Saliency-Aware Quantized Imitation Learning for Efficient Robotic Control

Supplementary Material

1. Experiments Details

1.1. Benchmark Details

Robot Manipulation: We employed the LIBERO benchmark [35] to assess the efficacy of SQIL within the realm of robot manipulation. The LIBERO benchmark includes four distinct task suites, each designed to facilitate life-long learning studies. Our research focused on implementing quantization techniques during the imitation learning process across these suites, subsequently evaluating the robustness and performance of the resulting quantized policies. Each suite comprises 10 distinct tasks accompanied by 50 human teleoperation demonstrations. LIBERO-Spatial evaluates spatial relationship understanding through varied object layouts; LIBERO-Object tests policy performance across different objects within identical layouts; LIBERO-Goal examines task-oriented behavior under consistent object and layout conditions with varied goals; and LIBERO-Long involves comprehensive long-horizon tasks that incorporate diverse objects, layouts, and objectives. To illustrate, the following are examples of tasks from each suite, with corresponding visualizations provided in Fig. 8:

- **LIBERO-Spatial:** "Pick up the black bowl **next to the ramekin** and place it on the plate."
- **LIBERO-Object:** "Pick up **the butter** and place it in the basket."
- **LIBERO-Goal:** "Open the middle drawer of the cabinet."
- **LIBERO-Long:** "Put the black bowl in the bottom drawer of the cabinet and close it."

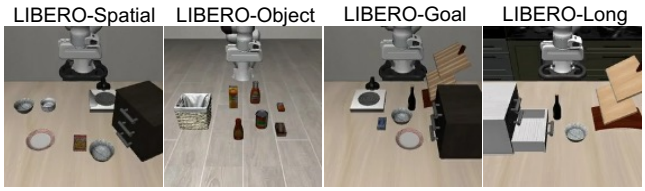

Figure 8. LIBERO benchmark.

In our experiments, we used datasets that were specifically modified for compatibility with the OpenVLA [27] framework, which included enhancements such as high-resolution image processing, image rotation, and the exclusion of unsuccessful demonstrations. We conducted 500 experiments for each task suite.

we extended our evaluations to real-world scenarios. Using the UR5 robot, we configured an environment that mirrors the BridgeDataV2 setup, specifically adapted to resemble a toy sink setting. This setup allowed us to collect data that is comparable in complexity and variability to our simulated environments. We assembled an expert dataset by conducting 20 episodes across the following four tasks, aiming to capture diverse manipulative actions:

- "Put the eggplant into the pot."
- "Put the cup into the plate."
- "Stack the purple cup on the green cup."
- "Lift the eggplant."

Autonomus Driving: For evaluation, we utilized the NoCrash benchmark [8]. This benchmark evaluates the generalization capabilities from Town 1, characterized by its European town setup with single-lane roads and T-junctions, to Town 2, noted for its smaller scale and distinct textural differences. The benchmark includes three traffic density levels: empty, regular, and dense. These levels define the number of pedestrians and vehicles present in each map scenario. We conducted our performance evaluations under the NoCrash-dense setting, using metrics such as the success rate proposed by NoCrash and the driving score from the CARLA LeaderBoard [47], along with rewards derived from CARLA [10]. The success rate is the proportion of routes completed without collisions or blockages, while the driving score is calculated based on penalties for infractions. Our infraction analysis measures occurrences such as pedestrian and vehicle collisions and red light violations per kilometer.

Physics Simulation Tasks: Performance evaluation in physics simulation tasks involved measuring the average return values for each task within the DeepMind Control Suite [46]. Detailed descriptions of each task are as follows:

- **Cartpole Balance**: The task requires controlling a cart to keep a pole upright by moving the cart along a track, focusing on balance and stability.
- **Walker Stand**: This task involves maintaining an upright posture for a bipedal walker robot without it undertaking any additional locomotion.
- **Hopper Stand**: A one-legged robot must remain stable and upright, testing the agent's ability to balance a dynamically unstable object.
- **Cheetah Run**: The goal is to maximize the forward velocity of a quadrupedal cheetah robot, emphasizing speed control and efficient limb coordination.
- **Finger Spin**: The agent must control a robotic finger to spin an unactuated body continuously, assessing precision and control consistency.
- **Humanoid Stand**: The task requires a humanoid robot to maintain a standing position, testing balance and stability in a complex robot with many degrees of freedom.
- **Humanoid Walk**: Extending the humanoid stand task, this requires the robot to walk at a specified speed, focusing on the coordination of bipedal locomotion.

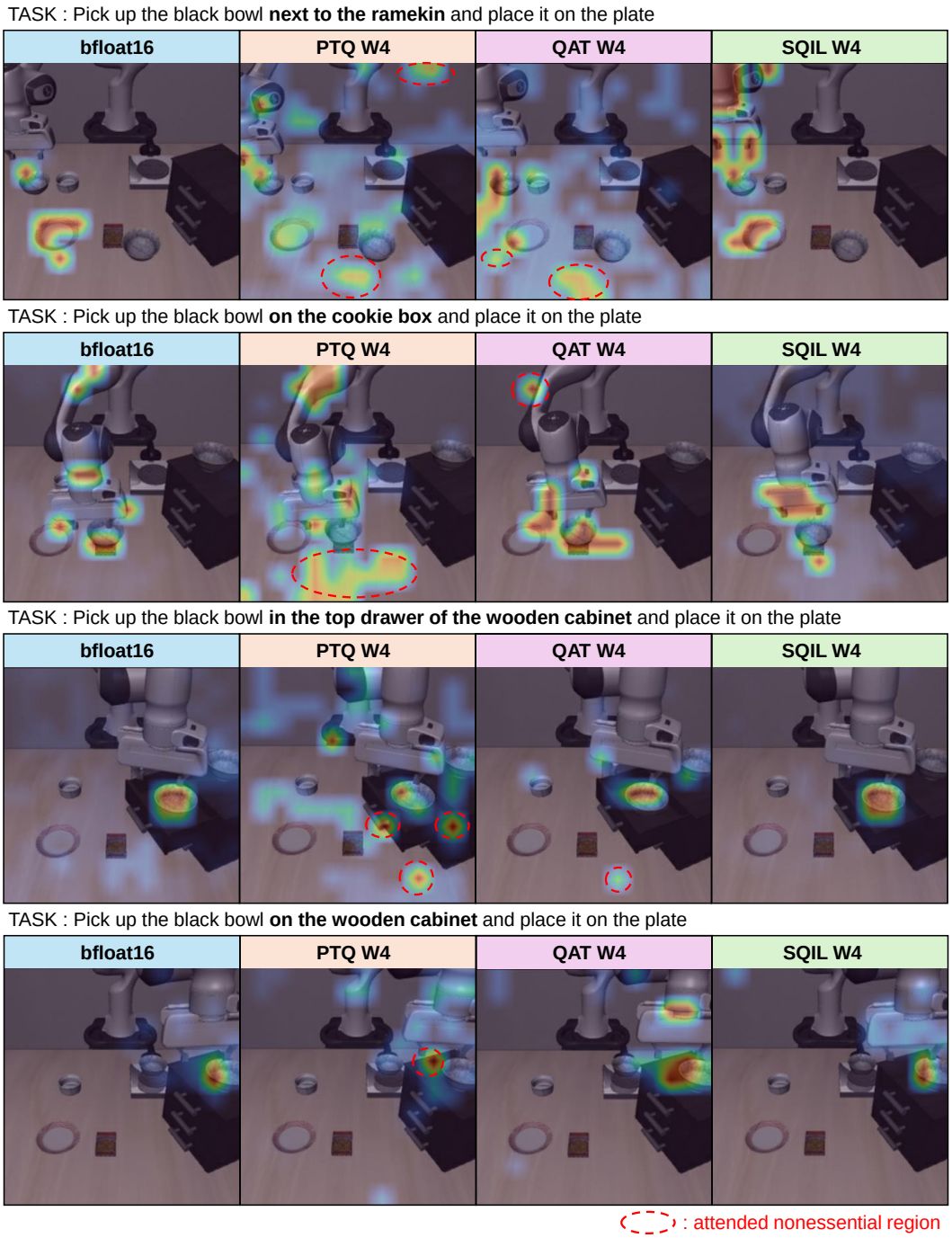

Figure 9. Comparison of attention visualization for tasks successfully completed on the LIBERO-Spatial benchmark.

1.2. Hyperparameter Setting

Robot Manipulation: For robot manipulation tasks, we utilized channel-wise quantization for weights. Both QAT and SQIL models commence training from weights initialized using PTQ (AWQ [34]). QLORA [9] (r=32) is employed to freeze the quantized weights, allowing updates solely to the adaptor, which comprises 110M parameters. The discrepancy metric D used is the average of the L2 distances. For SQIL, the hyper-parameter β is set at 2. Training proceeds wih a learning rate of 5e-4 for a total 50,000 steps.

Autonomus Driving: In autonomous driving, tensor-wise quantization is employed for both weights and activations using LSQ [12]. The discrepancy metric D utilized measures

Method	Success Rate % ↑										
	task1	task2	task3	task4	task5	task6	task7	task8	task9	task10	Avg.
FP	85.3 ± 3.1	92.0 ± 2.8	83.3 ± 1.9	94.0 ± 4.5	72.7 ± 2.2	87.3 ± 3.1	91.3 ± 2.8	81.3 ± 1.5	80.0 ± 2.2	73.3 ± 2.9	84.0
RTN	81.3 ± 1.9	65.3 ± 3.1	74.0 ± 3.8	88.0 ± 2.1	54.7 ± 3.5	81.3 ± 3.0	92.0 ± 2.0	76.0 ± 4.0	60.7 ± 2.9	60.0 ± 3.1	73.3
AWQ	84.7 ± 1.1	84.7 ± 2.1	72.0 ± 3.8	79.3 ± 1.9	74.0 ± 2.8	88.0 ± 1.5	92.0 ± 1.5	**79.3 ± 1.9**	77.3 ± 2.5	69.3 ± 1.9	80.1
QAT	**86.0 ± 1.5**	88.7 ± 3.1	77.3 ± 3.9	82.0 ± 2.9	70.7 ± 2.4	**88.7 ± 1.8**	90.7 ± 2.5	74.7 ± 1.5	74.0 ± 2.0	**76.0 ± 2.1**	80.9
QRD	74.7 ± 3.3	78.7 ± 3.6	76.0 ± 4.1	88.0 ± 2.0	62.7 ± 1.3	75.3 ± 2.4	76.0 ± 4.3	64.7 ± 3.5	46.7 ± 2.1	52.0 ± 4.8	69.5
SQIL	**86.0 ± 3.1**	**92.7 ± 1.8**	**83.3 ± 3.3**	**93.3 ± 4.1**	**74.7 ± 2.2**	86.7 ± 4.0	**92.7 ± 2.3**	76.7 ± 3.4	**79.3 ± 2.0**	73.3 ± 4.1	**83.9**

Table 10. Comparison of success rates across different tasks in LIBERO-Spatial, evaluated on *OpenVLA* with INT4 quantization.

the average of the L2 distances of the policy network's final logits. The quantized policy is trained over 15 epochs with a learning rate of 1e-4. For SQIL, the hyper-parameter β is set at 2.

Physics Simulation Tasks: For physics simulation tasks, tensor-wise quantization for both weights and activations is achieved using LSQ. The discrepancy metric D used is the average of the L2 distances of the policy network's final logits. Training uses demonstration data with a learning rate of 3e-4 and extends over 1,000,000 steps. The hyper-parameter β is set at 2.

2. Attention Map Analysis

Additional visualizations are provided in Fig. 9 for Sec. 4.4, illustrating a broad spectrum of tasks. The full-precision (FP) policy consistently demonstrates precise focus on relevant task objects and their specific locations, particularly where interactions with the robot arm are likely to occur. In contrast, policies quantized using PTQ (AWQ) often misdirect attention towards irrelevant areas, frequently overlooking the critical zones necessary for successful task execution. QAT represents an improvement, more accurately targeting relevant object areas, though it occasionally still attends to non-essential regions. The SQIL significantly enhances this focus, motivating a closer alignment of the quantized policy's attention with that of the FP policy on important state. This alignment contributes to actions that are more likely based on relevant and accurate situational awareness, reflecting the reasoning processes of the FP policy.

3. Ablation Study

Additional Real-World Evaluation (BridgeDataV2). We additionally evaluate quantized policies on the BridgeData V2 setup. As shown in Fig. 10, SQIL recovers performance close to the full-precision policy, outperforming PTQ and QAT.

Quantization Impact Analysis. To assess the distinct impacts of different quantization methods, we separately evaluate each approach, as detailed in Table 10. Applying RTN quantization without any calibration for reducing quantization errors results in a significant average performance degradation of 10.7% compared to the baseline. Using QAT,

	Task	#Trials	#Successes			
			FP	PTQ	QAT	**SQIL**
A	Put Eggplant into Pot	10	7	5	6	6
B	Put Cup into Plate	10	7	4	5	7
C	Stack Purple Cup on Green Cup	10	3	0	1	3
D	Lift Eggplant	10	5	2	4	5

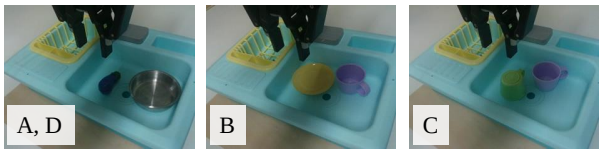

Figure 10. *OpenVLA* real-world UR5 robot evaluation tasks and results with various INT4 quantization.

which utilizes the expert dataset as a ground truth for fine-tuning, shows performance improvements over AWQ. However, applying QRD, which aims to reduce discrepancies with the FP policy without ground truth, actually results in a decrease in performance. In contrast, SQIL losses brings substantial performance improvements, as SQIL acts as a positive guide on mission-critical state.

ViT vs. LLM: Quantization Impact on Performance. As demonstrated in Table 11, the quantization (PTQ) of the ViT component results in minimal performance changes, whereas quantization of the LLM component significantly impacts performance. This suggests that the reasoning capabilities of the LLM are crucial for achieving action decisions similar to those of the FP policy, indicating that SQIL fine-tuning effectively compensates for the losses from quantizing the LLM.

OpenVLA		Suc. Rate ↑
ViT	LLM	
FP	FP	74.0%
INT4	FP	**73.4%**
FP	INT4	71.3%
INT4	INT4	70.8%

Table 11. Quantization impact of ViT and LLM components on the LIBERO for *OpenVLA*.

k	1	2	3	4	5	6
Suc. Rate ↑	49.3%	49.2%	49.4%	49.2%	48.2%	47.7%

Table 12. Performance comparison of timestep frequency k for SIS calculation in *OpenVLA* with SQIL on LIBERO-Long.

Cost for SIS and Finetuning. SIS is a one-time process per existing expert dataset. We exploit spatio-temporal redun-

Discrepancy Metric D			Saliency Weight (β)					Saliency Threshold (p)			
Setting	L2-norm	KL div.	1.5	2	3	4	5	10%	15%	20%	30%
Suc. Rate↑	**73.2%**	72.9%	49.0%	**49.2%**	48.9%	48.2%	47.8%	48.8%	**49.3%**	49.2%	48.5%

Table 13. Comprehensive performance comparisons in the LIBERO-Spatial (D) and LIBERO-Long (β, p) benchmarks for *OpenVLA* with SQIL: discrepancy metrics D, β for state importance, and varying p, which represents the threshold percentage of top saliency scores.

Weighting Noise Dstribution	α_t Gaussian	α_t Laplacian	α_t Uniform	SIS Gaussian
Avg. Success Rate (%)	73.2 ± 0.6	73.2 ± 0.6	73.0 ± 0.4	71.8 ± 1.0

Table 14. SQIL ablations on LIBERO benchmark with *OpenVLA*.

dancy to reduce offline cost: perturbations are applied over a grid of $N \times N$ patches that evenly divide the image, and SIS is computed every k-th frame, reusing previous scores for intermediate frames. Empirically, $N = 8$ and $k = 4$ yielded robust results across all experiments and domains (Table 12). Additionally, SQIL serves as a drop-in replacement for unavoidable fine-tuning in VLA domain adaptation [2] due to variations (e.g., camera viewpoints, lighting, and hardware) using the same expert data and hyperparameters.

Hyperparameter Robustness. We examine the sensitivity of hyperparameters within the SQIL, as detailed in Table 13. Opting for the L2-norm as the discrepancy metric D yields slightly better results than the KL divergence. For SQIL, the hyperparameter β shows stable performance within the 1.5 to 3 range. The threshold T demonstrates resilience, delivering superior performance from the top 10% to 20% range, and maintaining robust results even at 30%, compared to QAT. These results confirm the robustness of our framework's hyperparameters. As shown in Table 14, SQIL exhibits remarkable *stability across hyperparameter variations*, enabling shared settings across domains. Changing saliency noise distribution had negligible effect, while using raw SIS values as weights degraded performance—confirming the importance of our balanced weighting approach.

Aggressive Quantization. We assess performance under more aggressive quantization. Fig. 11 shows that our methods enhance robustness, particularly in INT3 weight quantization of *OpenVLA*.

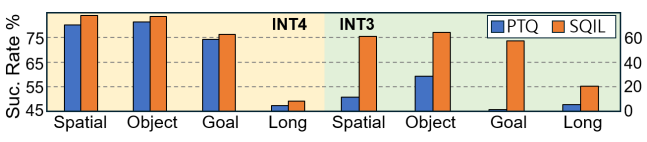

Figure 11. Performance comparison of various quantization methods across different bit precisions on *OpenVLA* in the LIBERO.

Specification	**NVIDIA Jetson AGX Orin**	**Portable Workstation**
	CPU/GPU	**GPU**
Computing Architecture	ARM Cortex-A78AE 12 Cores, 2.2GHz 2048-core NVIDIA Ampere GPU with 64 Tensor Cores	GeForce RTX 2080 Ti
Cache (L1/L2)	CPU: 64KB/256KB GPU: 3MB/4MB	GPU: 64KB/5.5MB
Memory	64GB LPDDR5 SDRAM	11GB GDDR6
ISA	ARM v8.2-A (64 bit) / CUDA 12.0	CUDA 12.0

Table 15. Hardware Specifications of NVIDIA Jetson AGX Orin and Portable Workstation.

4. Implementation Details

4.1. Detailed settings

Device Settings: For our experimental setup, we utilized NVIDIA Jetson AGX Orin 64GB and RTX 2080Ti GPU. NVIDIA Jetson AGX Orin 64GB is equipped with a 12-core Arm Cortex-A78AE CPU, an NVIDIA Ampere architecture GPU. The device runs on Ubuntu 20.04 64-bit LTS OS with GNU gcc/g++ version 9.3.0 with a 30W power mode. For each experiments, we employ the following devices:

- Autonomous Driving (*CILRS*, Weight-Activation Quantization):
  - CPU (FP32, FP16, W8A8): NVIDIA Jetson AGX Orin 64GB (ARM Cortex-A78AE)
  - GPU (FP32, W8A8, W4A4): NVIDIA RTX 2080 Ti
- Vision-Language Action (*OpenVLA*, Weight Quantization):
  - GPU (16-bit, 8-bit, 4-bit): NVIDIA Jetson AGX Orin (2048-core NVIDIA Ampere GPU)

A comprehensive summary of the hardware specifications employed in our experiments is provided in Table 15.

Energy Consumption Measurement: To measure energy consumption in our experiments, we employ the jetson-stats library, which is specifically designed for use with NVIDIA Jetson devices. This library leverages the capabilities of the Triple Channel Voltage/Current Monitor (Texas Instrument INA3221) integrated into NVIDIA Jetson devices [40]. The INA3221 sensor provides detailed measurements of voltage, current, and power consumption for various power rails on the device, allowing for precise monitoring and analysis of the on board power usage.

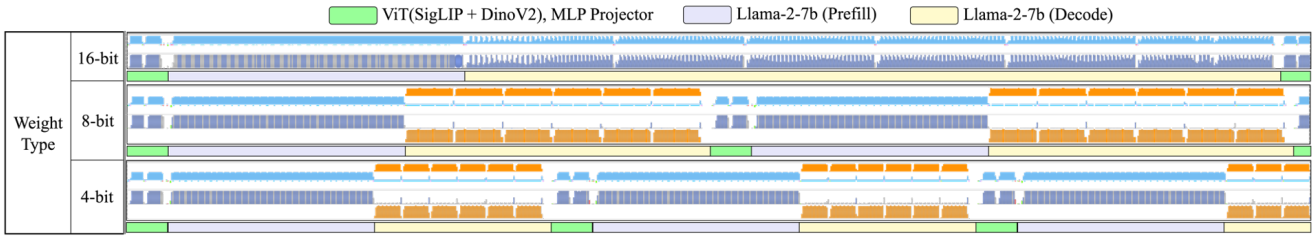

Figure 12. Hardware execution timeline comparison of 16-bit, 8-bit, and 4-bit weight datatypes for *OpenVLA*, measured using *NVIDIA Nsight Systems* over a total duration of 1,000 ms. The visualization highlights differences in execution patterns and latency across the three weight datatypes. Each stage is distinguished by different colors.

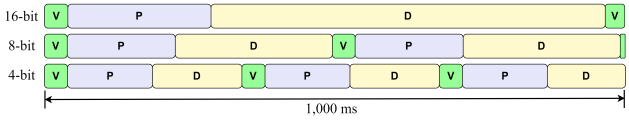

Figure 13. Timeline comparison of *OpenVLA* for 16-bit, 8-bit, and 4-bit datatypes over a total duration of 1,000 ms. V represents the ViT+MLP projector, P denotes the prefill operation, and D refers to the decode operation of the backbone LLM (Llama-2-7b).

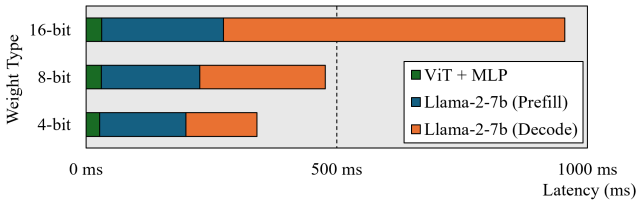

Figure 14. Latency breakdown of *OpenVLA* for 16-bit, 8-bit, and 4-bit weights within a single step, including ViT+MLP, Llama-2-7b (Prefill), and Llama-2-7b (Decode) stages.

4.2. Analysis

We provide a detailed analysis and breakdown of the inference computation time of the *OpenVLA* model. The inference process is divided into four main stages: (1) ViT (SigLIP + DINOv2), (2) MLP Projector, (3) Backbone LLM (Prefill), and (4) Backbone LLM (Decode).

NVIDIA Nsight Systems Hardware Execution Report: NVIDIA Nsight Systems is a comprehensive performance analysis tool that captures and visualizes detailed hardware execution traces. In Figure 12, we present the analysis results of the hardware execution trace recorded for weight types of 16-bit, 8-bit, and 4-bit.

Timeline Comparison & Breakdown: Figure 13 shows a timeline comparison for each weight type based on the actual time ratio, illustrating the proportion of time reduction achieved with different weight types. Figure 14 presents the latency breakdown for each stage and analyzes how reducing the weight type affects the execution time. As shown in our experimental results, the latency of the decode operation in the backbone LLM is reduced the most, achieving up to a 2.5× speedup in overall execution time.

These experimental results confirm that the decode operation in the inference stage of the *OpenVLA* model exhibits memory-bound characteristics, as demonstrated by our data. Weight quantization reduces the amount of data that needs to be transferred between memory and processing units, thereby alleviating memory bandwidth limitations and enhancing overall execution speed. By employing weight quantization techniques, we alleviate these memory-bound limitations, resulting in actual speedups in hardware execution time on Vision-Language Action model.