CoT-VLA: Visual Chain-of-Thought Reasoning for Vision-Language-Action Models

(Q. Zhao et al., CVPR 2025, NVIDIA × Stanford × MIT)

Paper Review

2025.10.31

Jeongyeon Lee

https://cot-vla.github.io/

contents

1. Introduction

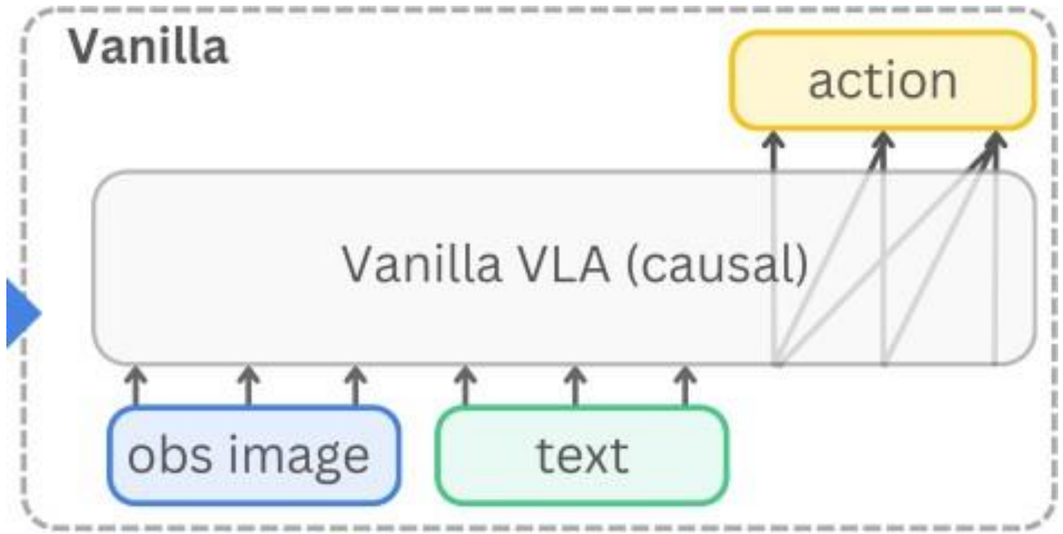
- VLA (Vision-Language-Action Models) :
  image와 text를 입력으로 받아 action을 출력하는 모델.

- CoT (Chain-of-Thought) :
  모델이 정답만 내는 게 아니라, Reasoning 단계를 단계적으로 보여주는 방식.

- CoT-VLA :
  로봇이 복잡한 작업을 수행하기 전에
  "다음 단계는 이런 그림이 되어야지" 하고 시각적으로 미리 생각해보는 모델.
  즉, 행동하기 전에 시각적으로 reasoning 하는 VLA.

2. Related work: VLA

- 기존 VLA 모델의 동작 방식
  - 입력 : 이미지, 텍스트
  - 이후 로봇의 다음 행동을 즉시 예측.
  - 즉, 직접적인 입력-출력 매핑만 함. (중간 추론 X)



2. Motivation

- 기존 VLA 모델의 한계점.

1. 여러 단계를 거쳐야 하는 장기 과제 수행의 어려움.
   : 중간 계획 부재 -> 계획이 없기 때문에 지시를 제대로 지키지 못하거나 엉뚱한 행동으로 빠지기 쉬움.
   Ex) '서랍을 열고, 그 안의 그릇을 꺼낸 뒤, 서랍을 닫아라'

2. 데이터 활용의 한계
   - action까지 모두 labeling된 로봇 시연 데이터를 모으는 것은 매우 어렵고 비용이 많이 듦.
   - action은 없지만 간단한 caption이 달려있는 데이터는 많지만, 이를 활용하지 못함.

2. Related work: CoT

- 기존 VLA 구조와 CoT-VLA 구조 비교

1. 기존 VLA 구조
   input : image, text
   output : action

2. CoT-VLA 구조
   input : image, text
   중간 reasoning : subgoal 이미지
   output : action sequence

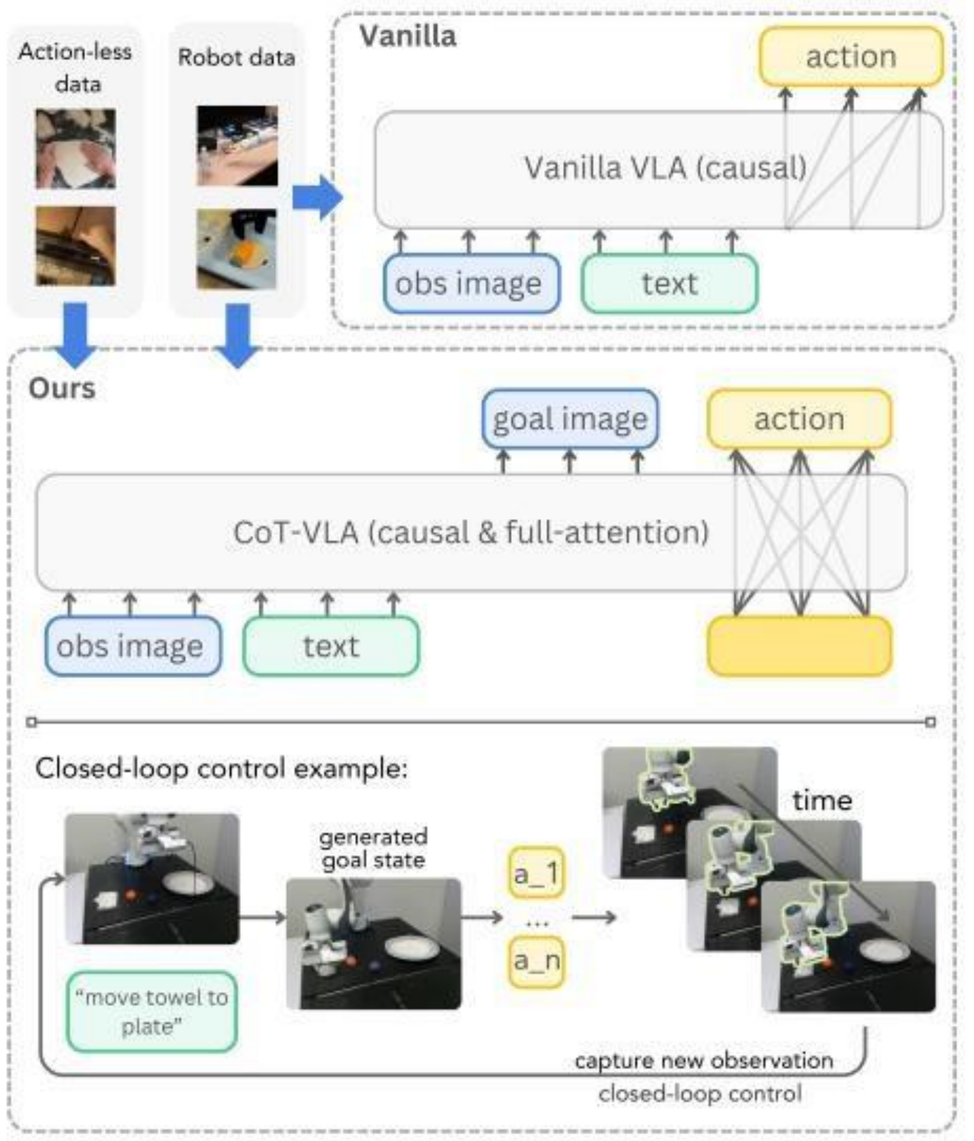


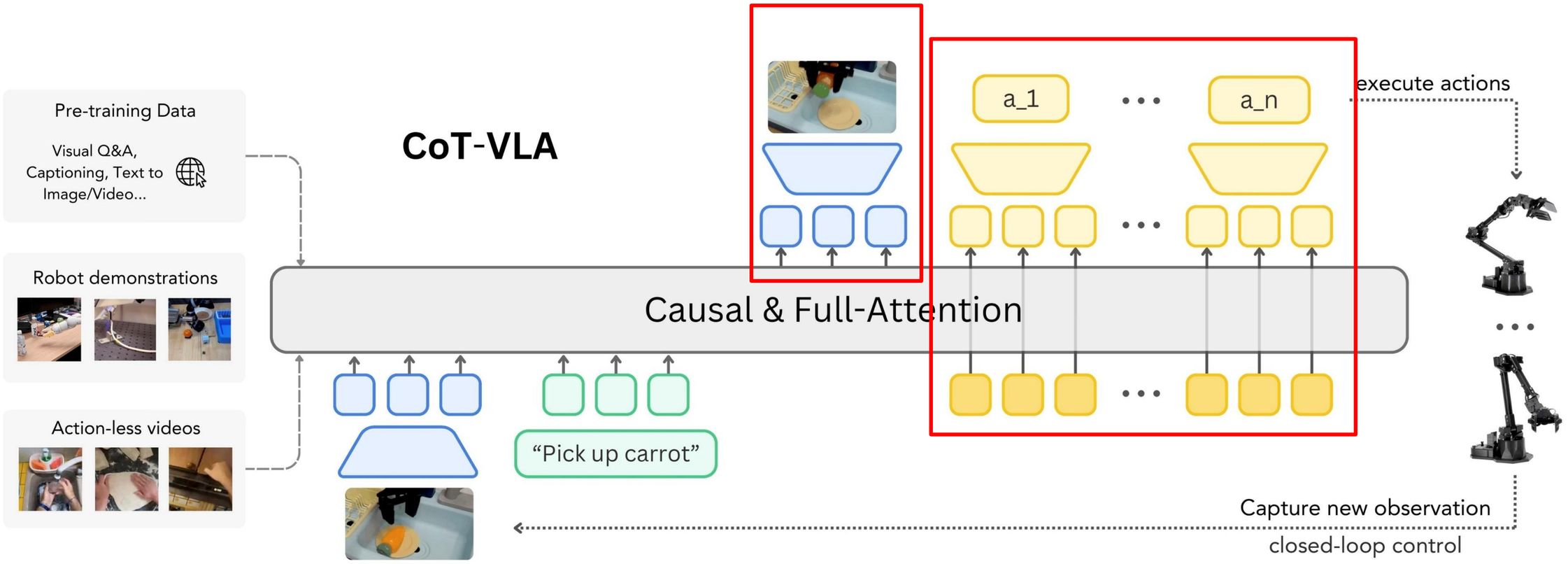
Figure 1. **Comparison between vanilla VLA and CoT-VLA frameworks**. Prior VLA models (top) directly predict robot ac-

2. Motivation

- 기존 CoT의 한계
  - 초창기 텍스트 기반의 CoT를 시도. (LLM)
    Ex) 물건을 잡으려면 -> 팔을 뻗어야 하고, 손가락을 오므려야함.
  - image를 text로 바꾸는 순간 정보가 손실.
    - 공간적 정보 손실
    - 세밀한 제어의 어려움.

3. CoT-VLA

- CoT-VLA의 문제 해결 아이디어
  1. 시각적 사고 단계: subgoal image 생성 (미래 상태 예측).
  2. 행동 실행 단계: 그 이미지에 도달하기 위한 action 시퀀스 생성.



3. CoT-VLA:

(1) subgoal image

- subgoal image 생성
  - 현재 화면과 언어 지시가 주어지면 모델은 앞으로 이렇게 되어야 한다는 미래 장면을 그림으로 그려봄.
  - subgoal image의 역할 : 몇 프레임 뒤의 목표에 맞는 상태로 그림을 먼저 만들어내는 것.
  - Ex) “가지를 통에 담아라”는 지시가 들어오면, subgoal image는 가지가 로봇 손에 들려 있는 장면이 됨.

Figure 5. **Task execution examples for LIBERO, Bridge-V2, and Franka-Tabletop using CoT-VLA.** For each task: Left: text instruction (l) and initial state (s_0^{obs}). Middle: generated intermediate goal states ($\hat{s}_t$) demonstrating visual chain-of-thought reasoning, where each goal image is conditioned on both the instruction and the most recent observation. Right: final state (s_T^{obs}) upon task completion. Complete execution trajectories are available in the supplementary video.

3. CoT-VLA:

(2) action sequence

- action sequence 생성
  - subgoal image와 현재 장면을 동시에 보면서, 해당 상태에 도달하기 위해 필요한 action sequence 를 예측함.
  - action chunking : 짧은 action sequence를 여러 개 뽑아서 실행 -> 자연스러운 제어 가능.

Figure 5. **Task execution examples for LIBERO, Bridge-V2, and Franka-Tabletop using CoT-VLA.** For each task: Left: text instruction (l) and initial state (s_0^{obs}). Middle: generated intermediate goal states ($\hat{s}_t$) demonstrating visual chain-of-thought reasoning, where each goal image is conditioned on both the instruction and the most recent observation. Right: final state (s_T^{obs}) upon task completion. Complete execution trajectories are available in the supplementary video.

3. CoT-VLA:

(3) Training procedure

- action 라벨이 없는 일반 영상 데이터 사용 가능.
  - subgoal image 는 앞 image와 뒷 image의 쌍으로 학습됨. -> action 라벨이 없어도 훈련이 가능.

- 대규모 비디오 데이터셋 활용.
  - Epic-Kitchens나 Something-Something V2 같은 대규모 비디오 데이터셋 활용.
  - -> 다양한 장면 변화 패턴 학습.

- EPIC-KITCHENS [27]:
- 기관: University of Bristol 등.
- 내용: 1인칭(egocentric) 시점의 주방일 비디오. (e.g., "칼을 집는다") '행동 레이블은 없고' 텍스트 설명만 있음.
- 목표: Subgoal 생성 훈련 전용. 로봇 팔 제어 데이터는 없지만, "칼을 집는다"라는 명령 전후의 시각적 변화를 학습.

- Something-Something V2 [20]:
- 기관: TwentyBN.
- 내용: "물체를 왼쪽에서 오른쪽으로 민다" 같은 기본 동작 비디오 모음. 역시 '행동 레이블 없음'.
- 목표: L_{visual} 훈련 전용. 다양한 물체와 동작에 대한 시각적 인과관계 학습.

3. CoT-VLA

- CoT-VLA의 구체적인 구성 요소
  1. RQ-VAE를 통한 효율적인 이미지 토큰화
  2. Hybrid attention
  3. Action Discretization (액션 이산화)

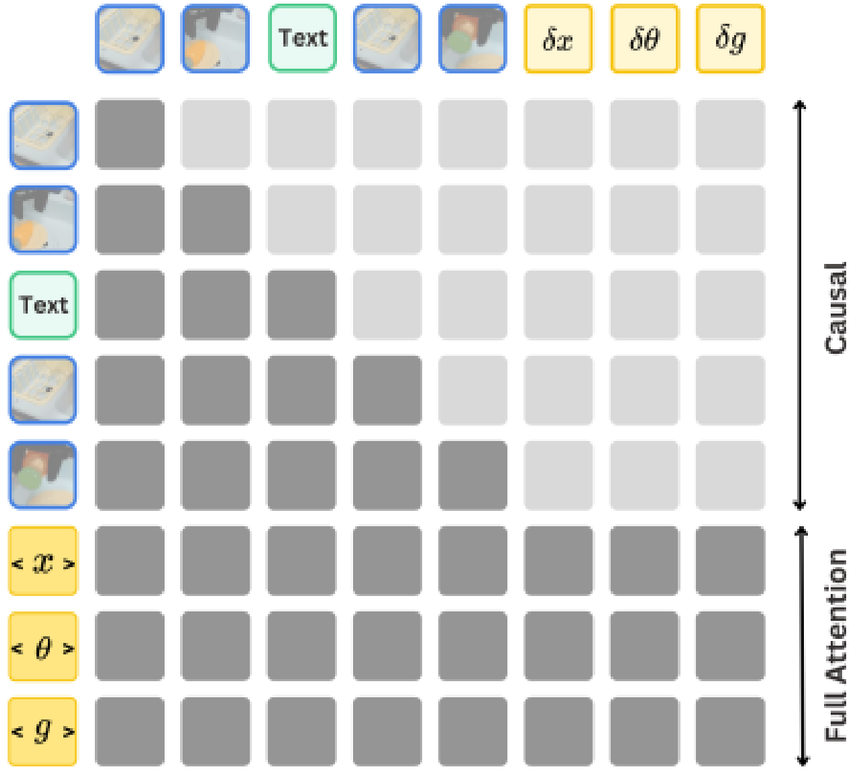


Figure 3. **Hybrid attention mechanism in CoT-VLA**. We use causal attention for image or text generation and full attention for action generation. $[x]$, $[\theta]$ and $[g]$ are special tokens for parallel decoding of actions.

3. CoT-VLA

- CoT-VLA의 구체적인 구성 요소
  1. RQ-VAE를 통한 효율적인 이미지 토큰화
  2. Hybrid attention
  3. Action Discretization (액션 이산화)

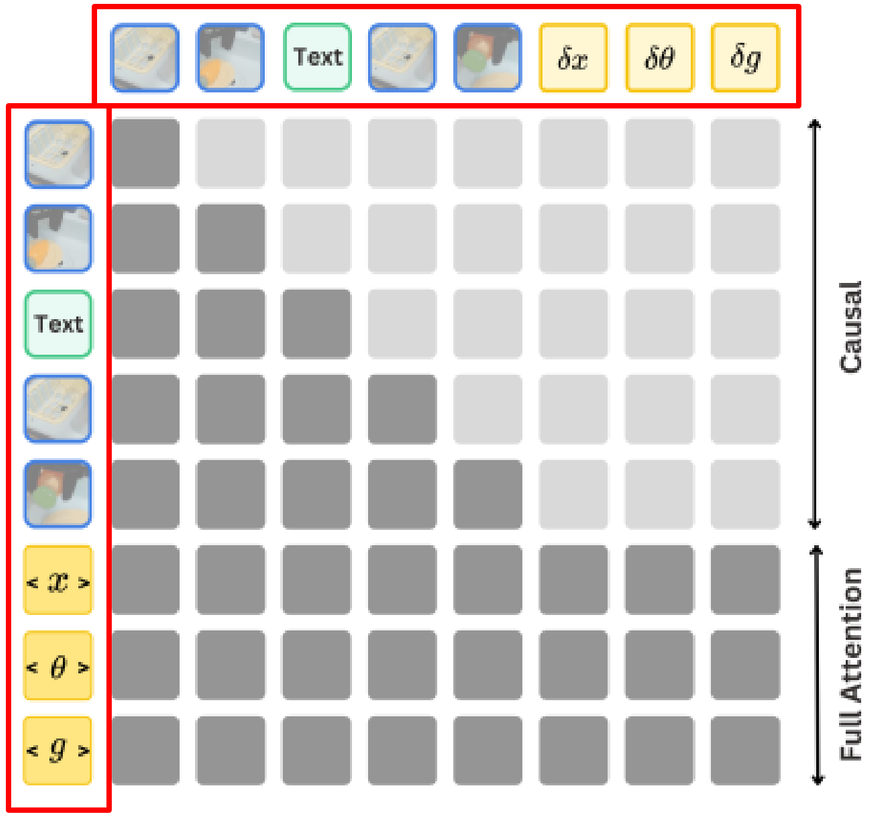


Figure 3. **Hybrid attention mechanism in CoT-VLA**. We use causal attention for image or text generation and full attention for action generation. $[x]$, $[\theta]$ and $[g]$ are special tokens for parallel decoding of actions.

3. CoT-VLA

1. RQ-VAE를 통한 효율적인 이미지 토큰화

- CoT-VLA의 기반 모델
  : VILA-U (언어 토큰과 시각 토큰을 하나의 문장처럼 처리하는 통합 모델)
  - RQ-VAE(Residual Quantization VAE) 기법
    - VQ-VAE (Vector Quantization) 기법에 "이미지를 한 번에 토큰화하지 않고 점진적으로 잔차를 줄여가면서 더 세밀하게 표현하는 방식"을 추가한 기법.
    - : 이미지를 discrete tokens으로 만듦. -> 이미지도 LLM의 입력으로 넣을 수 있게 됨.
      - discrete tokens : 이미지를 시각적 단어 사전(코드북)에서 가장 가까운 단어의 번호로 바꾸어 문장처럼 토큰화.
    - Residual step : 여러 개의 코드북을 겹쳐 쓰는 방식. -> 더 정밀한 토큰 표현 얻을 수 있음.

- 이미지 토큰화 과정 : 256x256 해상도의 이미지를 16x16x4 토큰으로 분해
  - 4 -> RQ-VAE의 깊이인 Residual step.

Ex)
- Codebook 1: '서울' (큰 범위)
- Codebook 2: '강남구' (1단계 잔차)
- Codebook 3: '삼성동' (2단계 잔차)
- Codebook 4: '코엑스' (3단계 잔차)

3. CoT-VLA

- CoT-VLA의 구체적인 구성 요소
  1. RQ-VAE를 통한 효율적인 이미지 토큰화
  2. Hybrid attention
  3. Action Discretization (액션 이산화)

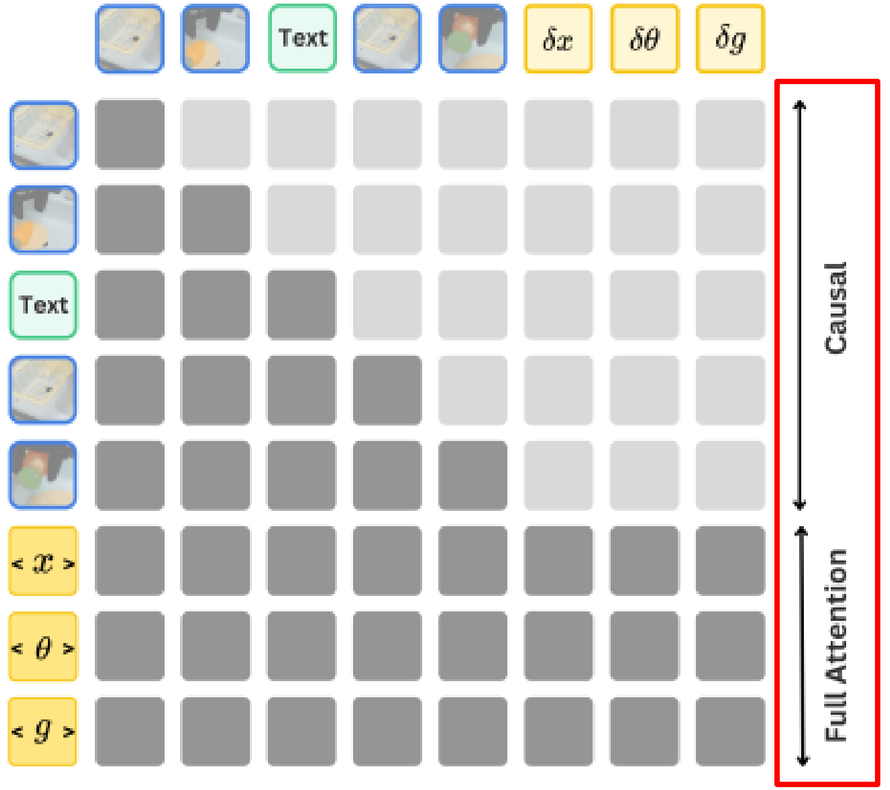


Figure 3. **Hybrid attention mechanism in CoT-VLA**. We use causal attention for image or text generation and full attention for action generation. $[x]$, $[\theta]$ and $[g]$ are special tokens for parallel decoding of actions.

3. CoT-VLA

2. Hybrid attention

- Image/text 생성 -> causal attention
- action 제어 -> full attention 사용

- causal attention
  - 자기 회귀 방식 : 항상 앞쪽 토큰만 보고 뒤를 예측.
  - 텍스트 생성, 이미지의 픽셀을 순차적으로 찍어낼 때 사용.

- full attention
  - 모든 토큰이 서로를 동시에 볼 수 있게 하는 방식.
  - 로봇 행동 = 여러 차원으로 동시에 구성
    - 로봇 팔의 위치, 회전 각도 등
  - 각 차원이 상호작용 -> 안정적인 행동 시퀀스 생성 가능.

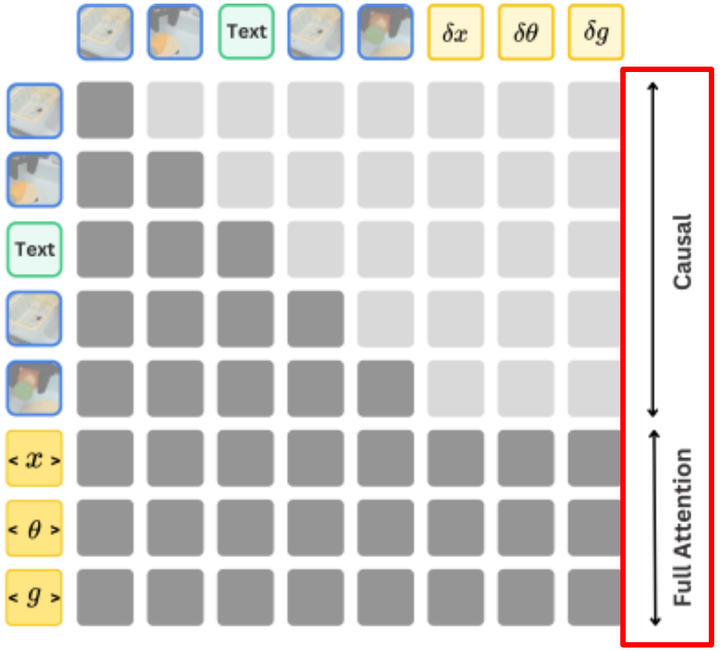


Figure 3. **Hybrid attention mechanism in CoT-VLA**. We use causal attention for image or text generation and full attention for action generation. $[x]$, $[\theta]$ and $[g]$ are special tokens for parallel decoding of actions.

3. CoT-VLA

- CoT-VLA의 구체적인 구성 요소
  1. RQ-VAE를 통한 효율적인 이미지 토큰화
  2. Hybrid attention
  3. Action Discretization (행동 이산화)
     : 로봇 동작을 토큰처럼 다룰 수 있게 됨.
     - VILA-U (LLM)는 '단어(Token)' 같은 이산적인 데이터만 처리 가능.
     - 로봇 팔의 실제 움직임은 연속적인 실수값.
     - Ex) x축으로 0.0132cm 이동.
     - '연속적인 행동 값'을 '이산적인 토큰'으로 번역 해야함.

3. CoT-VLA

3. Action Discretization (행동 이산화)

- 7-DoF Action : 로봇 팔의 행동은 7개의 숫자로 표현됨.
  x, y, z 위치 변화 + 3축 기준 회전 변화 (roll, pitch, yaw) + 그리퍼 열림/닫힘. (위치 3 + 자세 3 + 그리퍼 1)
  Gripper state : 로봇의 손 부분, 집게가 얼마나 열려있는지 혹은 닫혀있는지를 나타내는 단 하나의 숫자 값

1. Binning : 7개의 차원에 대해 전체 훈련 데이터셋에서 1%~99% 범위의 최솟값/최댓값을 찾음.

2. 이 범위를 균등하게 256개의 구간으로 나눔.
   ex) x축의 이동값이 -1.0~+1.0 사이라고 하면, 'bin 1'은 -1.0, 'bin 128'은 0.0, 'bin 256'은 +1.0

3. Token Repurposing
   : 기존 텍스트 토크나이저 (LLM의 단어사전)에서 가장 적게 사용되는 단어 토큰 256개를 재활용함.
   (새로운 '행동 토큰' 256개를 만드는 것 x)
   ex) LLM에게는 'x축 0.0 이동'이 'zZz'라는 단어 토큰을 예측하는 것과 동일한 작업이 됨.

즉, 7-DoF 행동 a_i 는 7개의 토큰으로 변환됨. m 길이의 Action-chunk는 7*m개의 토큰 시퀀스가 됨.

3. CoT-VLA

3. Action Discretization (행동 이산화)

(+) Q. Binning에서 왜 0%, 100%가 아니라 1%, 99%?
A. 데이터의 극단값에 의한 왜곡 막기 위해서.

(+) Q. Token Repurposing에서 왜 가장 "적게" 사용되는 단어를 사용?
A. 이러한 토큰들은 사전 훈련된 '언어적 의미'를 거의 가지고 있지 않기 때문.
기존 의미와 충돌 없이 '로봇 행동'이라는 새로운 의미를 빠르고 안정적으로 할당 가능.
ex) 이 'the'는 문법적인 'the'? 로봇 행동 'the'?

3. CoT-VLA

- 학습 목표 및 손실 함수
  1. 기존 VLA 방식
  2. CoT-VLA 접근 방식 – subgoal image, action sequence 생성
  3. 손실 함수 (L_visual, L_action)

3. CoT-VLA

- 학습 목표 및 손실 함수
  1. 기존 VLA 방식

$$\hat{\mathbf{a}}_t \sim P_\theta(\mathbf{a}_t|\mathbf{s}_t, l)$$

- **목표**: 현재 시점의 행동 a_t를 예측.
- **입력**:
  - s_t: 현재 관찰 이미지 (current observation)
  - l: 언어 명령 (language instruction)
- P_θ: 파라미터 θ를 가진 VLA 모델.

⇒ 모델이 현재 이미지(s_t)와 명령(l)을 보고, **중간 생각 없이** 바로 다음 행동(a_t)을 예측(출력).

3. CoT-VLA

- 학습 목표 및 손실 함수

  2 . CoT-VLA 접근 방식 – subgoal image , action sequence 생성

$$\hat{\mathbf{s}}_{t+n} \sim P_\theta(\mathbf{s}_{t+n}|\mathbf{s}_t, l) \quad (2)$$

$$\{\hat{\mathbf{a}}_t, \ldots, \hat{\mathbf{a}}_{t+m}\} \sim P_\theta(\{\mathbf{a}_t, \ldots, \mathbf{a}_{t+m}|\mathbf{s}_t, l, \mathbf{s}_{t+n}) \quad (3)$$

  - **목표:** n 프레임 뒤의 **미래 Subgoal 이미지** s_{t+n}을 생성(예측).
  - **입력:**
    - s_t: 현재 관찰 이미지
    - l: 언어 명령
  - **훈련 데이터:** 로봇 데이터(D_r) + **행동 없는 비디오**(D_v)

⇒ 모델이 현재 이미지와 명령을 보고, '시각적 생각' 을 통해 **중간 목표 지점의 모습**(s_{t+n})을 그림.

3. CoT-VLA

- 학습 목표 및 손실 함수

  2 . CoT-VLA 접근 방식 – subgoal image , action sequence 생성

$$\hat{\mathbf{s}}_{t+n} \sim P_\theta(\mathbf{s}_{t+n}|\mathbf{s}_t, l) \quad (2)$$

$$\{\hat{\mathbf{a}}_t, \ldots, \hat{\mathbf{a}}_{t+m}\} \sim P_\theta(\{\mathbf{a}_t, \ldots, \mathbf{a}_{t+m}|\mathbf{s}_t, l, \mathbf{s}_{t+n}) \quad (3)$$

  - **목표**: m 길이의 **행동 시퀀스** $\{a_t, \ldots, a_{t+m}\}$를 예측. 여러 개의 행동을 묶음으로 예측 -> 안정적
  - **입력**:
    - s_t: 현재 관찰 이미지
    - l: 언어 명령
    - s_{t+n}: **방금 생성한(훈련 시에는 정답) Subgoal 이미지**
  - **해석**: "모델이 현재 상태(s_t)와 **최종 목표(l), 방금 생각한 '시각적 중간 목표(s_{t+n})'까지** 함께 참고하여, 이 목표에 도달하기 위한 행동 묶음($\{a_t, \ldots, a_{t+m}\}$)을 예측"
  - **훈련 데이터**: 행동 레이블이 필요하므로 **로봇 데이터(D_r)만** 사용.

3. CoT-VLA

- 학습 목표 및 손실 함수
  3. 손실 함수 (L_visual, L_action)

$$\mathcal{L}_{\text{visual}} = -\sum_{j}\sum_{d=1}^{D} \log P_\delta(k_{jd}|k_{j,<d}) \tag{4}$$

  - **목표:** Subgoal 이미지를 잘 생성하도록 모델(Depth Transformer P_δ)을 훈련.
  - **배경:** RQ-VAE를 통해 여러 겹(Residual)의 이산적인 토큰으로 변환.
  - **변수:**
    - j: 이미지의 각 위치 (e.g., 16×16 그리드의 한 칸)
    - d: Residual 깊이 (1부터 D까지, 논문에선 $D = 4$)
    - k_{jd}: 위치 j의 d번째 깊이의 **정답 토큰**
    - $k_{j,<d}$: d번째 깊이 이전(1부터 $d-1$까지)의 토큰들

  ⇒ 이미지의 모든 위치(j), 모든 깊이(d)에 대해, **이전 깊이의 토큰들($k_{j,<d}$)을 보고 다음 깊이의 정답 토큰 (k_{jd})을 맞추게 함.**

  - 표준적인 Autoregressive (자기회귀) 이미지 생성 모델의 Cross-Entropy Loss. 즉, "정답 Subgoal 이미지 토큰이 나올 확률을 최대화함."

3. CoT-VLA

- 학습 목표 및 손실 함수
  3. 손실 함수 (L_visual, L_action)
  - L_{action} : 실제로 로봇이 어떻게 움직였는지가 담긴 행동 레이블

$$\mathcal{L}_{\text{action}} = -\sum_{i=1}^{m} \log P_\theta(\mathbf{a}_t ... \mathbf{a}_{t+m} | l, s_t, s_{t+n}) \qquad (5)$$

  - **목표:** 행동 시퀀스를 잘 예측하도록 모델(P_θ)을 훈련.
  - **변수:**
    - $a_t ... a_{t+m}$: **정답 행동 시퀀스 (토큰들)**

⇒ 주어진 입력(현재 이미지 s_t, 명령 l, Subgoal 이미지 s_{t+n}) 하에서, **정답 행동 시퀀스($a_t ... a_{t+m}$)가 나올 확률을 최대화.**

  - 표준적인 Sequence-to-Sequence 모델의 Cross-Entropy Loss.

3. CoT-VLA

- 학습 목표 및 손실 함수
  3. 손실 함수 (L_visual, L_action)

$$\mathcal{L} = \mathcal{L}_{\text{action}} + \mathcal{L}_{\text{visual}} \qquad (6)$$

⇒ 모델이 두 가지 일을 동시에 잘하도록 훈련. (1) **Subgoal 이미지 잘 그리기** ($\mathcal{L}_{visual}$), (2) **그 Subgoal에 도달하는 행동 잘하기** ($\mathcal{L}_{action}$). 두 Loss를 더해서 한 번에 최적화.

- L_visual : 시각적 추론. '정답 행동'이 필요 없고, '미래 이미지'만 있으면 됨.

-> 행동 없는 대규모 비디오 데이터까지 활용 가능.

- L_action : 행동 학습. '정답 행동'이 있는 로봇 데이터로만 훈련함.

4. Experiment

- LIBERO 벤치마크 실험
  - 목표: 시뮬레이션에서 베이스라인(Diffusion Policy, Octo, OpenVLA) 대비 성능 비교.
  - 공간 인식(Spatial), 객체 조작(Object), 목표 달성(Coll), 장기 계획(Long)
  - 평균 81.1%의 성공률
  - Long-horizon 과제에서 69%의 성공률 -> 시각적 추론이 장기 작업 계획에 도움이 됨.

	Average (↑)	Spatial (↑)	Object (↑)	Goal (↑)	Long (↑)
Diffusion Policy	72.4 ± 0.7%	78.3 ± 1.1%	**92.5 ± 0.7%**	68.3 ± 1.2%	50.5 ± 1.3%
Octo fine-tuned	75.1 ± 0.6%	78.9 ± 1.0%	85.7 ± 0.9%	84.6 ± 0.9%	51.1 ± 1.3%
OpenVLA fine-tuned	76.5 ± 0.6%	84.7 ± 0.9%	88.4 ± 0.8%	79.2 ± 1.0%	53.7 ± 1.3%
CoT-VLA-7B (ours)	**81.13 ± 0.6 %**	**87.5 ± 1.4%**	91.6 ± 0.5%	**87.6 ± 0.6%**	**69.0 ± 0.8%**

Table 1. **LIBERO benchmark experimental results.** For each task suite (Spatial, Object, Goal, Long), we report the average success rate and standard error across 3 seeds with 500 episodes each. CoT-VLA achieves the best or competitive performance across all LIBERO benchmarks suites compared to baseline approaches. The bolded entries correspond to highest success rates while underlined entries correspond to second-highest.

4. Experiment

- WindowX 로봇팔을 활용한 Bridge V2 실험
  - 시각 일반화, 동작 일반화, 의미적 일반화, 언어적 그라운딩의 네 가지 범주에서 성능을 평가
  - "Put carrot on plate"처럼 새로운 위치에서의 동작을 요구하는 과제에서는 60% 성공률로 오픈 VLA의 45%보다 우수

Category	SUSIE	Octo	OpenVLA	CoT-VLA
Visual	30%	35%	**75%**	65%
Motion	10%	10%	45%	**60%**
Semantic	20%	0%	40%	**50%**
Language	40%	40%	**75%**	70%

Table 2. **Bridge-V2 Comparison.** Success rates across four generalization categories, with 10 trials per category and partial credit scoring following [29]. **Visual**: "put eggplant into pot" with cluttered environments; **Motion**: "put carrot on plate" with height variations; **Semantic**: "take purple grapes out of pot"; **Language**: "put eggplant or red bottle into pot".

4. Experiment

- Frank Tabletop 환경 실험
  - 목표: 적은 데이터(10~150개)로 새로운 실제 로봇에 적응하는 능력 테스트.
  - 여러 지시문이 혼합된 복잡한 과제에서도 가장 높은 평균 성공률.
  - 사전 훈련(OpenX + Action-less video)을 통해 학습한 시각적 추론 능력이, 적은 데이터만으로도 새로운 환경에 빠르게 적응하는 데 도움이 됨.

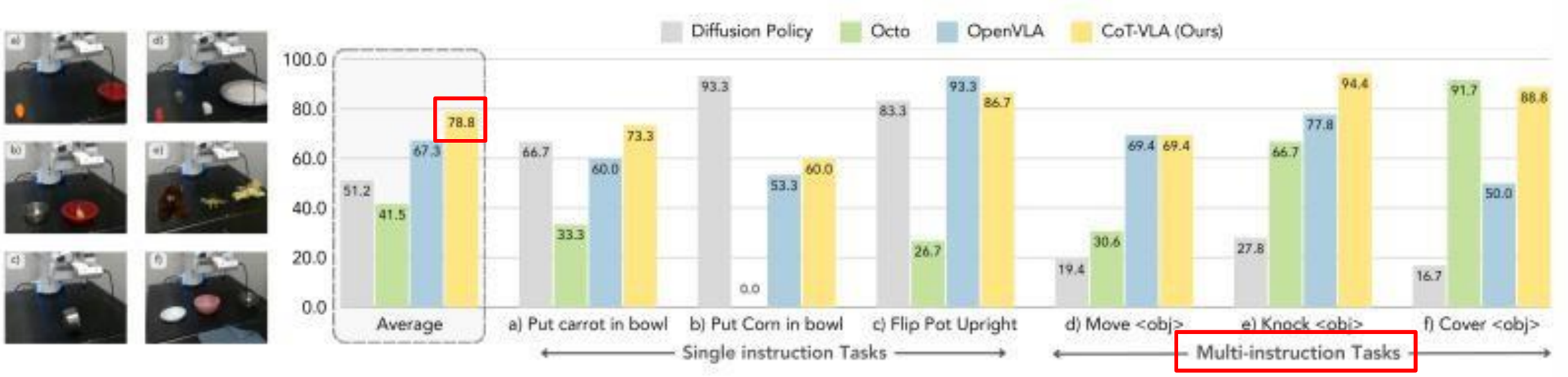
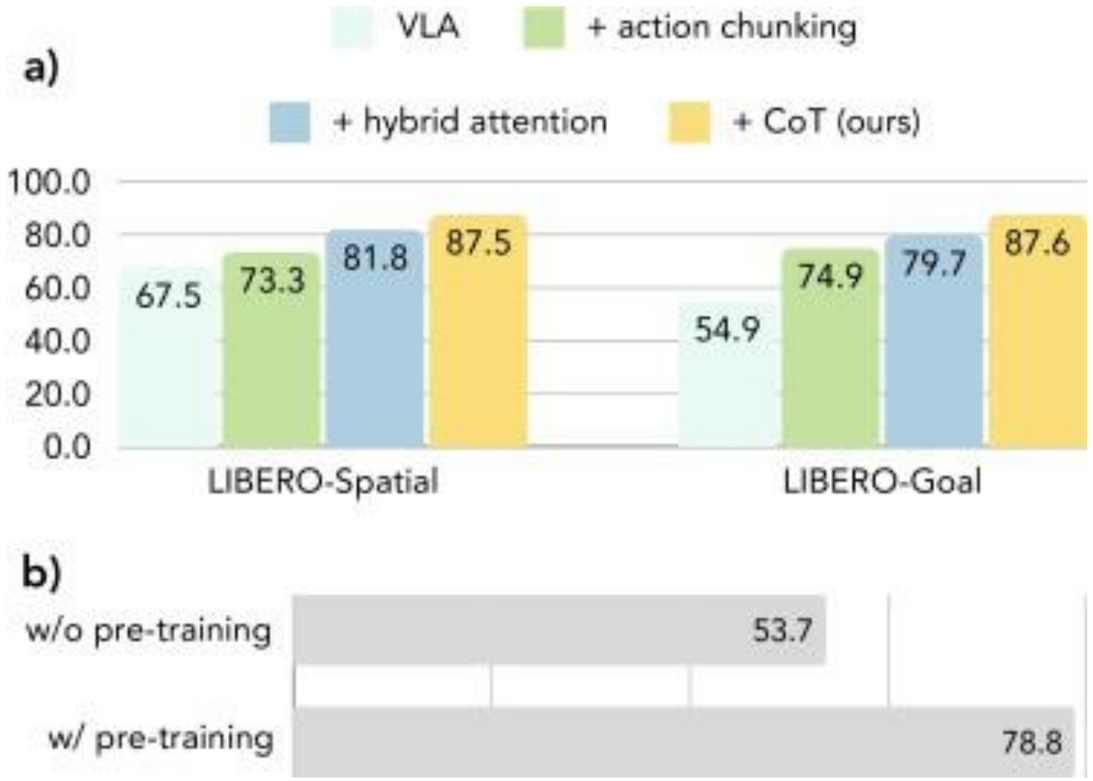


Figure 4. **Franka-Tabletop comparisons.** Evaluation across six distinct manipulation tasks, with separate models trained per task. Left: Representative initial states for each task setup. Right: Task-specific success rates and cross-task averages for our method and baselines. CoT-VLA achieves best average performance and demonstrates strong capabilities in both single-instruction and multi-instruction scenarios.

4. Experiment

- CoT-VLA 핵심 요소별 성능 변화 분석



a)

- Action chunking과 Hybrid attention도 성능 향상에 기여했지만, Visual CoT(시각적 사고) 자체도 5.7%p라는 유의미한 성능 향상을 '추가로' 가져옴.

b)

- w/o pre-training (53.7%):
- 'Without Pre-training' (사전 훈련 없이)
- OpenX 데이터셋과 행동 없는 비디오(EPIC-KITCHENS 등)를 사용한 대규모 사전 훈련 단계를 모두 건너뜀.
- 기반 모델인 VILA-U를 바로 Franka Tabletop의 적은 데이터에 파인튜닝했을 때의 성능.

- w/ pre-training (78.8%):
- 'With Pre-training' (사전 훈련 포함)
- 논문에서 제안한 방식 그대로 대규모 사전 훈련을 모두 수행한 뒤, Franka-Tabletop 데이터에 파인튜닝했을 때의 성능.

4. Experiment

- CoT-VLA의 현재 병목(bottleneck) 위치
  - 모델을 일부러 처음 보는 어려운 작업(out-of-distribution task)에 투입.
  - 조건 1: 모델이 스스로 생성한 Subgoal Image를 사용 (현재 방식).
  - 조건 2: 사람이 직접 제공한 "정답" Subgoal Image (Ground-Truth)를 사용.

  - (Subtask 1 기준) 모델이 스스로 생성한 Subgoal을 썼을 땐 성공률이 20%였는데,
    완벽한 "정답" Subgoal을 주자 성공률이 60%.

  - CoT-VLA의 행동 예측 능력은 이미 충분히 강력함.
  - 즉, 현재 모델의 성능을 가로막는 병목은 행동이 아니라 Subgoal Image를 생성하는 '시각적 추론 능력'임.

	Sub-task 1	Sub-task 2
Generated Goal Images	20%	0%
Ground-truth Goal Images	**60%**	**40%**

Table 3. **Better visual reasoning helps.** Success rates compar-

- 한계점
  - 특정 상황에서의 성능 불안정
    - 특정 상황에서 action chunking 구조 때문

  - 추론 속도
    - Subgoal Image를 매번 생성해야 하기 때문에, Vanilla VLA보다 추론 속도가 **평균 7배 느림.**

  - Subgoal Image의 불확실성
    - 행동의 오작동 가능성 : 중간에 잘못되면 뒤에도 꼬임.
    - 새로운 불확실성의 원인

5. My Experiment

- VLA
- 3D vision

+ Implementation Details

- **Subgoal Horizon:**
- u_l: Subgoal 이미지를 샘플링할 최소 프레임 간격
- u_u: Subgoal 이미지를 샘플링할 최대 프레임 간격
- 행동x : [20], [27]

Dataset	Weight	u_l	u_u
Bridge [16, 60]	24.14%	5	10
RT-1 [3]	6.90%	5	10
TOTO [81]	10.34%	20	24
VIOLA [83]	10.34%	15	20
RoboTurk [42]	10.34%	1	2
Jaco Play [43, 51]	10.34%	10	15
Berkeley Autolab UR5 [8]	10.34%	5	10
Berkeley Fanuc Manipulation [82]	10.34%	10	15
Something2Something [20]	3.45%	5	7
EPIC-KITCHEN-100 [27]	3.45%	5	7

Table 4. Dataset Weights and Hyperparameters

Hyperparameter	Pre-training
Learning Rate	1e-4
LR Scheduler	Cosine decay
Global Batch Size	2048
Image Resolution	256 × 256
Action Token Size	10
Epoch	10

Table 5. Hyperparameters for pre-training

- For fine-tuning on LIBERO [37] and Franka-Tabletop [29] experiments, we fine-tune the model (LLM backbone, projector, depth transformer) with constant learning rate 1e-5 for 150 epochs.