Learning Fine-Grained Bimanual Manipulation with Low-Cost Hardware

RSS 2023

양현서

July 12, 2024

About

- Author: Tony Z. Zhao, Vikash Kumar, Sergey Levine, Chelsea Finn

- Conference: RSS 2023

Motivation

- **Fine manipulation tasks** such as
  - threading cable ties
  - slotting a battery

Motivation

- **Fine manipulation tasks** such as
  - threading cable ties
  - slotting a battery

- **Requires**
  - precision
  - careful coordination of contact forces
  - closed-loop visual feedback

Low cost and imprecise hardware for fine manipulation tasks

Suggestion

Low-cost system that performs end-to-end **imitation learning** directly from real demonstrations, collected with a custom teleoperation interface

Introduction

- Fine manipulation tasks require **precision and coordination**

Introduction

- Fine manipulation tasks require **precision and coordination**
- Current systems are **expensive and complex**

Introduction

- Fine manipulation tasks require **precision and coordination**
- Current systems are **expensive and complex**
- Goal: Develop a low-cost, effective system for bimanual manipulation

Introduction

- Fine manipulation tasks require **precision and coordination**
- Current systems are **expensive and complex**
- Goal: Develop a low-cost, effective system for bimanual manipulation
- Key contributions:
  - **Low-cost** hardware setup
  - Novel imitation learning algorithm **(ACT)**
  - Successful demonstration on various tasks

System Design - Low-cost hardware

- ViperX 6-DoF robot arms
- 3D printed “see-through” fingers, gripping tape
- **Cost: <$20k**

System Design - Design principles

- Versatile

System Design - Design principles

- Versatile
- User-friendly

System Design - Design principles

- Versatile
- User-friendly
- Repairable

System Design - Design principles

- Versatile
- User-friendly
- Repairable
- Easy-to-build

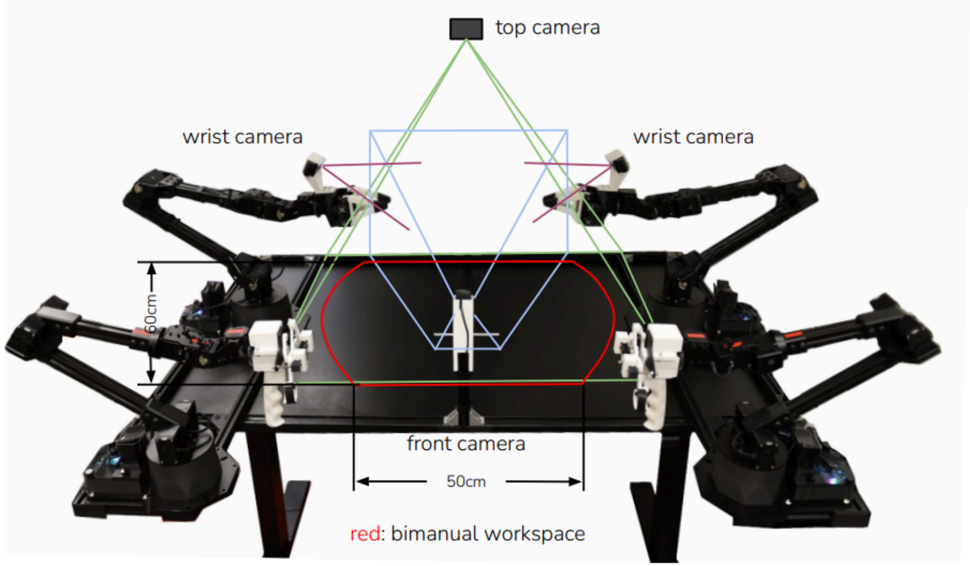
System Design - Design principles - Teleoperation setup

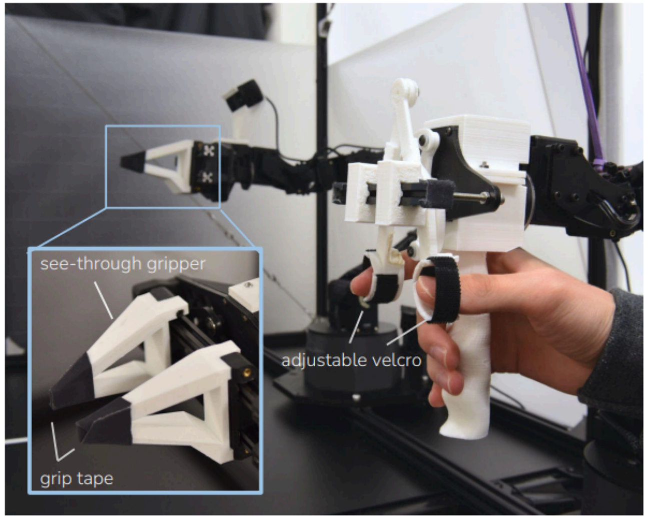
- Joint-space mapping for control

- High-frequency control (50Hz)

Joint space mapping for control

- Directly maps “Leader” joint angles to “Follower” joint angles
- Solves IK failing problem





Imitation Learning Algorithm

Imitation Learning Algorithm

- **Challenges**:
  - Compounding errors in policy
  - Non-stationary human demonstrations

Imitation Learning Algorithm

- **Challenges**:
  - Compounding errors in policy
  - Non-stationary human demonstrations

- **Solution: Action Chunking with Transformers (ACT)**:
  - Predicts sequences of actions (chunks)
  - Reduces effective horizon of tasks
  - Uses temporal ensembling for smoothness

Training ACT on a New Task

- Record leader joint positions as actions

Training ACT on a New Task

- Record leader joint positions as actions

- Observations: follower joint positions, 4 camera feeds

Training ACT on a New Task

- Record leader joint positions as actions

- Observations: follower joint positions, 4 camera feeds

- Train ACT: predict future actions from observations

Training ACT on a New Task

- Record leader joint positions as actions
- Observations: follower joint positions, 4 camera feeds
- Train ACT: predict future actions from observations
- Test: use policy with lowest validation loss

Training ACT on a New Task

- Record leader joint positions as actions
- Observations: follower joint positions, 4 camera feeds
- Train ACT: predict future actions from observations
- Test: use policy with lowest validation loss
- **Challenge: Compounding errors**

Action Chunking

- **Groups individual actions into units** for efficient storage and execution

Action Chunking

- **Groups individual actions into units** for efficient storage and execution

- Reduces the **effective horizon** of long trajectories

Action Chunking

- **Groups individual actions into units** for efficient storage and execution

- Reduces the **effective horizon** of long trajectories

- Every k steps, the agent receives an observation and generates k actions

Action Chunking

- **Groups individual actions into units** for efficient storage and execution

- Reduces the **effective horizon** of long trajectories

- Every k steps, the agent receives an observation and generates k actions

- Mitigates issues with non-stationary demonstrations

Action Chunking

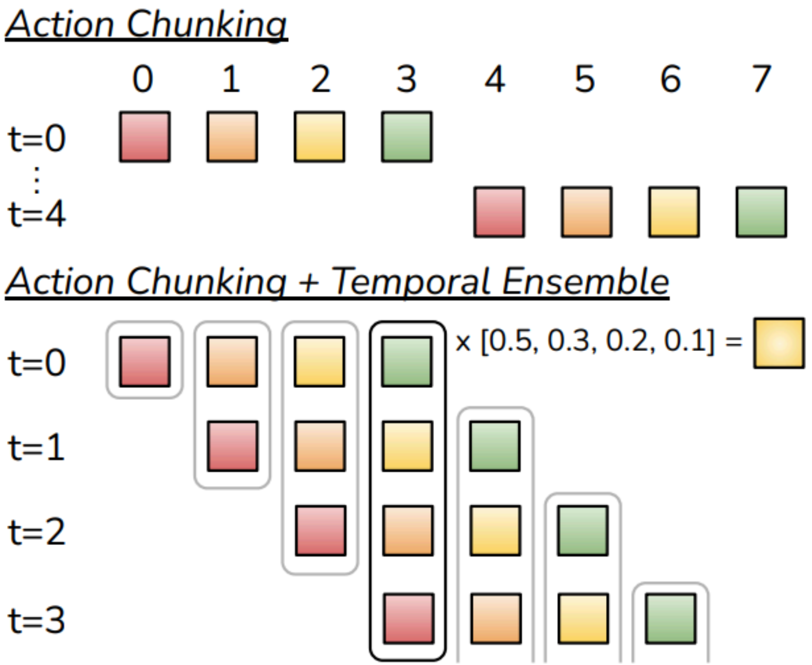


Fig. 5: We employ both Action Chunking and Temporal Ensembling when applying actions, instead of interleaving observing and executing.

Temporal Ensemble

- Creates **overlapping action chunks**

Temporal Ensemble

- Creates **overlapping action chunks**

- Queries the policy at **every step** for precise and smoother motions

Temporal Ensemble

- Creates **overlapping action chunks**
- Queries the policy at **every step** for precise and smoother motions
- **Combines** predictions using a weighted average

Temporal Ensemble

- Creates **overlapping action chunks**
- Queries the policy at **every step** for precise and smoother motions
- **Combines** predictions using a weighted average
- No additional training cost, only extra inference-time computation

Temporal Ensemble

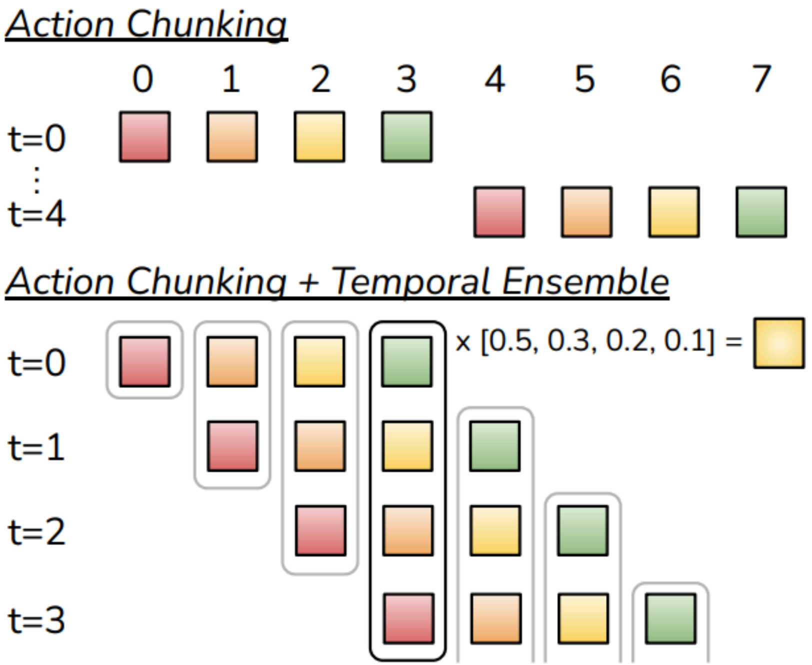


Fig. 5: We employ both Action Chunking and Temporal Ensembling when applying actions, instead of interleaving observing and executing.

Modeling Human Data

- Human demonstrations are **noisy and inconsistent**

Modeling Human Data

- Human demonstrations are **noisy and inconsistent**

- **Different** trajectories can be used for the same observation

Modeling Human Data

- Human demonstrations are **noisy and inconsistent**
- **Different** trajectories can be used for the same observation
- Human actions are more **stochastic** where precision matters less

Modeling Human Data

- Human demonstrations are **noisy and inconsistent**
- **Different** trajectories can be used for the same observation
- Human actions are more **stochastic** where precision matters less
- Policy must **focus on high precision** areas

Conditional Variational Autoencoder (CVAE)

- Train action chunking policy as a generative model

Conditional Variational Autoencoder (CVAE)

- Train action chunking policy as a generative model
- Only decoder (policy) used in deployment

Conditional Variational Autoencoder (CVAE)

- Train action chunking policy as a generative model
- Only decoder (policy) used in deployment
- Maximize log-likelihood of demonstration action chunks

Architecture

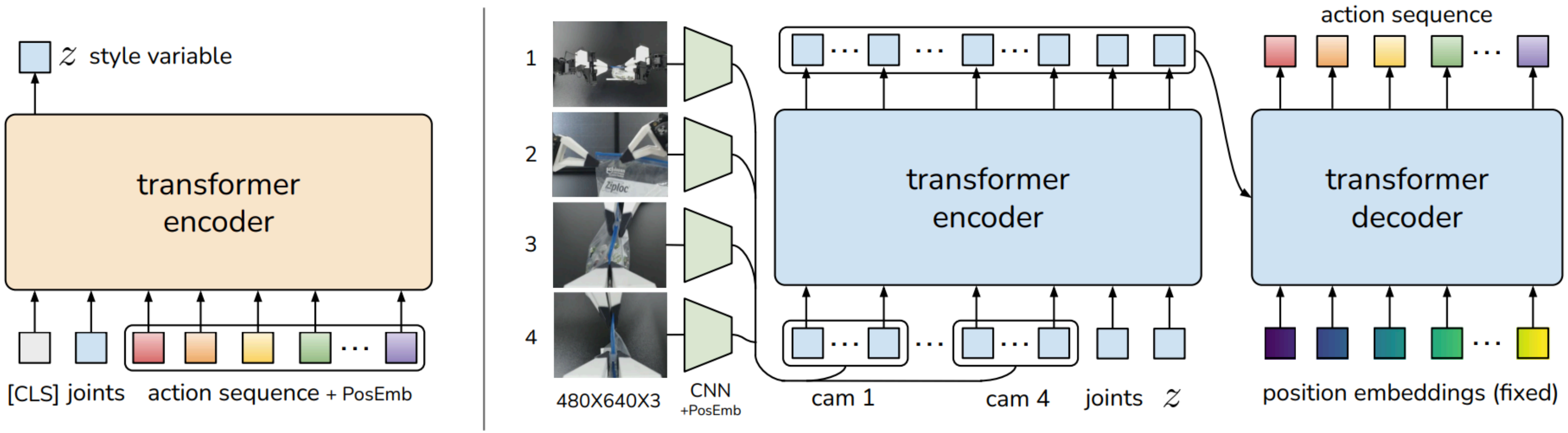


Fig. 4: *Architecture of Action Chunking with Transformers (ACT).* We train ACT as a Conditional VAE (CVAE), which has an encoder and a decoder. *Left:* The encoder of the CVAE compresses action sequence and joint observation into z, the style variable. The encoder is discarded at test time. *Right:* The decoder or policy of ACT synthesizes images from multiple viewpoints, joint positions, and z with a transformer encoder, and predicts a sequence of actions with a transformer decoder. z is simply set to the mean of the prior (i.e. zero) at test time.

- typo: synthesizes images → information

Implementation of ACT: Encoder

- CVAE encoder and decoder implemented with **transformers**

Implementation of ACT: Encoder

- CVAE encoder and decoder implemented with **transformers**

- **BERT-like transformer** encoder used

Implementation of ACT: Encoder

- CVAE encoder and decoder implemented with **transformers**
- **BERT-like transformer** encoder used
- Inputs: current joint positions and target action sequence

Implementation of ACT: Encoder

- CVAE encoder and decoder implemented with **transformers**
- **BERT-like transformer** encoder used
- Inputs: current joint positions and target action sequence
- Outputs: mean and variance of “style variable” z

Implementation of ACT: Encoder

- CVAE encoder and decoder implemented with **transformers**
- **BERT-like transformer** encoder used
- Inputs: current joint positions and target action sequence
- Outputs: mean and variance of “style variable” z
- Only used during training - z set to 0 during test

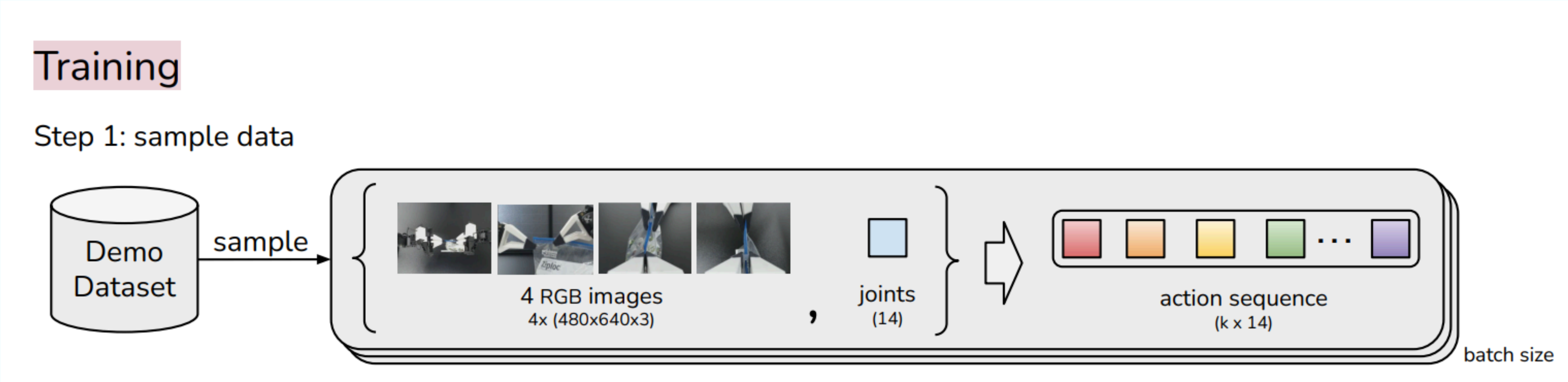
Implementation of ACT: Decoder

- Predicts next k actions
- Inputs: current observations and z
- Observations: 4 RGB images and joint positions of 2 robot arms
- ResNet18 used for image processing

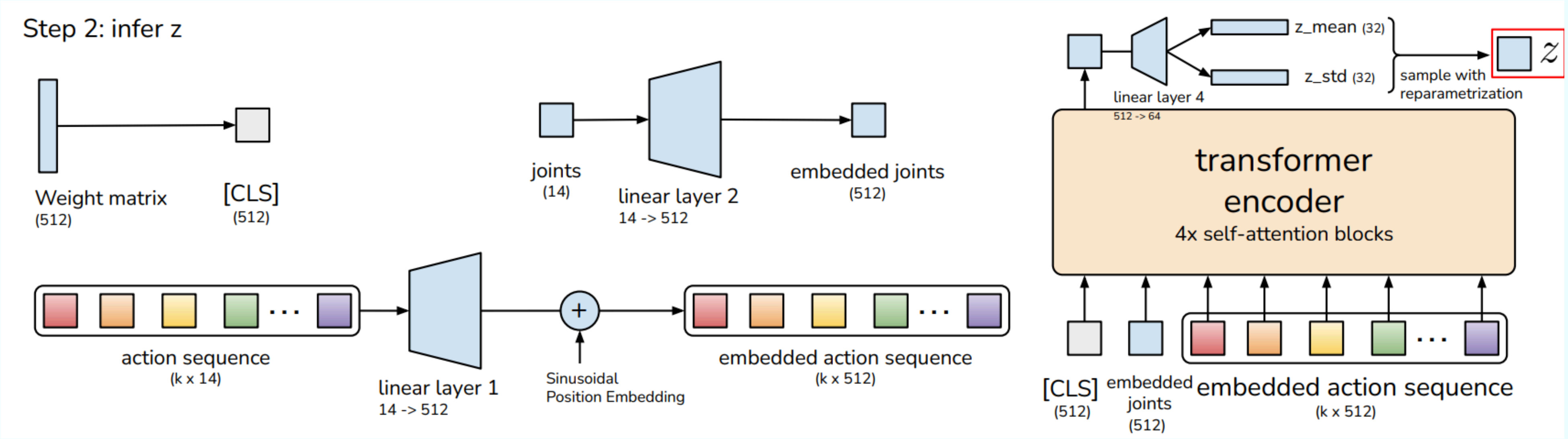
Implementation of ACT: Decoder

- Transformer encoder synthesizes information
- Transformer decoder generates action sequence
- L1 loss used for precise action sequence modeling

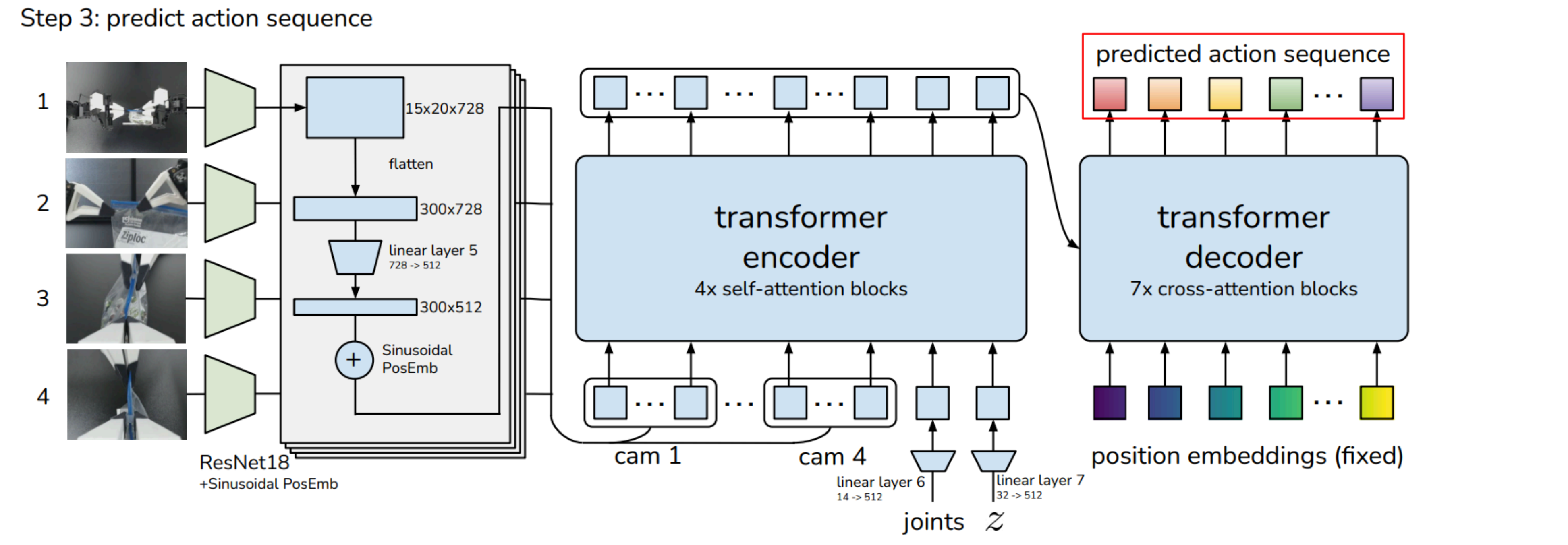
Model architecutre: Training I



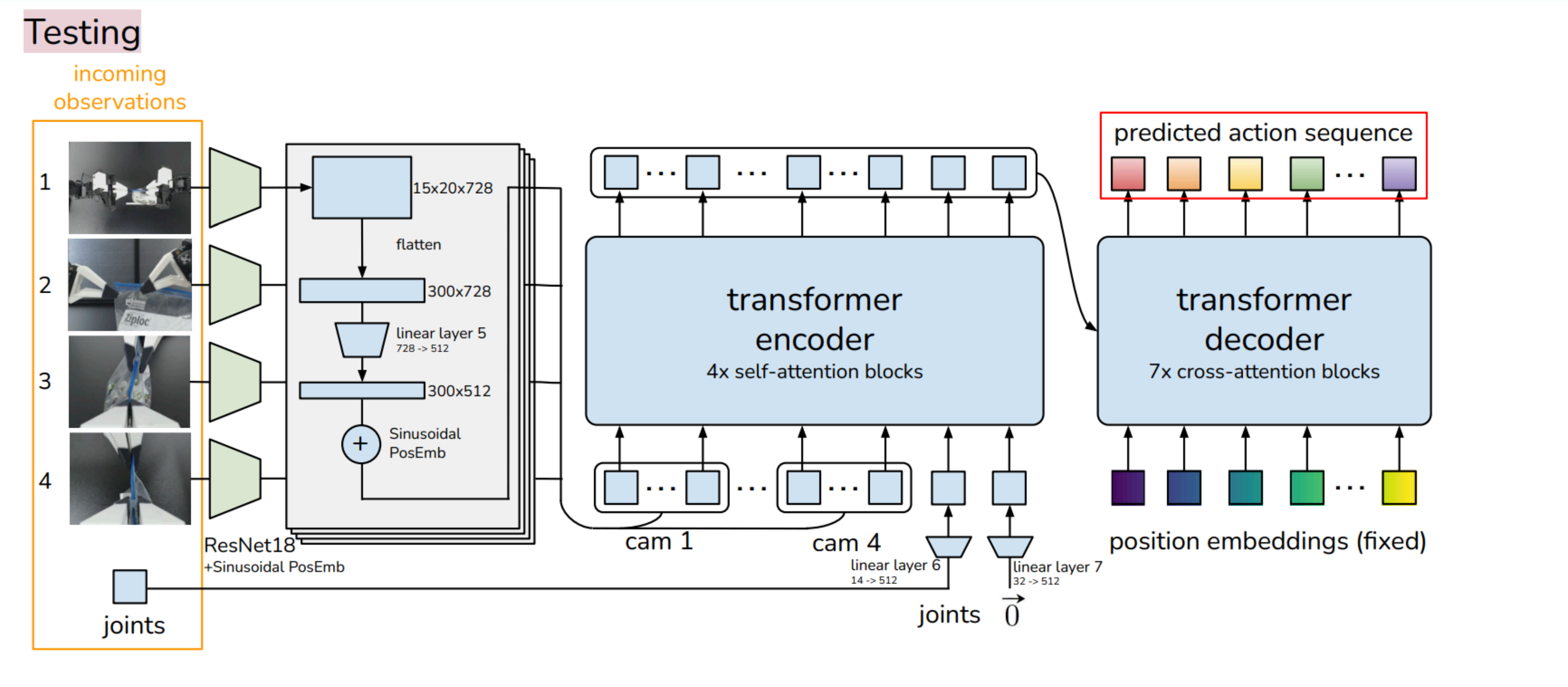
Model architecutre: Training II



Model architecutre: Training III



Model architecutre: Testing



Experiment Tasks

- Slide Ziploc: Grasp and open ziploc bag slider
- Slot Battery: Insert battery into remote controller slot
- Open Cup: Open lid of small condiment cup
- Thread Velcro: Insert velcro cable tie into loop
- Prep Tape: Cut and hang tape on box edge
- Put On Shoe: Put shoe on mannequin foot and secure velcro strap
- Transfer Cube (sim): Transfer red cube to other arm
- Bimanual Insertion (sim): Insert peg into socket in mid-air

Challenges

- Requires fine-grained bimanual control

Challenges

- Requires fine-grained bimanual control

- Perception challenges (e.g., transparency, low contrast)

Challenges

- Requires fine-grained bimanual control
- Perception challenges (e.g., transparency, low contrast)
- Random initial placement of objects

Challenges

- Requires fine-grained bimanual control
- Perception challenges (e.g., transparency, low contrast)
- Random initial placement of objects
- Need for visual feedback to correct perturbations

Challenges

- Requires fine-grained bimanual control
- Perception challenges (e.g., transparency, low contrast)
- Random initial placement of objects
- Need for visual feedback to correct perturbations
- Precise manipulation needed

Data Collection

- Collected using ALOHA teleoperation for 6 real-world tasks

Data Collection

- Collected using ALOHA teleoperation for 6 real-world tasks
- Each episode: 8-14 seconds (400-700 time steps at 50Hz)

Data Collection

- Collected using ALOHA teleoperation for 6 real-world tasks
- Each episode: 8-14 seconds (400-700 time steps at 50Hz)
- 50 demonstrations per task (100 for Thread Velcro)

Data Collection

- Collected using ALOHA teleoperation for 6 real-world tasks
- Each episode: 8-14 seconds (400-700 time steps at 50Hz)
- 50 demonstrations per task (100 for Thread Velcro)
- Total: 10-20 minutes of data per task

Data Collection

- Collected using ALOHA teleoperation for 6 real-world tasks
- Each episode: 8-14 seconds (400-700 time steps at 50Hz)
- 50 demonstrations per task (100 for Thread Velcro)
- Total: 10-20 minutes of data per task
- Scripted policy / human demonstrations for simulated tasks

Human demonstrations are stochastic

- Mid-air handover example: position varies each time

Human demonstrations are stochastic

- Mid-air handover example: position varies each time
- Policy must learn dynamic adjustments, not memorization

Experiment Comparison

- Compared ACT with four methods:
  - BC-ConvMLP: Simple baseline with convolutional network
  - BeT: Uses Transformers, no action chunking, separate visual encoder
  - RT-1: Transformer-based, predicts one action from history
  - VINN: Non-parametric, uses k-nearest neighbors

Experiment Comparison

- Compared ACT with four methods:
  - BC-ConvMLP: Simple baseline with convolutional network
  - BeT: Uses Transformers, no action chunking, separate visual encoder
  - RT-1: Transformer-based, predicts one action from history
  - VINN: Non-parametric, uses k-nearest neighbors

- ACT directly predicts continuous actions

Experiment Results

Experiment Results

- ACT outperforms all prior methods in both simulated and real tasks

Experiment Results

- ACT outperforms all prior methods in both simulated and real tasks

- Simulated tasks: ACT shows 20%-59% higher success rates

Experiment Results

- ACT outperforms all prior methods in both simulated and real tasks
- Simulated tasks: ACT shows 20%-59% higher success rates
- Real-world tasks: Slide Ziploc (88%), Slot Battery (96%)

Experiment Results

- ACT outperforms all prior methods in both simulated and real tasks
- Simulated tasks: ACT shows 20%-59% higher success rates
- Real-world tasks: Slide Ziploc (88%), Slot Battery (96%)
- ACT’s performance in Thread Velcro was lower (20%) due to precision challenges

Experiment Results

	Cube Transfer (sim)			Bimanual Insertion (sim)			Slide Ziploc (real)			Slot Battery (real)		
	Touched	Lifted	Transfer	Grasp	Contact	Insert	Grasp	Pinch	Open	Grasp	Place	Insert
BC-ConvMLP	34 | 3	17 | 1	1 | 0	5 | 0	1 | 0	1 | 0	0	0	0	0	0	0
BeT	60 | 16	51 | 13	27 | 1	21 | 0	4 | 0	3 | 0	8	0	0	4	0	0
RT-1	44 | 4	33 | 2	2 | 0	2 | 0	0 | 0	1 | 0	4	0	0	4	0	0
VINN	13 | 17	9 | 11	3 | 0	6 | 0	1 | 0	1 | 0	28	0	0	20	0	0
ACT (Ours)	**97 | 82**	**90 | 60**	**86 | 50**	**93 | 76**	**90 | 66**	**32 | 20**	**92**	**96**	**88**	**100**	**100**	**96**

TABLE I: Success rate (%) for 2 simulated and 2 real-world tasks, comparing our method with 4 baselines. For the two simulated tasks, we report [training with scripted data | training with human data], with 3 seeds and 50 policy evaluations each. For the real-world tasks, we report training with human data, with 1 seed and 25 evaluations. Overall, ACT significantly outperforms previous methods.

	Open Cup (real)			Thread Velcro (real)			Prep Tape (real)				Put On Shoe (real)			
	Tip Over	Grasp	Open Lid	Lift	Grasp	Insert	Grasp	Cut	Handover	Hang	Lift	Insert	Support	Secure
BeT	12	0	0	24	0	0	8	0	0	0	12	0	0	0
ACT (Ours)	**100**	**96**	**84**	**92**	**40**	**20**	**96**	**92**	**72**	**64**	**100**	**92**	**92**	**92**

TABLE II: Success rate (%) for the remaining 3 real-world tasks. We only compare with the best performing baseline BeT.

Ablation: Action Chunking and Temporal Ensembling

- Action chunking reduces compounding errors by dividing sequences into chunks
- Performance improves with increasing chunk size, best at $k = 100$
- Temporal ensembling further improves performance by averaging predictions

Ablation: Training with CVAE

- CVAE models noisy human demonstrations
- Essential for learning from human data, removing CVAE objective significantly drops performance

Ablation: Training with CVAE

- CVAE models noisy human demonstrations
- Essential for learning from human data, removing CVAE objective significantly drops performance
- Human data success rate drops from 35.3% to 2% without CVAE

Ablation: Is High-Frequency Necessary?

- Human shows higher performance at 50Hz compared to 5Hz

Ablation: Is High-Frequency Necessary?

- Human shows higher performance at 50Hz compared to 5Hz

- Tasks: threading zip cable tie and unstacking plastic cups

Ablation: Is High-Frequency Necessary?

- Human shows higher performance at 50Hz compared to 5Hz
- Tasks: threading zip cable tie and unstacking plastic cups
- 50Hz: faster and more accurate task completion

Ablation: Is High-Frequency Necessary?

- Human shows higher performance at 50Hz compared to 5Hz
- Tasks: threading zip cable tie and unstacking plastic cups
- 50Hz: faster and more accurate task completion
- 50Hz reduces teleoperation time by 62% compared to 5Hz

Ablation graphs

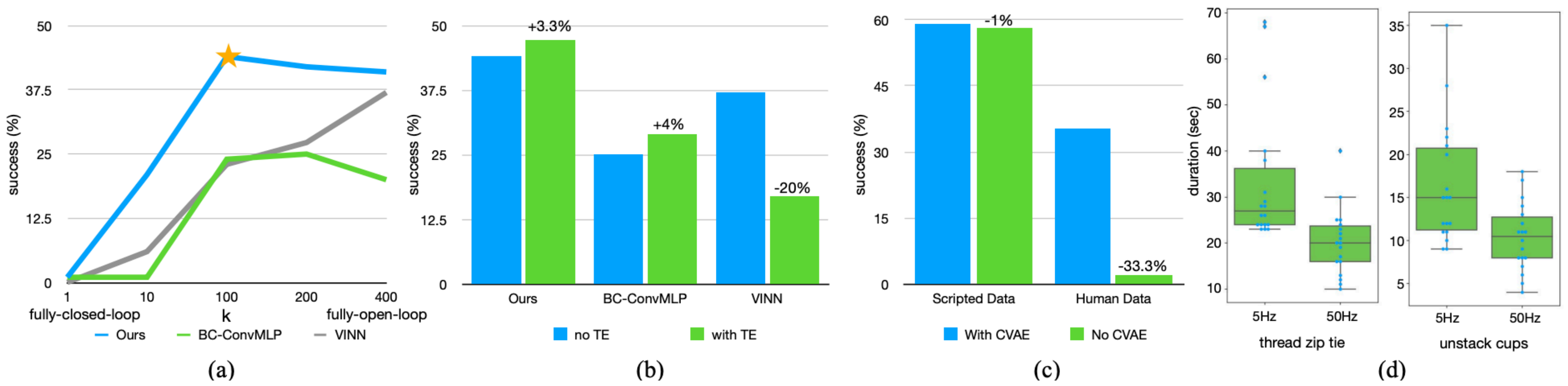


Fig. 8: *(a)* We augment two baselines with action chunking, with different values of chunk size k on the x-axis, and success rate on the y-axis. Both methods significantly benefit from action chunking, suggesting that it is a generally useful technique. *(b)* Temporal Ensemble (TE) improves our method and *BC-ConvMLP*, while hurting *VINN*. *(c)* We compare with and without the CVAE training, showing that it is crucial when learning from human data. *(d)* We plot the distribution of task completion time in our user study, where we task participants to perform two tasks, at 5Hz or 50Hz teleoperation frequency. Lowering the frequency results in a 62% slowdown in completion time.

Limitations and Conclusion

- Presented a low-cost system for fine manipulation
- Components: ALOHA teleoperation system and ACT imitation learning algorithm
- Enables learning fine manipulation skills in real-world
- Examples: Opening a translucent condiment cup, slotting a battery (80-90% success rate, 10 min demonstrations)

- Limitations: Tasks beyond current capabilities, e.g., buttoning a dress shirt
- Hope: Important step and accessible resource for advancing fine-grained robotic manipulation