

ManiWAV: Learning Robot Manipulation from In-the-Wild Audio-Visual Data (Appendix)

Anonymous Author(s)

Affiliation

Address

email

1 Method Details

1.1 Audio Latency Calibration

Similar to using a clapperboard in film production, we tap on the contact microphone and match the time between the frame when the finger is observed to be in contact with the microphone and the corresponding audio signal captured by the contact microphone. A visual illustration is shown in Fig. 1. The audio is received about 0.06s after the image is received, as a result the total audio latency is $0.17 + 0.06 = 0.23$ s, where 0.17s is the calibrated image latency following the approach in [1].

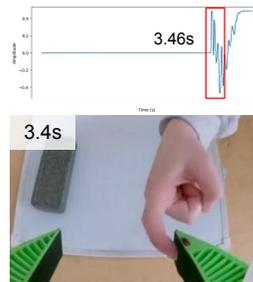


Fig 1

1.2 Model Details

Image Augmentation. Each image is randomly cropped with a 95% ratio and then resized to its original resolution in the replay buffer. To make the learned model more robust to different lighting conditions at test time, we apply ColorJitter augmentation with brightness 0.3, contrast 0.4, saturation 0.5, and hue 0.08.

Transformer Encoder. We use one transformer encoder layer with 8 heads to fuse the vision and audio features. We set feedforward dimension to 2048 and dropout ratio to 0.0.

End-to-End Training Details. For each task, the entire model is end-to-end trained on 2 NVIDIA GeForce RTX 3090 GPUs for 60 epochs, with a batch size of 64. We use the AdamW optimizer with $lr=1e-4$, $\text{betas}=[0.95, 0.999]$, $\text{eps}=1.0e-8$, $\text{weight_decay}=1.0e-6$, and apply EMA (Exponential Moving Average) on the weights.

2 Evaluation Details

2.1 In-the-Wild Data Collection

We collect 274 in the wild data for the bagel flipping task in total, including 52 in a conference room, 37 and 44 in two kitchens, 46 in an office, 76 on lounge tables, and 19 in a cafe, using 4 different pans. Examples of the environments are shown in Fig. 2.

2.2 Result Details

2.2.1 Pouring Task

A breakdown of the success rate for each substep in the pouring task is shown in Tab. 1. ‘Grasp’ is successful if the robot grasps the white cup stably in its end effector, ‘Pour’ is successful if the robot pours all objects in the white cup to the pink cup on the table, ‘Place’ is successful if the robot places the white cup on the table after pouring. By using audio feedback to infer the

	Grasp	Pour	Place
OURS	90	100	90
Vision only	100	0	8.3
1s audio	91.7	30	0
10s audio	100	60	16.7

Table 1: Success Rate Breakdown.



Fig 2: Example Scenes in the In-the-Wild Dataset.

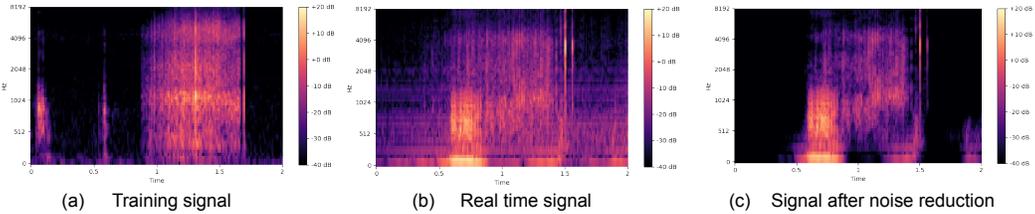


Fig 3: Spectrogram Visualization.

33 object state (whether there’s object in the cup or not), our method is able to reliably guide the policy to pour
 34 objects and place the cup, whereas the baselines either never executes the pour action or the place action,
 35 resulting in low substep success rate. Videos of the policy rollouts can be found on the [project website](#).

36 2.2.2 Taping Task

37 A breakdown of the success rate for each sub-
 38 step in the taping task is shown in Tab. 2.
 39 ‘Touch’ is successful if the robot slides along
 40 the tape while maintaining contact, ‘Sense’ is
 41 successful if the robot chooses the correct tape,
 42 ‘Pick’ is successful if the robot successfully
 43 grasps the tape. ‘Place’ is successful if the
 44 robot successfully places the tape on top of the wires. By leveraging audio feedback to infer the object surface
 45 material (whether the tape is a ‘hook’ or ‘loop’), our method is able to reliably guide the policy to choose the
 46 correct tape whereas the baselines make random decisions, as shown in the ‘Sense’ step success rate. Videos
 47 of the policy rollouts can be found on the [project website](#).

	Touch	Sense	Pick	Place
OURS	90	90	80	80
Vision only	80	30	90	80
Env Mic	80	30	90	40
Noise Reduction	80	40	90	90

Table 2: Success Rate Breakdown.

48 **Noise Reduction Algorithm.** We use the non-stationary noise reduction method introduced in [2]. The
 49 algorithm computes a spectrogram from the audio waveform, and then apply IIR filter forward and backward
 50 on each frequency channel to obtain a time-smoothed version of the spectrogram. A mask is then computed
 51 base on the spectrogram from estimating a noise threshold for each frequency band of the signal/noise. And
 52 finally, a smoothed, inverted version of the mask is applied on the original spectrogram to cancel noise. In our
 53 experiments, we apply this algorithm directly on the real time audio signals before feeding it to the model
 54 for inference. In Fig. 3, we show spectrogram visualization of the real time signal (b) and the signal after
 55 reduction (c). Even though the noise reduction seems to successfully remove most of the robot noises and
 56 some other background noises, it does not preserve the original signal well, and still result in domain gap as
 57 comparing to the training signal (a).

58 **2.2.3 Wiping Task**

59 We listed out each test scenario in the wiping task evaluation and success (1) / failure (0) for each method.

	Test Scenario	OURS	Vision only	MLP fusion	Noise masking	No noise aug
0	T1 (square)	1	1	0 (incomplete)	0 (press)	1
1	T1 (heart)	1	1	0 (incomplete)	1	0 (press)
2	T1 (circle)	1	0 (incomplete)	1	1	1
3	T3 (star)	1	1	1	0 (press)	0 (press)
4	T3 (+1 inch)	1	0 (float)	0 (incomplete)	1	0 (incomplete)
5	T3 (+5 inch)	1	0 (press)	0 (incomplete)	0 (press)	0 (press)
6	T3 (changing height)	1	0 (float)	0 (float)	0 (press)	1
7	T3 (changing height)	1	0 (press)	0 (incomplete)	0 (press)	0 (press)
8	T2 (white noise)	1	1	0 (press)	1	1
9	T2 (white noise)	1	1	1	1	1
10	T2 (construction noise)	1	1	1	0 (press)	0 (press)
11	T2 (music)	0 (float)	1	0 (incomplete)	1	0 (incomplete)
12	T1 (board orientation)	0 (float)	0 (float)	0 (float)	0 (float)	0 (incomplete)
13	T1 (board position)	1	1	0 (incomplete)	1	1
14	T3 (unseen eraser)	1	0 (float)	0 (press)	0 (press)	0 (press)
15	T3 (unseen eraser)	1	0 (incomplete)	1	1	0 (press)
16	T3 (-1 inch)	1	0 (float)	1	1	0 (float)
17	T3 (-1 inch)	1	0 (float)	0 (incomplete)	0 (press)	0 (press)
18	T3 (-2 inch)	0 (float)	0 (float)	1	1	0 (float)
19	T3 (-2 inch)	1	0 (float)	0 (incomplete)	0 (press)	1
	Success rate	85%	40%	35%	50%	35%

Table 3: Wiping Test Scenario Breakdown.

60 ‘Float’ means the robot keeps floating above the board without contacting the board before it wipes off the shape.
 61 ‘Press’ means the robot exerts too much force downward and causes the gripper to bend against the board.
 62 ‘Incomplete’ means the robot fails to follow the shape, either stops wiping early or wipes in wrong location.

63 As we can see from Tab. 3, ‘Float’ and ‘Press’ are most
 64 common in the [Vision only] baseline especially when the
 65 table height is different than training, likely due to the fact
 66 that it’s insufficient to infer contact from the top-down view
 67 wrist-mount camera image alone (as shown on the right).

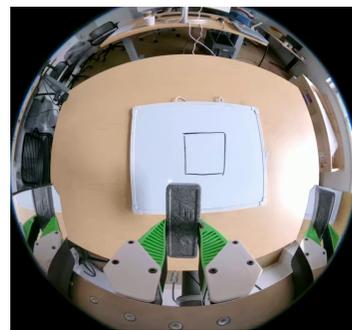


Fig 4: Camera view in the wiping task.

68 The most common failure case in [MLP fusion] is ‘incomplete’,
 69 where the robot stops wiping and releases the eraser before
 70 the shape is completely wiped off. We hypothesize that this
 71 is because simply fusing vision and audio features with MLP
 72 layers lose information that’s crucial for inferring the stage of
 73 the task.

74 Without noise augmentation, the policy exhibits various unex-
 75 pected behaviors including ‘press’, ‘incomplete’, and ‘float’,
 76 because of the big domain gap between training and testing. The policy achieves better performance by simply
 77 masking out the robot noise frequency range in the [Noise masking] baseline, however, it still fails half of the
 78 time and most of the failure cases are ‘pressing’ too hard against the board.

79 **2.2.4 Flipping Task**

80 We listed out each test scenario in the bagel flipping task evaluation and success (1) / failure (0) for each method.

	Test Scenario	OURS	Vision only	MLP policy	ResNet	AVID
0	T1	1	0 (lose)	1	0 (poke)	1
1	T1	1	0 (lose)	0 (displace)	1	0 (lose)
2	T1	1	0 (poke)	1	0 (displace)	0 (displace)
3	T1	1	0 (lose)	0 (poke)	1	1
4	T1	1	1	1	1	0 (lose)
5	T1	0 (lose)	0 (lose)	1	0 (displace)	1
6	T1	1	0 (lose)	1	1	1
7	T1	1	0 (poke)	0 (displace)	0 (poke)	1
8	T1	1	1	1	0 (displace)	0 (poke)
9	T1	1	1	0 (lose)	1	1
10	T1	1	1	1	0 (displace)	1
11	T1	1	0 (poke)	0 (poke)	0 (displace)	1
12	T1	1	0 (poke)	1	1	1
13	T1	1	0 (lose)	0 (lose)	0 (poke)	1
14	T2 (clap)	1	0 (poke)	1	1	0 (displace)
15	T2 (construction noise)	1	0 (poke)	1	0 (stuck)	1
16	T3 (unseen height)	1	1	1	1	0 (displace)
17	T3 (unseen height)	0 (lose)	0 (poke)	0 (lose)	0 (displace)	0 (lose)
18	T3 (unseen height)	1	0 (lose)	1	0 (poke)	0 (displace)
19	T3 (unseen height)	1	0 (displace)	0 (lose)	0 (displace)	0 (displace)
	Success rate	90%	25%	60%	35%	55%

Table 4: Flipping Test Scenario Breakdown.

81 ‘Poke’ means that the robot pokes the spatula on the side of the bagel instead of inserting the spatula between
82 the pan and the bottom of the bagel. ‘Lose’ means the robot loses contact with the bagel before it is flipped,
83 as a result, the bagel falls back to its original side. ‘Displace’ means that the spatula is displaced in the robot
84 end effector as comparing to its initial pose, as a result of the robot keeps moving down instead of switching
85 to slide the spatula along the bottom of the pan.

86 For all T1 scenarios, we randomize the robot initial pose and object positions (e.g. bagel and pan). We can
87 observe that the [Vision only] policy does not generalize well across initial configurations and most failure
88 cases are either poking on the side of the bagel (since it’s hard to infer from the image alone if the spatula is
89 contacting the bottom of the pan), or losing contact with the bagel early before it can be flipped.

90 Using a [ResNet] and [AVID] audio encoder results in the spatula to ‘displace’ most of the times, likely
91 because the model is not sensitive enough to the sound feedback of spatula touching the bottom of the pan,
92 and as a result keeps moving downward and causes the spatula to displace.

93 References

- 94 [1] C. Chi, Z. Xu, C. Pan, E. Cousineau, B. Burchfiel, S. Feng, R. Tedrake, and S. Song. Universal manipula-
95 tion interface: In-the-wild robot teaching without in-the-wild robots. *arXiv preprint arXiv:2402.10329*,
96 2024.
- 97 [2] T. Sainburg. timsainb/noisereducer: v1.0, June 2019. URL <https://doi.org/10.5281/zenodo.3243139>.