simulated data
  - a reinforcement learning environment
  - an inference and benchmark environment
- Creating a real physical setup used in:
  - the collection of imitation learning data-set based on real data
  - reinforcement learning fine tuning on real physical setup
  - a real inference and policy testing environment
- Creating a new VLA model capable of understanding sound which includes:
  - a new model architecture
  - a training process
  - a coherent software infrastructure to run it

The hope was to prove the relevance of audio information in a manipulation context and understand in what situations sounds could be useful.

1.3 Understanding machine learning

The first tree weeks of the internship were focused on discovering and learning about the world of machine learning, deep learning, the transformer architecture, LLM and then VLA models. As the field was completely new to me, this was a great learning experience. The resources I used were mainly online classes on Open Classrooms [13] [12] and the Hugging Face website [5] [6], and recordings from machine learning classes at MIT [4]. Great explanation videos where also available on YouTube, notably a series on machine learning by 3Blue1Brown [1] and a series on the architecture and inner workings of LLMs by Welch Labs Channel [19]. After having acquired the necessary fundamental knowledge, I wanted to try a concrete training of a machine learning model, and I decide to train one capable of counting the number of fingers on a hand. For the task, I used the Pytorch Python module and a freely available hand data-set to train

the model on. The first step was to create the architecture of the model. After multiple iterations and considerations, I ended up with the following architecture:

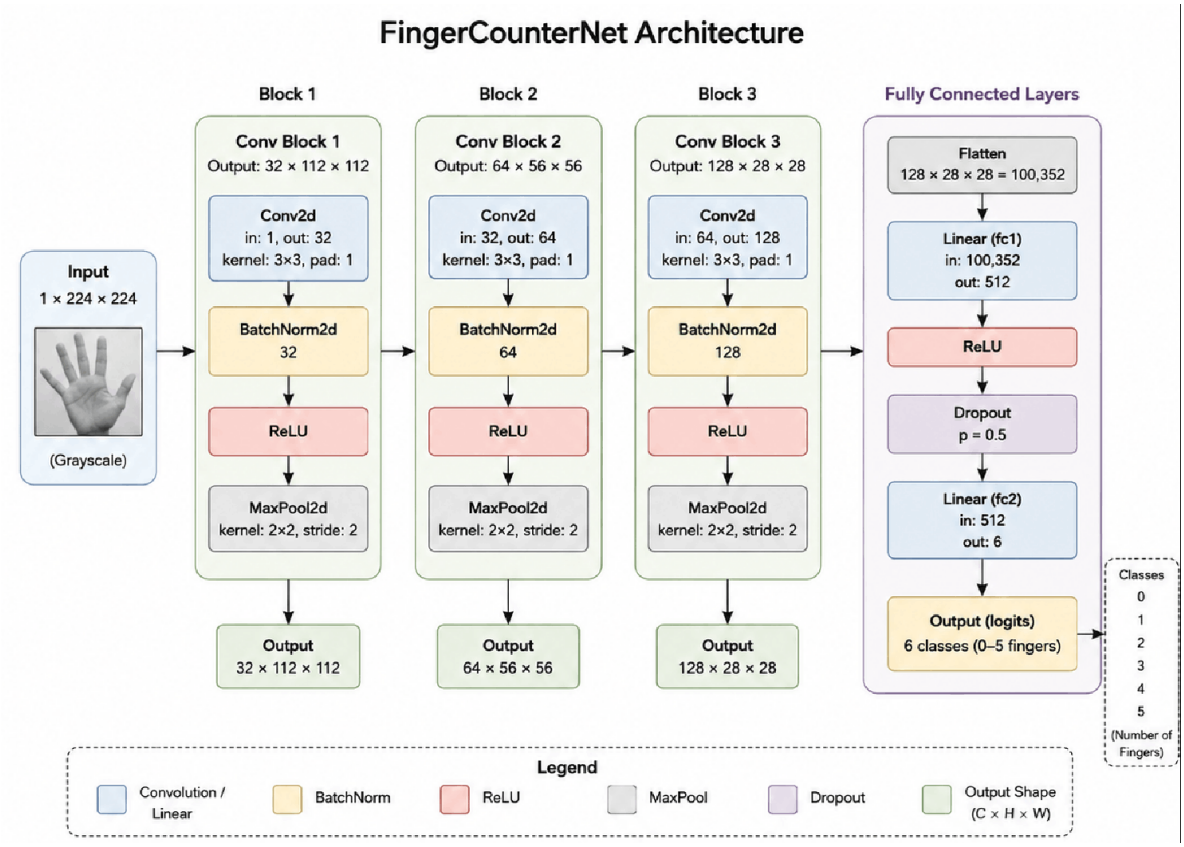


Figure 1: Finger counter model's architecture

The image is first preprocessed by applying the following transformations:

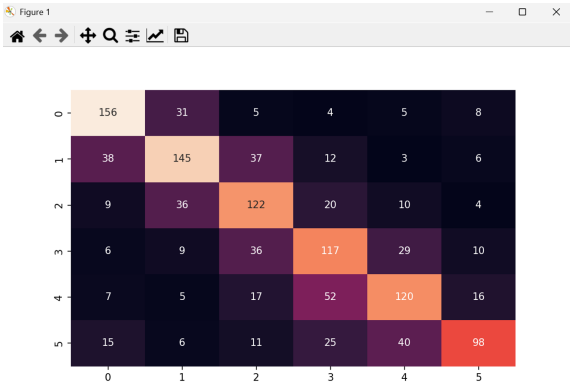
- Downscaling the image to a size of 224x224
- Transforming the image from RGB to a greyscale
- Stretching the image to a square image

This is done to reduce the size and standardise the input images, making the model smaller, faster to train, and faster to run.
The image then goes through three identical blocks. Their goal is to reduce the successive size of their input from 32x112x112 to 128x28x28 while retaining all the information contained inside the original image about the number of fingers. This is done by passing the input matrix into a 2D convolutional network, Batch Normalisation, ReLU activation, and 2×2 max pooling. The 2D convolutional network does the heavy compute work while the ReLU activation enables the model to learn non linear behaviours. The BatchNorm and MaxPool help reshape the output for faster training and more predictable behaviour of the data.

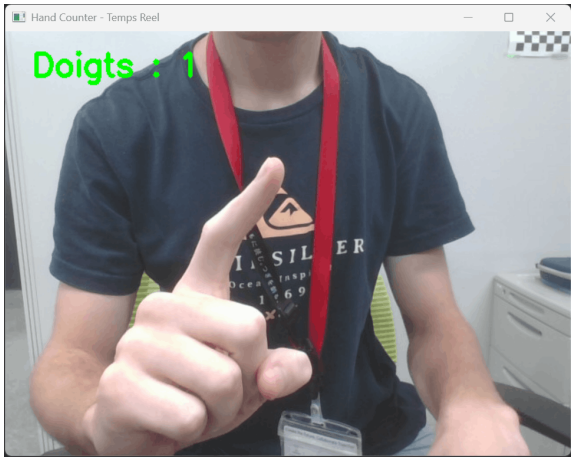
Finally, the output matrix if flattened into a single array of numbers that then passes through two successive interconnected layers of neurons to further reduce its size to 6 numbers between 0 and 1, each representing the probability the model assigns to the number of fingers in the input image.

The training was performed using freely available hand data-sets. The images where preprocessed to correspond to the input of the model. The data-set was widened by randomly rotating, cropping and shifting the original images. This gave the model more examples it could learn from, helping with robustness and generalisability.

After training we obtain the following Confidence matrix 3a :



(a) Confidence matrix: The X-axis shows the true number of fingers depicted on a picture and Y-axis represents the number of fingers predicted by the model



(b) Model inference testing

Figure 2: Results of the model

A perfect model would be a pure diagonal, meaning that each image is associated with the correct number of fingers. This shows that while with this architecture and training it is not perfect, it is still mostly able to retrieve the correct number of fingers from an image within ± 1 finger from the ground truth.

This approach aimed at training a small and compact model but today, most vision based model use RGB images as they contain much more information. In the case of finger counting, it could greatly help with differentiating the hand from the background and could help obtain a greater result at the expense of greater training time.

2 Preliminary research

2.1 Relevant papers and state of the art

After having acquired a satisfying understanding of machine learning, the next few weeks were spent researching relevant literature about VLA and MLLM that could understand sound. Although quite recent, major research in this field is being conducted,

lead by Audio VLA [18] and Hear [10].

The Audio-VLA paper [18] proposes a multimodal Vision-Language-Action model that integrates contact audio with visual and language information to improve robotic manipulation. The main motivation is that vision-only VLA models can struggle to perceive dynamic physical interactions, such as object contact and collisions, which are not always clearly observable visually. Audio-VLA addresses this limitation by using pretrained DINOv2 and SigLIP for visual features, AudioCLIP for audio features, and Llama 2 as the language backbone, with LoRA fine-tuning and a multimodal projection layer to align the different modalities.

The HEAR framework [10] proposed by IRMVLab extends the traditional Vision-Language-Action (VLA) paradigm to a Vision-Sound-Language-Action (VSLA) setting, where robots continuously perceive vision, sound, language, and proprioceptive information during manipulation. Similarly to Audio-VLA, HEAR addresses the limitations of vision-only policies by exploiting acoustic information such as contact sounds, alarms, speech, and process sounds. However, a key difference is that HEAR focuses much more explicitly on the temporal and causal nature of audio. The authors identify the "Blind Execution Interval" (BEI), where a short acoustic event may occur and disappear while the robot is executing an action chunk, preventing a conventional VLA policy from perceiving it. To address this issue, HEAR introduces a causal audio memory mechanism called Historizer, a multimodal reasoning module called Envisioner, a future-audio prediction module called Advancer, and an action generation module called Realizer.

As a base to create our VLA model we decided to choose the already existing LeRobot architecture with the SmolVLA model [14].

SmolVLA [14] is a compact Vision-Language-Action (VLA) model designed for robotic manipulation. It combines visual perception and natural language understanding with action prediction, allowing a robot to interpret its environment, understand a task described in natural language, and generate the corresponding actions. Compared with larger VLA models, SmolVLA focuses on reducing computational and memory requirements while maintaining useful performance for real-world robotic tasks. The model processes visual observations and language instructions and maps them to robot actions, enabling it to perform tasks such as grasping, moving, and manipulating objects. SmolVLA is particularly interesting for robotics research because its relatively lightweight architecture makes it more suitable for experimentation and deployment on limited hardware. It also provides a practical framework for studying how vision, language, and robot control can be integrated into a single model.

To create our own model we first thought about repurposing an already working VLA with audio from a research paper, unfortunately the resources provided by the laboratories where never enough to be able to fully recreate on our machines, the results they claimed to have achieved. Here is an overview of the papers we studied for the matter:

Model	Description	Available	Blocking point
Audio-VLA: Adding Contact Audio Perception to Vision-Language-Action Model for Robotic Manipulation [18]	Contact audio + vision + language + action for manipulation	Paper ✓ Code ✗ Weights ✗	No public implementation or checkpoints. Audio generated synthetically in RL-Bench/LIBERO.
HEAR: Towards the Vision-Sound-Language-Action Paradigm [10]	Continuous sound-centric manipulation using streaming audio world models	Paper ✓ Weights ✓ Code ✗	Available code incomplete, no mature training pipeline, benchmark still evolving. However, episode generation functional.
RoboOmni: Robot Manipulation in Omni-modal Context [17]	Proactive VLA + speech + contextual audio + proactive intent recognition	Paper ✓ Weights ✓ Code ✓	Audio mainly used for intent inference and contextual reasoning, not standalone manipulation feedback.
NaVLA²: A Vision-Language-Audio-Action Model for Multimodal Instruction Navigation [3]	Vision-language-audio-action navigation model	Paper ✓ Code ✗	Navigation only, not manipulation. No manipulation datasets or policies.
ELLSA: End-to-end Listen, Look, Speak and Act [16]	Speech-enabled embodied agent with perception, dialogue and actions	Inference ✓ Training ✗	Speech-centric, missing training data and reproducible training pipeline.
VLAS: Vision-Language-Action Model with Speech Instructions for Customized Robot Manipulation [20]	Speech instruction conditioned manipulation policy	Paper ✓ Training recipe ✓	No released checkpoints. Audio limited to speech commands.
SVA: Towards Speech-Enabled Vision-Language-Action Model [9]	Speech-command VLA	Paper ✓ Code ✗	Speech only. No environmental audio modelling.
Cosmos 3: Omnimodal World Models for Physical AI [11]	Audio + video + language + action world model → multimodal understanding and generation	Framework ✓ Models ✓	No released audio-to-action manipulation policy. Audio currently used mainly in world modelling rather than robotic control.

We chose for this internship to focus on an approach closer to the one proposed by Audio-VLA, leveraging the infrastructure of SmolVLA and adding on our own audio pipeline to the model.

2.2 SmolVLA trial

The first step of the creation of our new model was centred around understanding the inner workings of SmolVLA and training it on our own hardware and on our own data-sets.

2.2.1 Architecture and Inference Process

Multimodal observation and SmolVLM2

At each observation step, SmolVLA receives three main types of information: one or more RGB camera images, the current robot state, and a natural-language instruction describing the task. Multiple camera views can be used simultaneously, allowing the model to combine information from different viewpoints. The robot state provides proprioceptive information such as joint positions or other sensorimotor variables, depending on the robot and dataset.
The visual and textual inputs are processed by the corresponding components of SmolVLM2. Images are converted into visual representations, while the language instruction is tokenized into text tokens. The robot state is projected into the representation space expected by the model. These modality-specific representations are then combined so that the language-model backbone can process them jointly.
SmolVLM2 is based on the SmolLM2 language-model architecture. Its Transformer backbone contains repeated Transformer layers using attention mechanisms [15] and feed-forward sublayers, together with residual connections and normalization. The purpose of this stage in SmolVLA is not primarily to generate text. Instead, the Transformer produces contextualized multimodal features that describe the current task, visual scene and robot state in a representation that can be used by the action expert. The resulting feature representations act as conditioning information for the action expert. During inference, these representations do not need to be recomputed for every flow-matching integration step. They can be cached and reused while the action expert progressively refines the action sample.

Action expert

The second major component is the action expert. Rather than directly predicting the complete action sequence with a conventional regression head, SmolVLA uses flow matching to generate a continuous action chunk. At inference time, the process starts from a sample of Gaussian noise having the same temporal structure as the action chunk. The default action horizon is 50 future action steps. The number of action dimensions depends on the robot and its action representation, (50, 8) in the case of the SO100.

The noisy action representation is first projected into the hidden representation space of the action expert. The expert is a Transformer-based network that processes the action tokens while conditioning them on the contextual features produced by SmolVLM2.
Self-attention operates over the action representation and allows the model to capture dependencies between different elements of the predicted action sequence. This is particularly important because the 50 actions are not independent: consecutive actions should form a coherent temporal trajectory.
Cross-attention provides the connection between the action expert and the VLM. The action tokens act as queries, while the contextual features produced by SmolVLM2 provide the conditioning information. This allows the action expert to adapt its predicted motion to the visual scene, language instruction and robot state.

Flow matching

The output of the action expert at a given flow-matching time is a vector-field estimate. It should therefore not be interpreted directly as the final robot command.
Flow matching defines a continuous transformation between a simple initial distribution, here represented by noise, and the distribution of demonstrated robot actions. The action expert learns to estimate the direction in which the noisy action sample should move at a particular point along this transformation.
At inference time, the model starts from Gaussian noise corresponding to the initial point of the flow. The expert predicts a velocity field conditioned on the observation. A numerical integration procedure then updates the action sample according to this velocity. SmolVLA uses a fixed number of inference/integration steps in its standard configuration, commonly 10 steps, with the flow time parameter moving from the noisy endpoint toward the action endpoint.
After the final integration step, the resulting tensor represents the predicted action chunk. It contains the future sequence of robot commands in the action representation the robot will execute open loop.

Full prediction pipeline

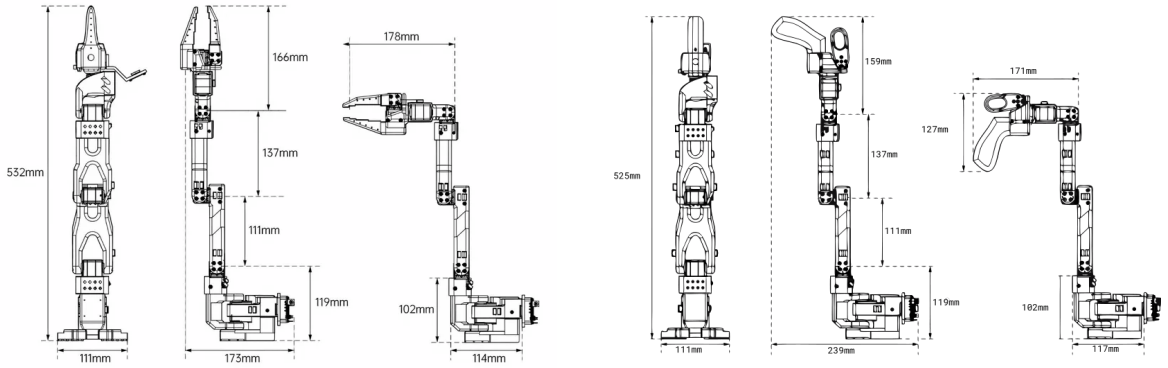
1. Acquire the current camera images, robot state and task instruction.
2. Process these inputs with SmolVLM2.
3. Obtain and cache contextual multimodal features.
4. Initialize a 50-step action chunk with Gaussian noise.
5. Project the noisy action representation into the action-expert hidden space.
6. Run the action expert conditioned on the cached VLM features.
7. Predict the flow velocity field.
8. Integrate the flow and repeat the refinement process for the configured number of inference steps.

9. Obtain the final action chunk.
10. Send the appropriate actions to the robot controller.
11. Acquire a new observation and repeat the process.

2.2.2 SO100 Setup description

The embodiment we chose was the SO100 as it was the one recommanded by HuggingFace. It came with the added benefit of having already been tested with the SmolVLA model and also having a lot of free manipulation data available for further fin tuning. It is a compact, low-cost robotic arm designed for research, education, and experimentation in robotic manipulation. This robot arm consists of a series of motorised pivot links with five degrees of freedom for positioning and orienting the end-effector. Starting from the base, the joints provide rotation of the arm and movement of the shoulder, elbow, and wrist. This configuration gives the robot a workspace suitable for common manipulation tasks such as grasping, picking, placing, and moving objects.
A camera can also be mounted on the wrist for visual input for an operator or an algorithm.

Controlling the position of this arm can prove to be quite challenging, especially with a keyboard or a controller. To avoid this issue, we also used a leader arm, which is structurally identical to the first one but where each joint is replaced with an odometer sensor that can accurately register the angle between each rigid section of the arm. The gripper is replaced with a handle and a trigger that is mapped to the gripping claw of the SO100 arm. This arm can be used to control the robotic one or to register manipulation data-sets for models training.



(a) SO100 follower arm (b) SO100 leader arm

Figure 3: SO100 mechanical designs

Our experimental setup consisted of the two previously mentioned arms, a camera on the wrist of the robotic arm and a top camera filming the hall setup including the arm and the manipulation area from a top angle. We chose a solid grey table as a support to make the task of identifying the object to pick up an manipulate easier.

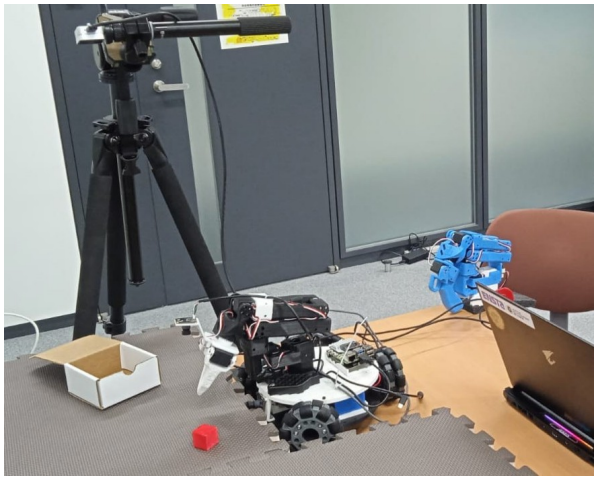


Figure 4: SO 100 manipulation setup overview

2.2.3 Task description and training

The training process consisted of recording multiple examples of a specific task we want the robot to execute. Each episode is characterised by a succession of pair of images captured by the wrist an top camera during the manipulation, forming two videos. For each image pair, the position of the different servo motors is also registered. A prompt is finally used to describe the task.
For the first training, we chose the simple task of picking up a red cube and placing it in a box. We then registered 50 episodes for the task by using the leader arm to control the follower arm during the manipulation.
After training the model for 8 hours the robots was able to complete the task but the accuracy speed and consistency was quite low. We concluded that this was due to a lack of variety in the training data as the examples we provided were "too perfect". During the inference time, as the robot manipulated the object, it could drop of miss the cube, but these scenarios where never present in the training data, so the algorithm would not know how to recover from this situation.

After this realisation, we decided to add 100 more episodes, bringing the total to 150 episodes and we continued the training of the model for an other 12 hours.

2.2.4 Results

We observed at the end of the second training that the movement of the arm was a lot faster and smoother then after the first one and the success rate climbed dramatically. The robot arm was able to consistently pick up the cub from different positions that where outside the immediate view of the wrist camera a drop theme inside the box.

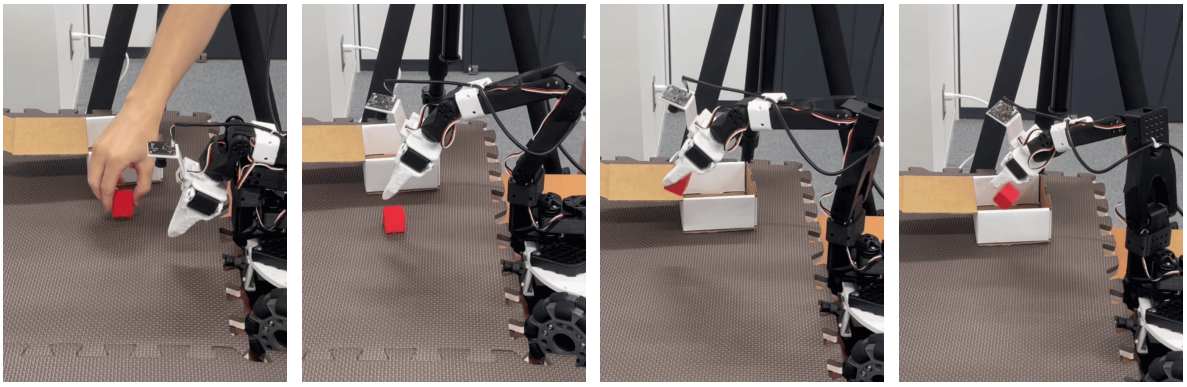


Figure 5: SO100 controled by SmolVLA manipulating a red cub

3 Simulation

3.1 Data needs

As for all deep learning algorithms, the quality and amount of data available for model training is very important. Having access to a good variety of the different tasks the robot will have to execute, as well as insuring the quality of the manipulations themselves, is of paramount importance. Large Language Models (LLM) such as ChatGPT or Claud circumvented this issue by accessing and scraping text and images from web pages across the hall internet. Text and image data are very easily accessible on the internet, but this cannot be said about high quality robotic manipulation, especially ones providing proprioceptive measurement as well as sound recording of the manipulation. Although collective efforts have started to take shape to gather and share large datasets for robotics manipulation training on HuggingFace, they still fail to provide the related sound emitted during the manipulation.

Robotics manipulation learning based only on imitation learning fails to improve upon the specific constraints that a human experiences, allowing only the robot to reproduce the actions as they were shown to him without enabling it to improve upon theme. Fine-tuning the policy with reinforcement learning after having learned the base policy threw supervised learning, enables the policy to innovate upon the demonstration and gain resilience.

Both data generation and reinforcement learning can be archived in a real setup but can end up being very time and money consuming. To learn a new tasks, hundreds or thousands of examples need to be recorded before training can even start and reinforcement learning needs a human in the training loop to manually reset the environment as well as be able to press the emergency stop button is the robot goes off track. All these constraints have lead to the development of simulation environments. Although effective, all of them lack the ability to dynamically generate sounds related to the ongoing interactions inside of the simulation.

This motivated the creating of *RLBench_audio*, the implementation of sound inside the simulation environment RLBench [7]. The following part will dive closer into the specifics of this implementation.

3.2 RLBench

RLBench [7] is a open-source robot learning benchmark and simulation environment designed to train and evaluate robotic manipulation systems. It is built on the CoppeliaSim simulator and was designed with the object of providing an efficient physics based reinforcement learning environment. We chose this environment for the following reasons:

- It already implemented classic robotics embodiment like the Franka Panda we were planning on using
- It provided an easy way to test model inference and train using reinforcement learning.
- It came with great support including accessible tutorials that covered all the main features including task creation.
- Audio VLA [18] had shown that audio implementation was possible inside of the environment

3.3 Sound architecture

Our contribution consists of a complete audio pipeline that allows for the simulation of contact and event based sounds. The pipeline consists of the following main steps:

1. Detects physical collisions at each simulation step using the CoppeliaSim API
2. Identifies the materials of the objects involved (metal, wood, plastic, glass, ceramic, etc.)
3. Retrieves an appropriate sound from a library indexed by (material pair, interaction type, force level)
4. Dynamically modulates the sound: amplitude $\leftarrow$ force, pitch $\leftarrow$ object size, duration $\leftarrow$ contact duration
5. Exposes an audio observation alongside the usual RGB observations

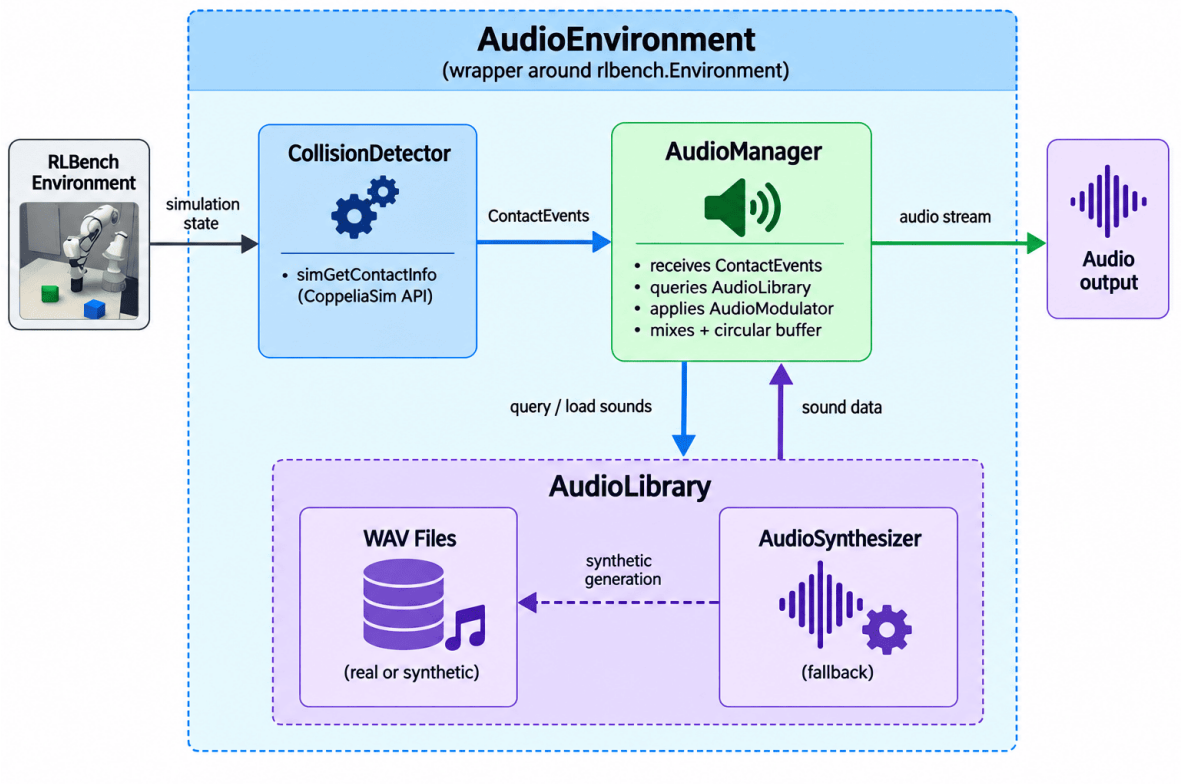


Figure 6: Diagram of the Audio Environment

3.3.1 Collision detection

To Generate the appropriate sound corresponding to the interactions between two objects, the collision detection file queries the following information:

- both objects materials: this information in retrieved from the name of the objects,
- the type of interaction: this corresponds to the nature of the contact, if the tangential relative speed is greater the the normal relative speed, the interaction in classified as scraping else it is qualified as an impact.
- the contact point: the Cartesian coordinate of the contact point
- the newton force
- velocity: the objects relative velocity
- the size of both objects

These information are then used by the audio manager to generate the sound.

3.3.2 Sound library

To obtain realistic sound aimed at reducing the sim-to-real gap, most of the sounds used are pre-recorded and indexed inside of an audio library, the index is defined by the following parameters:

- both objects materials
- the type of interaction

If the library fails to retrieve the sound because it does not exist inside of it, a fallback audio synthesizer is used. Each material is defined by the following parameters:

- density
- elasticity
- fundamental frequency
- decay

Depending on the interaction type, a sound is generated. Note that depending on the interaction, the generated sound may not be appropriate, this should be considered as a placeholder sound.

3.3.3 Sound modulation

Before playing the retrieved sound, a modulation in applied to the signal to more closely correspond to the specifics of the interaction: The amplitude is modulated according to the amplitude of the force and the pitch is modulated according size of the two objects.

3.3.4 Manipulation tacks

Using the *RLBench_audio* environment described above, multiple tasks where created, each focusing on a different aspect of audio oriented manipulation :

- The Buzzer Task 7 is the simplest one, the robot arm has to successively press the buttons on the table until it finds the one making a loud buzzer sound then stop. This task can show if the model can have a basic understanding of sounds and react upon hearing them.
- In Plug Charger Task 8, the arm has to plug a charger inside of a power strip. The challenge comes from the fact that the movement needed to complete the task also partially obscures the wrist camera. The objective is to see if sound could help achieve finer and more precise manipulation in the case of partial camera occlusion such as when manipulating in occluded spaces.

- The Wipe Board Task 9 explores the idea of sound helping retrieve contact information. With only video input, as both cameras are on top of the table, the robot arm has no way of knowing if the sponge is in contact with the board to wipe it. Contact sound could help achieve better results by giving distinct sounds when the sponge is correctly scraping against the board and when it is not.
- The Sort Materials Task 10 testes the capability of the model to differentiate between different materials that look the same base on the impact sound they make. The task gives to the model the name of a material, then the robot arm has to successively pick up and drop the cubes placed on the table, each one making a signature sound based on the material it is made of. Then the robot arm has to pick the right cube and correctly sort it in the box placed in front of it.

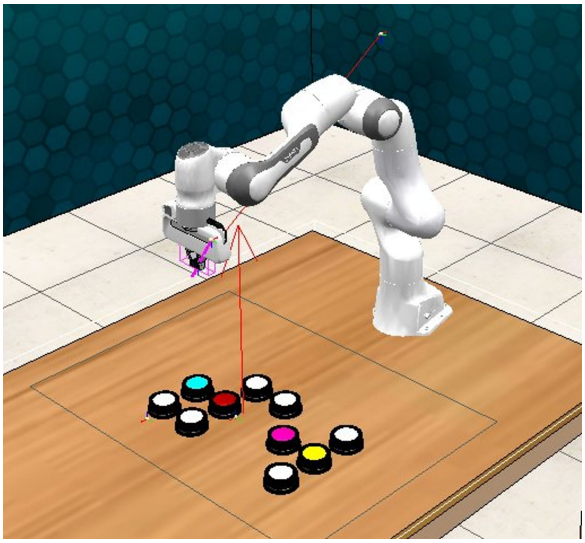


Figure 7: Buzzer Task

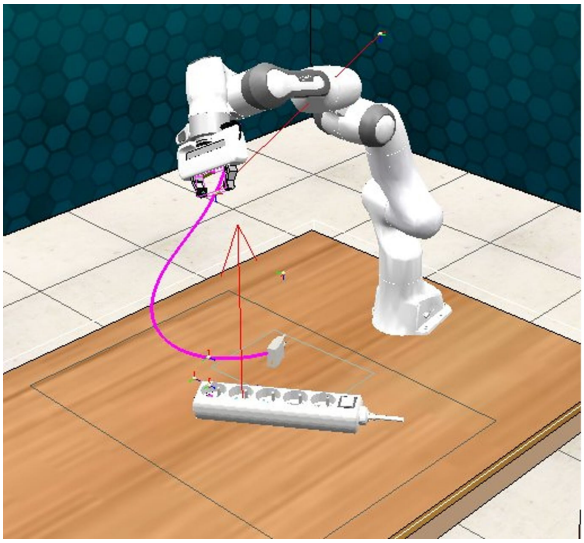


Figure 8: Plug Charger Task

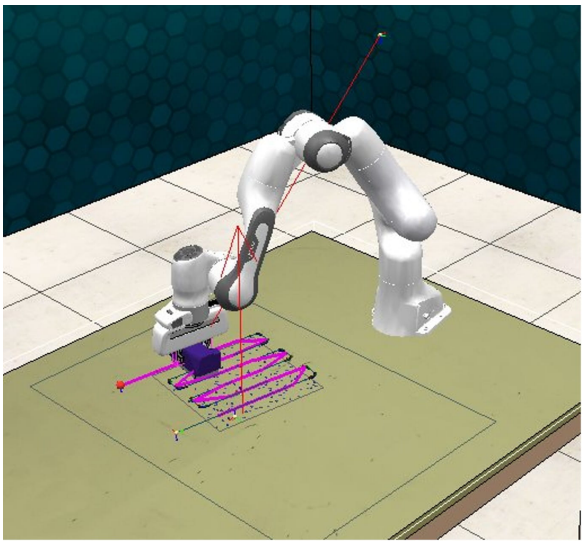


Figure 9: Wipe Board Task

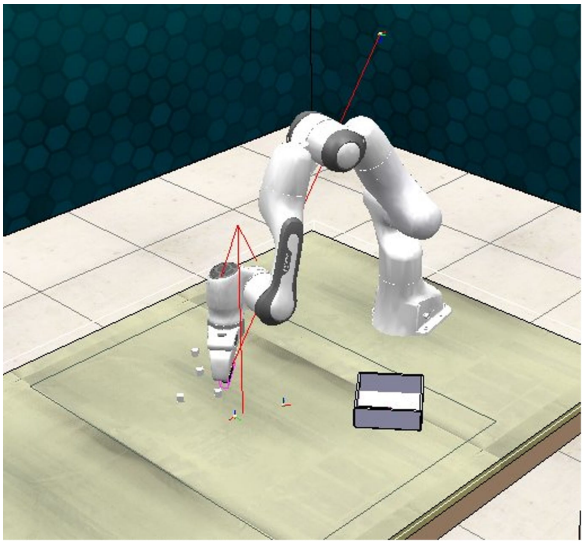


Figure 10: Sort Materials Task

Figure 11: Manipulation tasks inside of the CoppeliaSim simulation environment

RLBench_audio including the physical simulation running in Coppelia Sim, RLBench, and the sound wrapper all run inside of a docker container. The data set creating can be run from any terminal after having specified the task and the number of episodes to be generated. For each episode, the position of the key object is slightly randomized based on predefined rules, the textures of the background elements are dynamically pulled from a collection of predefined textures. The control is the archived by placing key way-points that define positions and actions inside the task. Movement and actions are interpolated between the different key-points to define complete trajectories. Then inverse kinematics is used to compute the desired position of each actuators to achieve the requested movement. Finally the episode is considered to have been successfully completed if a set of predefined conditions are met, for example, 'Is the right cube in

the right box after having follower the trajectory ?'.
If the episode is validated, then it is registered and the program moves on to the next simulated task.

3.3.5 Sound output for the model

At each step of the simulation, the audio manager overlays the sounds generated by the ongoing interactions and stores it inside a buffer and then outputs the last half-second of sound to be used as input for the VLA model.

3.4 Reinforcement learning

Due to a lack of time, reinforcement learning was not fully implemented, although the basic structure was completed, including inference testing, task success evaluation and an automated pipeline.

4 Physical Setup

This part of the project was mainly conducted by Thibault Lucas.

4.1 Franka Panda setup description

The setup we used was build around a Franka Emika Panda 12 robotic arm which had the advantage of being a widely used robotic arm for research, meaning that ample documentation was available. It also had the benefit of not being a mobile robot, meaning that manually resetting a task environment would only involve placing the correct objects on a table and not having to modify the entire layout of a room. This was important as the objective of having a physical setup was not only to have a real robot to test the model on, but also to have a platform to record real world data with. The video input was recorded via two RGBD cameras 15 place in a similar ways as described in 2.2.2, one on the wrist of the arm and on other one on top of the scene.
The sound sensors consisted of a stereo microphone 14 able to pick up and record global sounds, and two piezo-electric microphones 13 attached to the Franka Panda grippers with adhesive putty, capable of recording localised low volume sounds transmitted mainly threw vibrations inside of the end effector, for finer sound understanding [8]. These types of microphones are usually used to record instruments with soundboxes.



Figure 12: FRANKA EMIKA Panda



Figure 13: Piezo-electric Microphone



Figure 14: Zoom H5 4-Track Portable Recorder



Figure 15: RealSense Depth Camera D405

Figure 16: Setup sensors

4.2 Teleoperation

As mentioned previously with the example of the SO100 arm, teleoperation can prove quite difficult if the controller is not adapter to the arm, making fine manipulation almost impossible. After having tested multiple different teleoperation methods including a game controller, keyboard, the SO100 leader arm, none of them provided controls fine and intuitive enough to be pratically used.
The controller we finally settled upon is the on made specifically for the Franka Panda developed by GELLO18 as an opensource project, providing software and hardware to recreate they teleoperation setup. Unfortunately, the components needed to build the leader arm did not arrive in time so we could not test and record real world data with it.

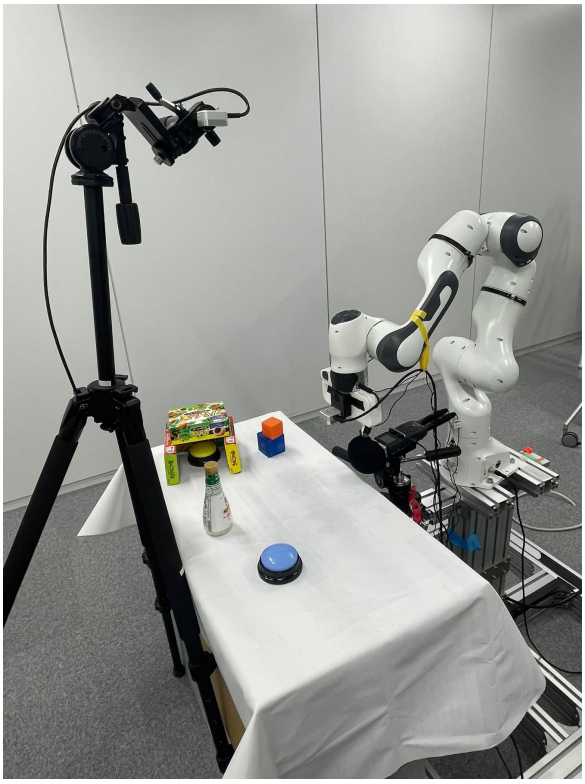


Figure 17: Manipulation setup overview



Figure 18: GELLO leader arm for the Franka Panda

4.3 Data-set recording

We used ROS2 as the main middle ware to communicate between the different sensors and programs. As such *rosbags* where particularly adapter to register the raw data from the manipulation examples. They allow to gather and play back information published on the different topics, recreating the exact timeline of events that occurred during the task completion. This data was then converted to the data-set format used by SmolVLA to be used for training. Contrary to data registered threw the simulation, the real data is fare more time and money consuming but also presents greater quality as it does not reproduce the biases and shortcuts the simulation takes to reduce computation time. We trade quantity for quality. This data is very important to reduce the sim-to-real gap (the performance drop that happens when a robot policy trained in a computer simulation fails or performs poorly when moved to real physical hardware).
In our case, the bulk of the data was generated using our simulation environment as is was fare faster to gather, but ideally we would like to have a third of the total data set be recorded on the real setup.

5 Audio SmolVLA

This part relies heavily on the previously explained architecture of SmolVLA **??**.

5.1 Implementation

To enable our model to understand sounds input, the first step was to incorporate audio into the already existing multimodal language model (SmolLM2). To do this after having acquired the sound, it goes threw an audio projector and encoder. The key constraint is to have the token size of the audio embedding vector to be the same size as the ones from the video and state vectors. The choice we have to make now is how to transform the initial audio information into vectors of the correct hidden size and same embedding space as the one used by SmolLM2. These vectors can then be concatenated with the vector from the other modalities.

We used Audio Spectrogram Transformer (AST) as the audio encoder. It takes a mel spectrogram as input, which represents the evolution of the sound over time and frequency. The frequency axis is mapped to the Mel scale, which better reflects the frequency perception of the human auditory system. The amplitudes are generally represented on a logarithmic or decibel scale, making the acoustic information easier to process.

This representation is particularly suitable for the proposed approach because it provides a compact representation of the audio signal while preserving detailed information in the lower-frequency range, which may be particularly relevant for robotic manipulation sounds. The spectrogram can also be treated similarly to an image, allowing the audio and visual modalities to be processed over corresponding observation periods.

5.2 Training strategies

Training a Vision-Language-Action (VLA) model proved to be a complex and time-consuming process. Because the model combines several components with different roles, training cannot necessarily be performed as a single optimization stage. Instead, it requires several phases and extensive empirical experimentation to determine appropriate training strategies and hyperparameters.
The main factors that need to be considered include the dataset size and type, data masking, number of training steps, learning rate, which components should be frozen, and whether to perform full fine-tuning or use parameter-efficient methods such as LoRA (Low-Rank Adaptation). LoRA keeps the pretrained model weights frozen and introduces small trainable low-rank matrices into selected layers. Only these additional parameters are updated during backpropagation, substantially reducing the number of trainable parameters and therefore the computational and time requirements of training.

Because the proposed model is composed of several specialized blocks, the training was organized into multiple stages. The first training strategy consisted of:

1. Audio projector alignment: the LLM backbone and action expert are frozen while

the audio projector is trained. The objective is to make the audio representation compatible with the other modalities.

2. LLM LoRA training: LoRA is applied to the LLM backbone to allow the model to gradually learn how to incorporate audio information.

3. Full model training with frozen projectors: the projectors are frozen while the remaining components, including the LoRA-adapted LLM, are trained.

Initial experiments revealed several limitations with this strategy. In particular, it was unclear how the model could learn meaningful action-related information during the first stage when the action-producing components were frozen. There was also a question regarding whether the model had sufficient capacity to learn the required tasks while simultaneously processing an additional audio modality.
These observations led to a second, refined training strategy:

1. Initial task training without audio: the complete model is first trained on tasks similar to the final objectives, but with the audio input set to zero. This allows the model to first learn the required robotic motions without having to process audio.

2. Audio projector pretraining: the audio encoder and projector are then trained separately on a larger audio data-set. A classifier is added after the projector, with the objective of predicting the corresponding sound label. This encourages the audio pathway to produce meaningful and discriminative representations.

3. Audio-enhanced training: the three stages of the first training strategy are then performed to progressively integrate the audio modality into the pretrained model.

4. Real-world adaptation: finally, the complete model is fine-tuned for a limited number of steps using real-world episodes. This stage allows the model to adapt to the characteristics of the physical setup and can be included or omitted depending on the experiment.

Overall, this second strategy follows a progressive learning approach: the model first learns how to perform the task, then learns to extract meaningful audio information, and finally learns to integrate this additional modality into its decision-making process.

5.3 Results

Our internship focused mainly on the creation of a complete framework for training and testing machine learning models. Unfortunately we were not able to complete a fully working model as we were in the middle of the training process when the internship ended. Most of what comes after is based on observations made on the preliminary results obtained after the first training tries.

5.3.1 Pipeline Testing

After having generated a data-set from the simulation of about 1000 different episodes across 7 different variations, the raw sensory outputs were able to be converted into our correct data-set format. We chose as a first test to train the model only on the Buzzer Task as in our opinions, it was the simplest one, not requiring a fine understanding of the sound, to be successfully completed. We then trained the model for about 15 hours using the first training method described above. The model was trained on a computer using one Nvidia RTX 6000 Blackwell. Learning parameters where empirically adjusted over several testes to maximise the learning efficiency. Once the model trained, we tested it by launching the pixi environment running the model, in parallel to the docker container running the RLBench simulation environment. Both environment were able to communicate threw a TCP server, the VLA model sending directly the control commands to the simulation and the simulation responding with the sensors outputs. The success of this first step showed that the hall pipeline not only worked but also was quite efficient to use. This was important as the objective with this work is for it to be reused and improved upon by researchers continuing the project.

5.3.2 Model shortcomings

The model we obtained after the first full training was not able to complete a task on its own. We diagnosed many aspects it could be improved upon in the future to hopefully obtain a better working model.

The model we chose was intended as a small, easy to train model. As such, the original SmolVLA paper warned that the model could struggle with long horizon task that required advances planing of movement. Adding a new modality on top of a model that had been optimised is size may have proven to be to much for its limited capabilities. Training a model on a new modality is also quite a hard process that requires large amount of data and a great know-how to pull off. The approach we chose (adding a modality to an already existing model) seemed at the time the easiest but retrospectively, taking a working MLLM capable of sound understanding and adding an action expert may have lead to better results. This approach cloud lead to great improvements in the future.
Newer VLA models also explore new directions to improve there long term planning. One such direction is having the model explicitly output as texte its current understanding of the citation. This output can then be interpreted during the training process to better guide the learning and offers a more comprehensive way of evaluating the strengths and weaknesses of a model. A similar idea is implemented inside of the HEAR framework [10] where the model is equipped with a learned world model that predicts sounds. This is used as a tool to force the model to better understand the types and temporality of sounds as predicting the future necessitates a fine understanding of the past and present event.

5.3.3 Physical setup testing

As explained before, as the model was not good enough, it could have been dangerous to test the model on the real hardware, because unpredictable movement could have damaged the robotic arm or the environment.

5.3.4 Sound limitations

The current process of acquiring sound by recording them on the real setup necessitates to register tens of recordings before generating even the first episode. This approach is not easily scalable, a sample needs to be registered for each pair of material that are susceptible to interact during the simulation and for each type of interaction that may occur. Finally the sound recorded itself varies from on recording hardware to an other.

To circumvent this, a future development of the project, initiated by fellow intern Ando Ko, aims to automatically generate sounds from videos using online training data. The proposed approach is to develop a video-to-sound generation model, potentially based on existing work such as the MMAudio project [2].

5.3.5 Software limitations and future

A limitation of the project is the need to eventually replace RLBench with another simulation environment. RLBench relies on CoppeliaSim, whose licensing conditions restrict its use for non-academic research. This is particularly relevant at AIST, which is not affiliated with a university.
Although RLBench itself has a less restrictive license, its dependence on CoppeliaSim creates limitations for future research and publication. Therefore, adopting a more suitable simulation environment will be necessary for future development.

Although the model was not able to be fully completed, the work provided is still very valuable as it lays out the ground work for further research. Much of what was described can be reused or repurposed for other projects and most of the practical knowledge, expertise, and competence is transposable to the creation of new models in the same environment.

6 Conclusion

This internship explored the integration of sound into Vision-Language-Action policies for robotic manipulation. Starting from the SmolVLA architecture, the project investigated how an additional audio modality could be incorporated into an existing VLA pipeline, while also addressing the lack of suitable tools and datasets for training and evaluating such models. Rather than focusing solely on the model itself, a significant part of the work therefore consisted in developing the infrastructure required to make this research possible.

The first stage of the project consisted in studying machine learning, multimodal models and VLA architectures, followed by experiments with the original SmolVLA model on a physical SO100 platform. These experiments provided a practical understanding of the complete VLA pipeline, from demonstration collection and dataset preparation to training and real-world inference. They also highlighted the importance of dataset diversity, as increasing the number and variety of demonstrations led to substantially more reliable and smoother robot behaviour. This experience provided the foundation for the subsequent development of the audio-augmented model.

A central contribution of the internship was the development of an audio-enabled simulation pipeline based on RLBench and CoppeliaSim. The resulting environment, referred to as $RLBench_audio$, can detect simulated physical interactions, identify the materials and type of contact involved, retrieve or synthesise corresponding sounds, and dynamically modulate them according to the interaction. This makes it possible to generate manipulation episodes containing synchronized visual, proprioceptive and audio information. Several tasks were also designed specifically to investigate situations in which sound could provide information that is difficult or impossible to obtain from vision alone, such as detecting contact, identifying objects through their impact sounds, or reacting to an acoustic event.

In parallel, an audio processing branch was integrated into SmolVLA. An Audio Spectrogram Transformer was used to encode Mel spectrograms, and an audio projector was developed to map these representations into the same embedding space used by the language model. Several training strategies were investigated, ultimately leading to a progressive approach in which the robot first learns the task, the audio representation is subsequently trained, and the different components are then progressively combined. This work established a practical framework for experimenting with audio-augmented VLA policies rather than relying on a purely theoretical extension of an existing architecture.

The project also prepared a physical platform for future sim-to-real experiments. A Franka Panda was equipped with RGB-D cameras, a global microphone and contact microphones, while ROS2 and rosbags were used to synchronise and record the different sensory modalities. A GELLO-based teleoperation system was also investigated for future demonstration collection. Although the teleoperation hardware was not available

early enough to record a substantial real-world dataset, the setup provides the necessary foundation for continuing the project with real robotic data.

The final audio-VLA experiments demonstrated that the complete data generation, conversion, training and inference pipeline could operate successfully in simulation. A dataset of approximately 1,000 simulated episodes was generated and converted into the format required by the VLA model, and the trained policy could communicate with the simulation environment through the complete inference loop. However, the available time did not allow the audio model to reach a level of performance sufficient for a meaningful quantitative evaluation. The first trained model was unable to complete the task autonomously, making it impossible to draw a reliable conclusion about the actual benefit of audio compared with a vision-only baseline.

Consequently, the main outcome of this internship is not a final demonstration that audio improves robotic manipulation, but the establishment of the experimental framework required to answer this question. The work provides reusable components for simulation, sound generation, dataset creation, audio processing, model training and inference, which can serve as a starting point for future experiments.

Several directions remain open. First, the audio generation process should be made more scalable and realistic, particularly by reducing the need to manually record sounds for each material and interaction. Video-to-sound generation is one possible direction for addressing this limitation. Second, the current simulation environment will eventually need to be replaced by a more suitable simulator for continued development and broader use. Third, future work should investigate alternative audio encoders and model architectures, as well as different strategies for integrating audio into the VLA. Finally, systematic experiments should compare audio-augmented and vision-only policies on tasks specifically designed around auditory information, including contact detection, partial visual occlusion and material identification.

Such experiments would make it possible to determine not only whether audio can improve robotic manipulation, but also under which conditions it provides information that cannot be reliably recovered from vision alone. The infrastructure developed during this internship therefore represents a first step towards studying sound as a complementary sensing modality for general-purpose robotic manipulation.

References

[1] 3Blue1Brown. *Neural Networks, Transformers and LLMs*. Educational video series, approximately 3 hours 30 minutes. 2026. URL: https://www.youtube.com/playlist?list=PLZHQObOWTQDNU6R1_67000Dx_ZCJB-3pi (visited on 08/18/2026).

[2] Ho Kei Cheng et al. *MMAudio: Taming Multimodal Joint Training for High-Quality Video-to-Audio Synthesis*. 2025. arXiv: 2412.15322 [cs.CV]. URL: https://arxiv.org/abs/2412.15322.

[3] Jugang Fan et al. "NaVLA2: A Vision-Language-Audio-Action Model for Multimodal Instruction Navigation". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. 2026.

[4] Peter Holderrieth and Ron Shprints. *MIT 6.S184: Flow Matching and Diffusion Models*. MIT course available on YouTube, approximately 6 hours. 2026. URL: https://www.youtube.com/playlist?list=PL57nT7tSGAAXwjhDYcxEycx5W7YoSrZyt (visited on 08/18/2026).

[5] Hugging Face. *LLM Course*. Chapter 1. 2026. URL: https://huggingface.co/learn/llm-course/chapter1/1 (visited on 08/18/2026).

[6] Hugging Face. *Robotics Course*. Course on robot learning and the LeRobot library. 2026. URL: https://huggingface.co/learn/robotics-course/unit0/1 (visited on 08/18/2026).

[7] Stephen James et al. *RLBench: The Robot Learning Benchmark and Learning Environment*. 2019. arXiv: 1909.12271 [cs.RO]. URL: https://arxiv.org/abs/1909.12271.

[8] Junsu Kim and Kyungdon Joo. "SoundAct: Learning Spatial Sound Awareness for Egocentric Robot Manipulation with Stereo Audio". In: *The Workshop on Manipulation Robustness: Towards Human-Level Robustness under Real-World Challenges (ICRA 2026)*. 2026. URL: https://openreview.net/forum?id=Ia1lDHwoR5.

[9] Lingxiao Li et al. "SVA: Towards speech-Enabled vision-Language-Action model". In: *Pattern Recognition* 173 (2026), p. 112915. ISSN: 0031-3203. DOI: https://doi.org/10.1016/j.patcog.2025.112915. URL: https://www.sciencedirect.com/science/article/pii/S003132032501578X.

[10] Chang Nie et al. "Towards the Vision-Sound-Language-Action Paradigm: The HEAR Framework for Sound-Centric Manipulation". In: *arXiv preprint arXiv:2603.16086* (2026).

[11] NVIDIA Research. "Cosmos 3: Omnimodal World Models for Physical AI". In: *Technical Report* (2026).

[12] OpenClassrooms. *Initiez-vous au Deep Learning*. Online course, approximately 8 hours. 2026. URL: https://openclassrooms.com/fr/courses/5801891-initiez-vous-au-deep-learning (visited on 08/18/2026).

[13] OpenClassrooms. *Initiez-vous au Machine Learning*. Online course, approximately 10 hours. 2025. URL: https://openclassrooms.com/fr/courses/8063076-initiez-vous-au-machine-learning (visited on 08/18/2026).

[14] Mustafa Shukor et al. *SmolVLA: A Vision-Language-Action Model for Affordable and Efficient Robotics*. 2025. arXiv: 2506.01844 [cs.LG]. URL: https://arxiv.org/abs/2506.01844.

[15] Ashish Vaswani et al. "Attention is All you Need". In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon et al. Vol. 30. Curran Associates, Inc., 2017. URL: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

[16] Siyin Wang et al. *End-to-end Listen, Look, Speak and Act*. 2026. arXiv: 2510.16756 [cs.AI]. URL: https://arxiv.org/abs/2510.16756.

[17] Siyin Wang et al. "RoboOmni: Proactive Robot Manipulation in Omni-modal Context". In: *arXiv preprint arXiv:2510.23763* (2025).

[18] Xiangyi Wei et al. "Audio-VLA: Adding Contact Audio Perception to Vision-Language-Action Model for Robotic Manipulation". In: *arXiv preprint arXiv:2511.09958* (2025).

[19] Welch Labs. *World Models*. Educational video series. 2026. URL: https://www.youtube.com/playlist?list=PLiaHhY2iBX9jvs1AWv323J0k3YtbLTI8b (visited on 08/18/2026).

[20] Wei Zhao et al. *VLAS: Vision-Language-Action Model With Speech Instructions For Customized Robot Manipulation*. 2025. arXiv: 2502.13508 [cs.RO]. URL: https://arxiv.org/abs/2502.13508.

List of Figures

1 Finger counter model's architecture . . . 3
2 Results of the model . . . 4
3 SO100 mechanical designs . . . 9
4 SO 100 manipulation setup overview . . . 10
5 SO100 controled by SmolVLA manipulating a red cub . . . 11
6 Diagram of the Audio Environment . . . 13
7 Buzzer Task . . . 16
8 Plug Charger Task . . . 16
9 Wipe Board Task . . . 16
10 Sort Materials Task . . . 16
11 Manipulation tasks inside of the CoppeliaSim simulation environment . 16
12 FRANKA EMIKA Panda . . . 18
13 Piezo-electric Microphone . . . 18
14 Zoom H5 4-Track Portable Recorder . . . 18
15 RealSense Depth Camera D405 . . . 18
16 Setup sensors . . . 18
17 Manipulation setup overview . . . 19
18 GELLO leader arm for the Franka Panda . . . 19