Human-to-Robot Imitation in the Wild

Shikhar Bahl, Abhinav Gupta⋆, Deepak Pathak⋆

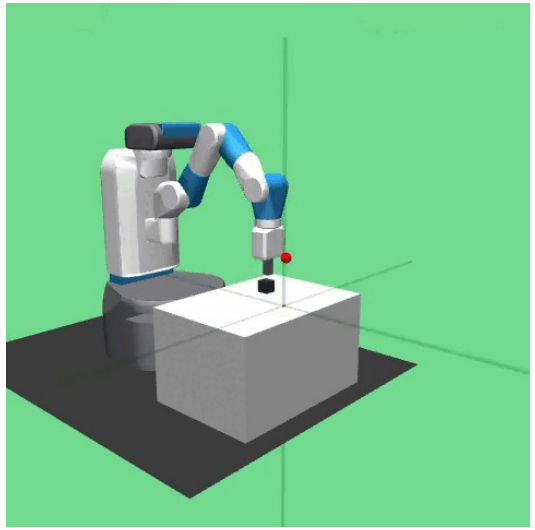
Presenters: Yue Yang, Ce Zhang

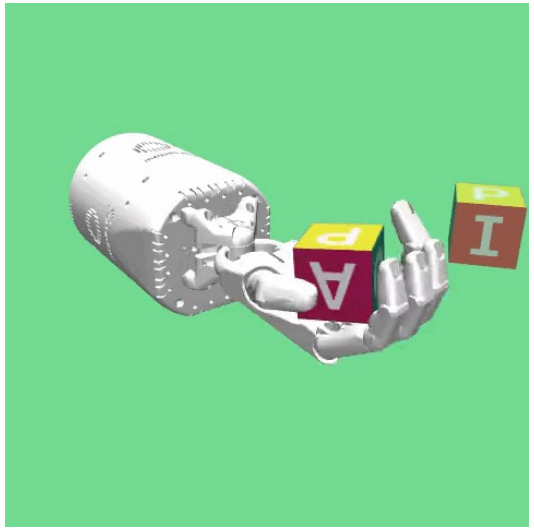
Outlines

- Introduction
- Method
- Experiments
- Conclusion
- Limitations

Introduction - Motivation

- **Motivation**: Current robot manipulation only have in simulation or table-top experiments in the lab. (Figures from [1])

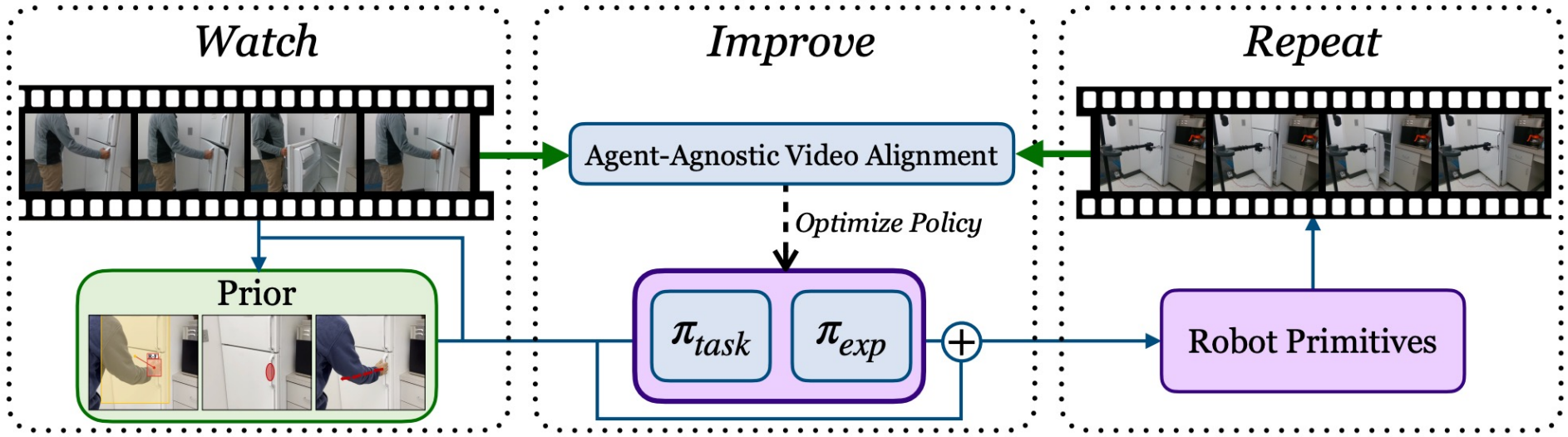




[1]. Brockman, Greg, et al. "Openai gym." *arXiv preprint arXiv:1606.01540* (2016).

Introduction - Motivation

- **WHIRL**: In-the-Wild Human Imitating Robot Learning.
  - 2-Step efficient real-world robot learning framework:
    1. **Learn a prior**: Use human videos to extract a ***prior*** over the intent of the demonstrator, and use this to initialize our agent's policy.
    2. **Improve the prior**: Use an efficient real-world policy learning scheme to ***improve*** over the human prior using interactions.



Introduction - Limitations of Current Methods

- **Limitations of current methods**:
  - Reinforcement Learning (RL):
    - ***Unsafe***: While the robot interacts with the physical world, it could hurt human or itself.
    - ***Reward function needs manually design***: For different tasks/environment/robot type, the expert needs to design different reward functions to make RL work.
    - ***Sample inefficient***: In most cases, expert can only design sparse (i.e., only a meagre amount of states in the state space can return a feedback signal) reward functions for many physical world tasks. These sparse rewards make it take a long time for the robot to learn.
  - WHIRL:
    - They claim WHIRL is safe.
    - No reward function needed.

Introduction - Limitations of Current Methods

- **Limitations of current methods**:
  - Imitation Learning (IL):
    - ***Data hungry***: IL methods require a large amount of robot demonstrations provided by an oracle. However, these demonstrations could be expensive to obtain in real world.
    - **Poor Generalization**: Hard to generalize to new tasks or different robots.
  - WHIRL
    - Videos of human performing tasks are much easier to obtain.
    - Can easily learn different tasks. For a new robot, just change to a new human-robot mapping.

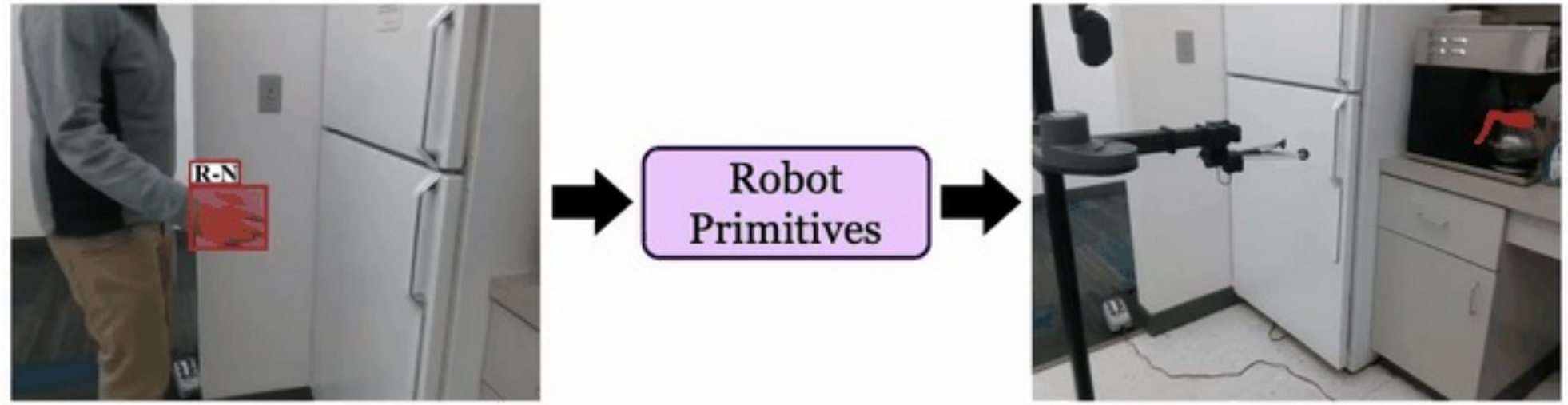
Method - Overview

- Human Priors. **WHIRL** extracts a prior over the intent of the human demonstrator, using it to initialize the agent's policy.

- Policy Learning via Interaction: an efficient real-world policy learning scheme that improves using interactions.

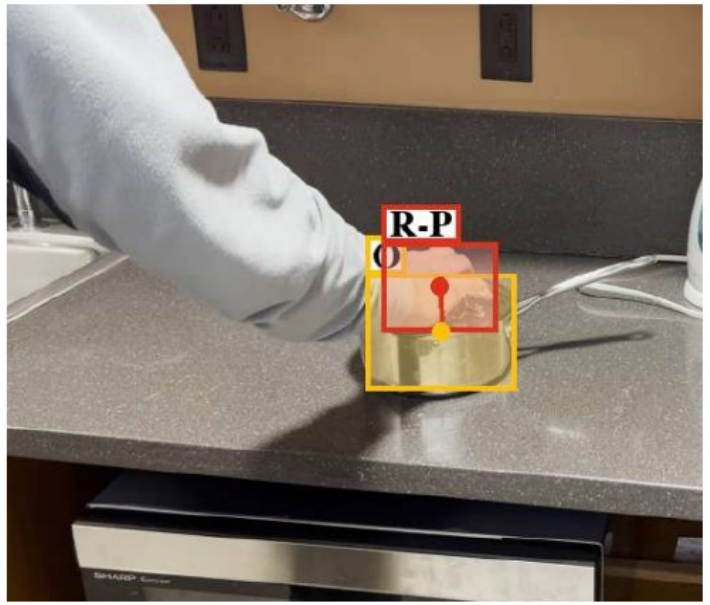
Method – Human Priors

- Human Priors. WHIRL extracts a prior over the intent of the human demonstrator, using it to initialize the agent's policy.
  - Extracting Human Priors
  - Converting Human Priors to Robot Priors



Method – Human Priors

- Extracting Human Priors.
  - Extracting Hand Information
  - Extracting Interaction Information
- Hand Information
  - h_t: x_t, y_t, z_t (3D hand location)
    - 100DOH (100 Days of Hands) detection model
    - depth camera
  - θ_t : yaw, pitch, roll (rotation of the wrist)
    - MANO (hand Model with Articulated and Non-rigid defOrmations)



100DOH detection model

- location (boxes)
- side (left/right)
- contact state (P: portable object)
- what object each hand is in contact with.

Method – Human Priors

- Extracting Human Priors.
  - Extracting Hand Information
  - Extracting Interaction Information
- Interatcion Information
  - $h_{interact}$, h_{mid}, h_{end}: hand location when the interaction starts, in process, or ends
    - run 100DOH detection model on every frame
  - $O_{1:T}$: commands to close or open the hand. T is the length of the video.
    - run 100DOH detection model on every frame

100DOH detection model

- location (boxes)
- side (left/right)
- contact state (N: no contact. P: portable object)
- what object each hand is in contact with.

Method – Human Priors to Robot Priors

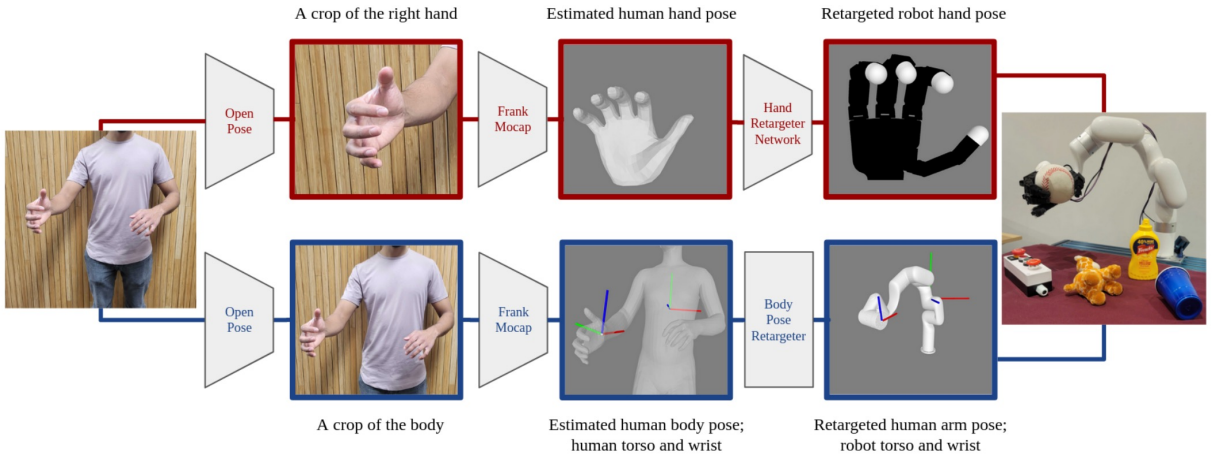
- Given Human Video V_k, we use a heuristic function to convert Human Priors to Robot Priors.

$f_{map}(h_{interact}, h_{mid}, h_{end}, \theta_{hand}, O_{1:T}) = w_{interact}, w_{mid}, w_{end}, \theta_{YPH}, g_{1:T}$

Robot Priors Ψ_k

  - $w_{interact}, w_{mid}, w_{end}$: gripper waypoints in robot's coordinate
  - θ_{YPH}: robot wrist rotation (yaw-pitch-roll format)
  - $g_{1:T}$: robot gripper open/close parameters.

An example mapping function (Neural Network, not heuristic) [2]:



[2]. Sivakumar, Aravind, Kenneth Shaw, and Deepak Pathak. "Robotic telekinesis: Learning a robotic hand imitator by watching humans on YouTube." *arXiv preprint arXiv:2202.10448* (2022).

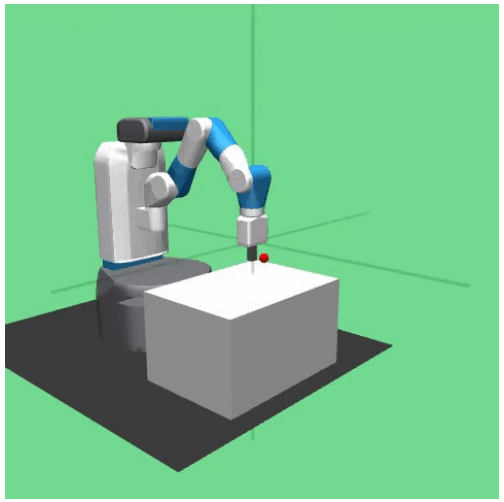
Method – Human Priors to Robot Priors

$f_{map}(h_{interact}, h_{mid}, h_{end}, \theta_{hand}, O_{1:T}) = w_{interact}, w_{mid}, w_{end}, \theta_{YPH}, g_{1:T}$

- $w_{interact}, w_{mid}, w_{end}$: gripper waypoints in robot's coordinate
- θ_{YPH}: robot wrist rotation (yaw-pitch-roll format)
- $g_{1:T}$: robot gripper open/close parameters.

Robot Priors Ψ_k

- The robot can directly execute Ψ_k to finish the task demonstrated in human video V_k.
- We use Ψ_k as an initial policy.



Reaching some waypoints [1]

Method – Important Annotations

- Ψ: The **robot prior** extracted from Step 1.
- Ψ_k: The robot prior obtained from the $\boldsymbol{k^{th}}$ video (#video = K).
- $\Delta\Psi$: The **residual** we want to learn to improve the robot prior.
- $\Delta\Psi_{k,m}$: The residual obtained from the $\boldsymbol{k^{th}}$ video and $\boldsymbol{m^{th}}$ rollout (#rollouts = M).
- $R_{k,m}$: The video taken for the $\boldsymbol{m^{th}}$ rollout based on Ψ_k + $\Delta\Psi_{k,m}$.

Method – Improve Robot Priors via Interaction

- The “Robot Priors” converted from “Human Priors” can only be a rough guideline. Directly execute the robot prior can **NOT** lead to success owing to:
  - Differences in morphologies between human and robot hands.
  - Inaccuracies in detections.
  - Errors in the calibration process.
- Therefore, we need to improve the robot prior, Ψ, by adding a residual, $\Delta\Psi$, to the prior.
- After we learn an ideal residual, $\Delta\Psi$, we could directly execute the $\Psi + \Delta\Psi$.

Method – Improve Robot Priors via Interaction

- Now, the **question** is: Given a robot prior Ψ and the corresponding video V, how to learn a $\Delta\Psi$?
- The **insights**: If we use Ψ plus an initial random $\Delta\Psi$ and then rollout, some rollouts will be "better" than some others (measured by some metrics). With the existence of the "better", we could do some optimization and will get an ideal (i.e., local/global optimal) $\Delta\Psi$ after reaching convergence.
- So the challenges are:
  - How to build the optimization mechanism -> Inspired by Cross Entropy Method (CEM)
  - How to design the proper metrics -> metric-1: human-to-robot video alignment. metric-2: maximal frame distance.

Method – Optimization Procedure

- **Representation** of $\Delta\Psi$: Conditional Variational Auto-Encoder (CVAE) -> can represent multi-modality ($\Delta\Psi$ is diverse even for one task because there're many trajectories to perform one task)
  - Condition: current video V.
  - Input and output: $\Delta\Psi$
  - Inference: Given a latent vector z and the video V, can output a $\Delta\Psi$.
- **Steps**
  - **Step 1**: For each rollout, we can infer one $\Delta\Psi$ with a random z based on the current CVAE. Then for one task, we can get M * K rollouts based on Ψ + the inferred $\Delta\Psi$: [< $\Delta\Psi_{1,1}$, $R_{1,1}$ >, < $\Delta\Psi_{1,2}$, $R_{1,2}$ >, < $\Delta\Psi_{1,3}$, $R_{1,3}$ >, …, < $\Delta\Psi_{2,1}$, $R_{2,1}$ >, < $\Delta\Psi_{2,2}$, $R_{2,2}$ >, < $\Delta\Psi_{2,3}$, $R_{2,3}$ >, …]
  - **Step 2**: Filter elite examples, $\{\Delta\Psi\}_{i=1}^{M*K*r}$, with some ratio r (e.g., 10%) via some metrics. Use these elite examples to train the CVAE.
  - Keep Step 1 and Step 2 until convergency condition is reached (my guess is: most rollouts get very similar score from the metrics).

Algorithm 1 Training Procedure for WHIRL

Require: Task videos: $V_{1:K}$, f_{map}: video to robot actions function, prior task and exploration policies: π, π_{exp}. Video-level (Φ) and frame-level (Φ_f) agent agnostic representation. M real world interactions per task.

while not converged **do**

Step 1

for $k = 1...K$ **do**
  $\Psi_k = f_{\text{map}}(V_k)$
  for $m = 1...M$ **do**
    Sample $\Delta\Psi_{k,m} = \pi_{\text{exp}}(V_k, \Psi_k)$ (prob: p)
    Sample $\Delta\Psi_{k,m} = \pi(V_k, \Psi_k)$ (prob: $1 - p$)
    $a_{j,m} = \Psi_k + \Delta\Psi_{k,m}$
    Execute $a_{k,m}$, collect video: $R_{k,m}$
  end for
end for

Step 2

for $j = 1...K$ **do**
  rank $\texttt{Cost}(\Phi(R_{k,m}), \Phi(V_k))$ for every m
  pick E = {elite examples}
  fit $\pi(.)$ as a VAE to $\Psi_{k,m} \in E$
  pick E_{exp} = $\{\Phi_f(R_{k,m})$ with highest "change"$\}$
  fit $\pi_{\text{exp}}(.)$ as a VAE to $\Psi_{k,m} \in E_{\text{exp}}$
end for

end while

Method – Two Metrics

- Metric is used to measure how good one thing is.
- Here, there are 2 things we aim for:
  1. Task Completion: The obvious one – we hope the robot could succeed to complete the task.
  2. More Exploration: It's possible that all the rollouts cannot complete the task. For instance, for the drawer opening task, it's possible that the gripper never reaches the right position if we rollout based on the robot prior. So we need some exploration of the environment.
- Therefore, we have
  - 2 CVAE to infer the $\Delta\Psi$. One aims to complete the task. Another one aims to make the collected rollout more diverse (i.e., increase the probability of containing better rollouts).
  - 2 metrics.

Method – Metric 1: Task Completion

- One important **assumption**: the effect that the agent had on the environment is more important than how the agent moved, since that can vary with different morphologies.
- This means if the environment, the robot interacts with, changes the same as the environment, the human interacts with, then the robot probably succeed to perform the task.
- So the metric 1 is invented to compare the difference between 2 videos with human and robot inpainted respectively: $|\Phi(V) - \Phi(R_k)||_2$

(a) Robot Image (b) Robot Inpainted (c) Human Image (d) Human Inpainted image

Method – Metric 2: More Exploration

- The metric to measure more exploration is straightforward: Aim to maximize the change that the agent causes in the environment.
- The actions are close to the prior, so it is likely that any changes caused by the agent in the environment will be meaningful and not destructive.

$$c_k = \max_{i,j} ||\Phi_f(R_{k,i}) - \Phi_f(R_{k,j})||_2$$

Method – Back to the Pseudo Code

Algorithm 1 Training Procedure for WHIRL

Require: Task videos: $V_{1:K}$, f_{map}: video to robot actions function, prior task and exploration policies: π, π_{exp}. Video-level (Φ) and frame-level (Φ_f) agent agnostic representation. M real world interactions per task.

while not converged **do**

Step 1

for $k = 1...K$ **do**
  $\Psi_k = f_{\text{map}}(V_k)$
  for $m = 1...M$ **do**
    Sample $\Delta\Psi_{k,m} = \pi_{\text{exp}}(V_k, \Psi_k)$ (prob: p)
    Sample $\Delta\Psi_{k,m} = \pi(V_k, \Psi_k)$ (prob: $1 - p$)
    $a_{j,m} = \Psi_k + \Delta\Psi_{k,m}$
    Execute $a_{k,m}$, collect video: $R_{k,m}$
  end for
end for

CVAE Inference

Step 2

for $j = 1...K$ **do**
  rank $\texttt{Cost}(\Phi(R_{k,m}), \Phi(V_k))$ for every m
  pick E = {elite examples}
  fit $\pi(.)$ as a VAE to $\Psi_{k,m} \in E$
  pick E_{exp} = $\{\Phi_f(R_{k,m})$ with highest "change"$\}$
  fit $\pi_{\text{exp}}(.)$ as a VAE to $\Psi_{k,m} \in E_{\text{exp}}$
end for

Metric 1

Metric 2

end while

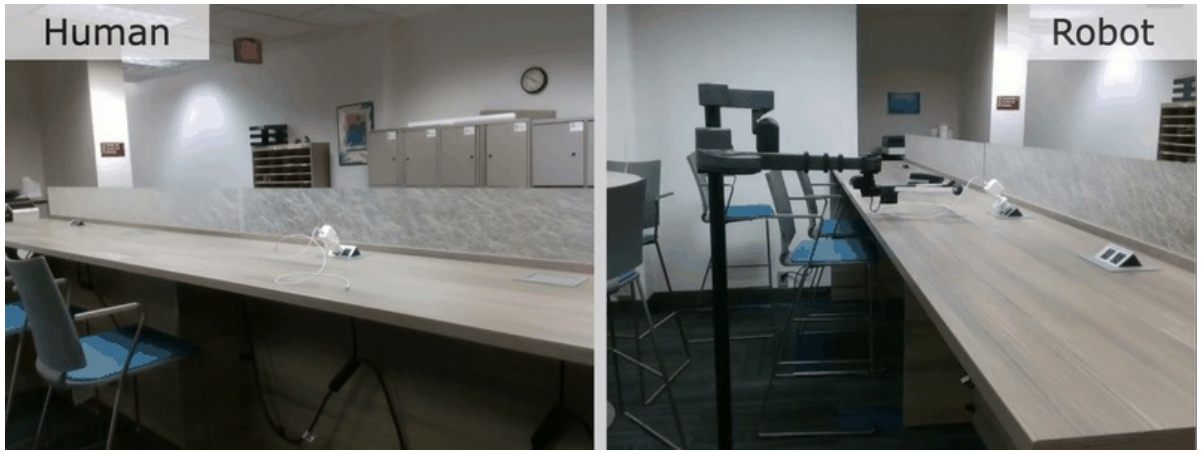
Experiments

- Hardware
  - Robot: Stretch Robot with 6 DoF arm and gripper
  - Camera: Intel Realsense D415 depth camera
- Environment and Data Collection
  - 20 every-day tasks: drawers, dishwashers, fridges, etc.
  - data collected in the wild setting

(a) Drawer (b) Dishwasher (c) Door

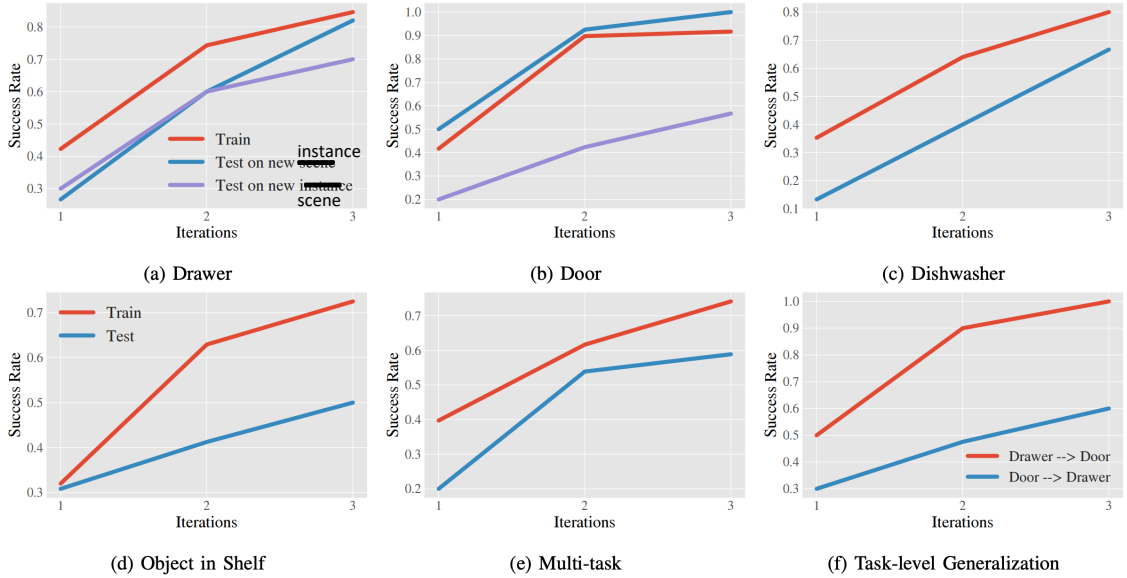
Experiments

- Robot Learning in the Wild
  - WHIRL is able to scale to a wide variety of tasks.
  - Training only takes a few hours.
  - Able to train on diverse locations and settings.



Experiments

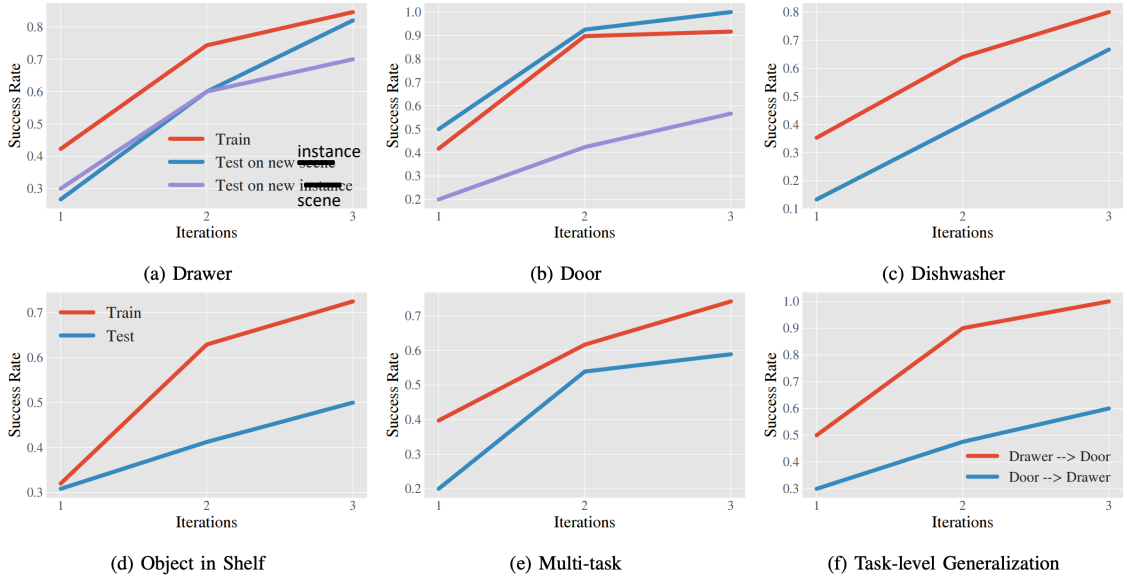
- Does policy learning iterations help?
- Can WHIRL to generalize to new instances?
- Can WHIRL to generalize to new scenes?
- How training a joint policy on different tasks would work?
- Can WHIRL to generalize to new tasks?



- a, b, c, d: train and test on 4 different tasks
- e: train and test on multiple tasks all together
- f: train on one task, test on another task

Experiments

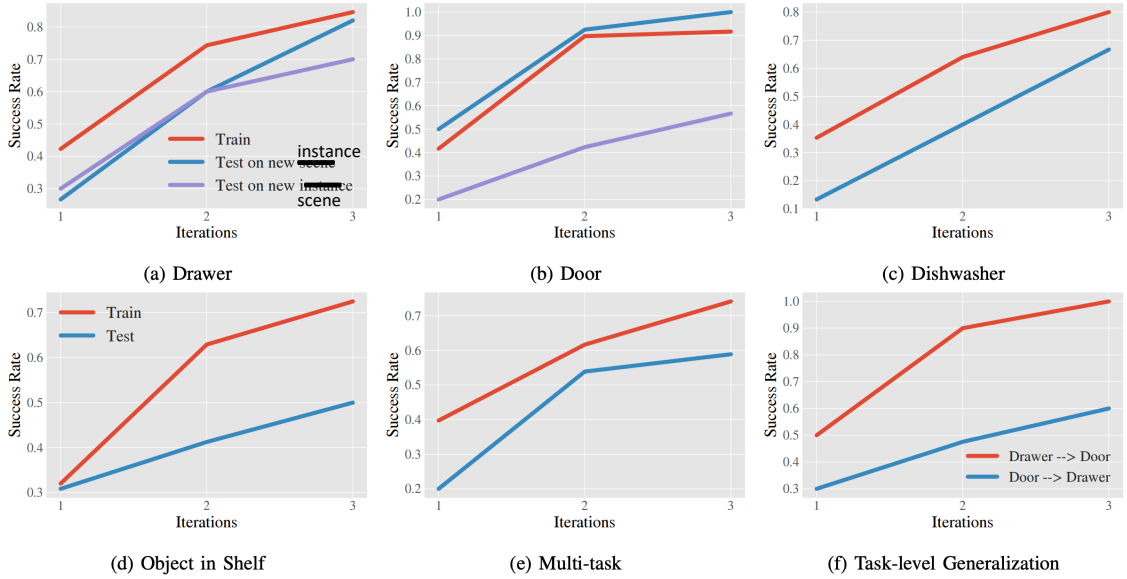
- Does policy learning iterations help?
  - Yes. Almost all curves are going up.
- Can WHIRL to generalize to new instances?
  - New instance: same task, same camera position but different object
  - Compare blue curves with red curves in a, b, c, d.
  - Depend on tasks.



- a, b, c, d: train and test on 4 different tasks
- e: train and test on multiple tasks all together
- f: train on one task, test on another task

Experiments

- Can WHIRL to generalize to new scenes?
  - New scene: same task, different camera position and different object
  - Compare purple curves and red curves in (a), (b).
  - good on Drawer but bad on Door, most likely due to large visual changes, as well as different geometry and calibrations



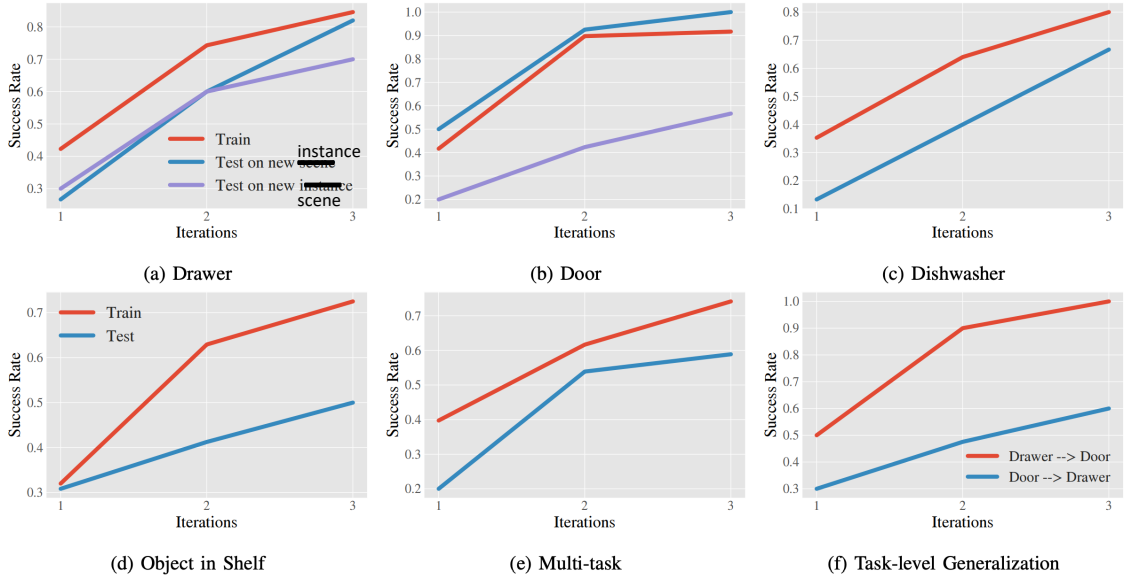
(a) Drawer (b) Door (c) Dishwasher

(d) Object in Shelf (e) Multi-task (f) Task-level Generalization

- a, b, c, d: train and test on 4 different tasks
- e: train and test on multiple tasks all together
- f: train on one task, test on another task

Experiments

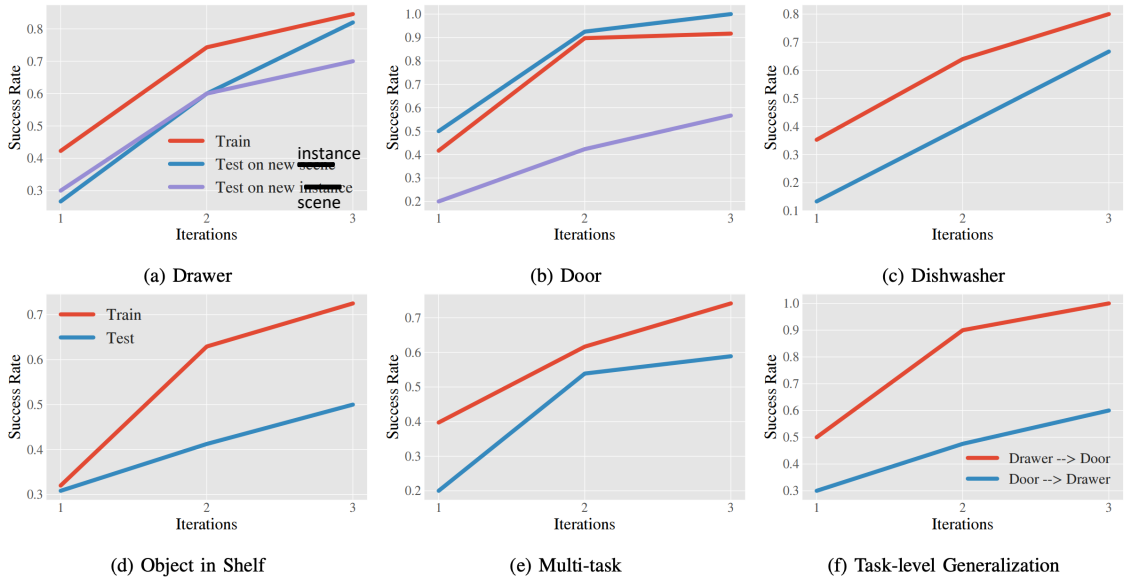
- How training a joint policy on different tasks would work?
  - Subfigure (e): train on (a) Drawer + (b) Door + (c) Dishwasher, test on (a) + (b) + (c)
  - Red curve in (e) is almost always lower than any or red curve in (a), (b) or (c), same for blue curve.
  - Multi-task training is worse than single task training.



- a, b, c, d: train and test on 4 different tasks
- e: train and test on multiple tasks all together
- f: train on one task, test on another task

Experiments

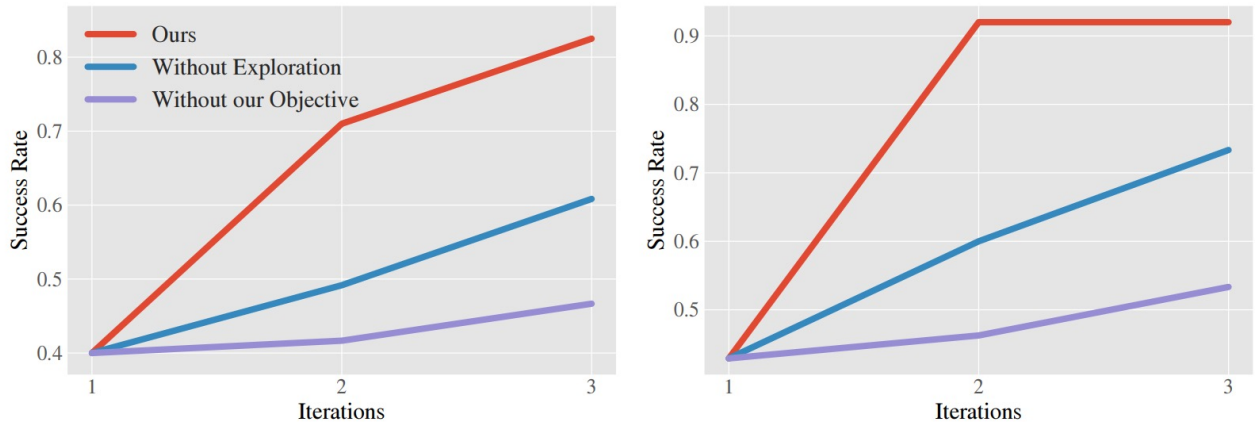
- Can WHIRL to generalize to new tasks?
  - Subfigure (f)
    - Red: train on Drawer, test on door
    - Blue: train on door, test on drawer
  - Some degree of task-level generalization.



- a, b, c, d: train and test on 4 different tasks
- e: train and test on multiple tasks all together
- f: train on one task, test on another task

Experiments - Ablations

- Policy Learning Iteration
- Exploration Policy
- Agent Agnostic Cost Function



(a) Drawer

(b) Door

Experiments - SOTA Comparison

- Behavior Cloning
- Offline RL: 4 variants of CQL
- WHIRL outperforms all previous methods by a large margin.

	Drawer	Door
No iterative improvement:		
Behavior Cloning	0.53	0.30
Offline RL (`CQL-ours`)	0.47	0.30
Offline RL (`CQL-CycleGAN`)	0.23	0.30
Offline RL (`CQL-TCN`)	0.27	0.20
Offline RL (`CQL`)	0.33	0.13
No agent-agnotic objective:		
WHIRL (ours)	0.47	0.53
No Exploration Policy:		
WHIRL (ours)	0.60	0.73
WHIRL (ours)	**0.83**	**0.92**

Conclusion

- This paper proposes WHIRL, an **efficient** real-world robot learning algorithm that can learn manipulation policies **in-the-wild** from **human videos**.
  - sampling-based policy optimization strategy
  - agent-agnostic representations
  - exploration policy
- WHIRL is able to work on 20 different tasks in the wild, and can generalize to new tasks and scenes on multiple tasks.

Limitations

- The demonstrated videos must have depth information.
  - Internet videos ususlly do not have it.
- Authors claim that their method is safe because all actions are around the robot prior. However, the robot prior is not accurate. Also, they adopted the exploration policy. Both these 2 could make the interaction process unsafe.
- The pipeline is so complicated and could be fragile. For instance, the robot prior could be extremely inaccurate when the view angle for human video is quite different from that used for robot video (Especially if you want to get online videos involved).

Thank you!