Generalization Through Hand-Eye Coordination: An Action Space for Learning Spatially-Invariant Visuomotor Control

Stanford Vision and Learning Lab

211006 한지수

Robotics 연구분야

- Manipulation / Motion Planning
  - Grasping, Pick-and-place etc.
- Perception
  - Reinforcement Learning
  - Robot Vision, Combination with Multimodal
  - Unsupervised/Self-supervised/Semi-supervised learning
  - Task: 3D Object detection / **Imitation Learning** / Human interaction etc.

Motivation

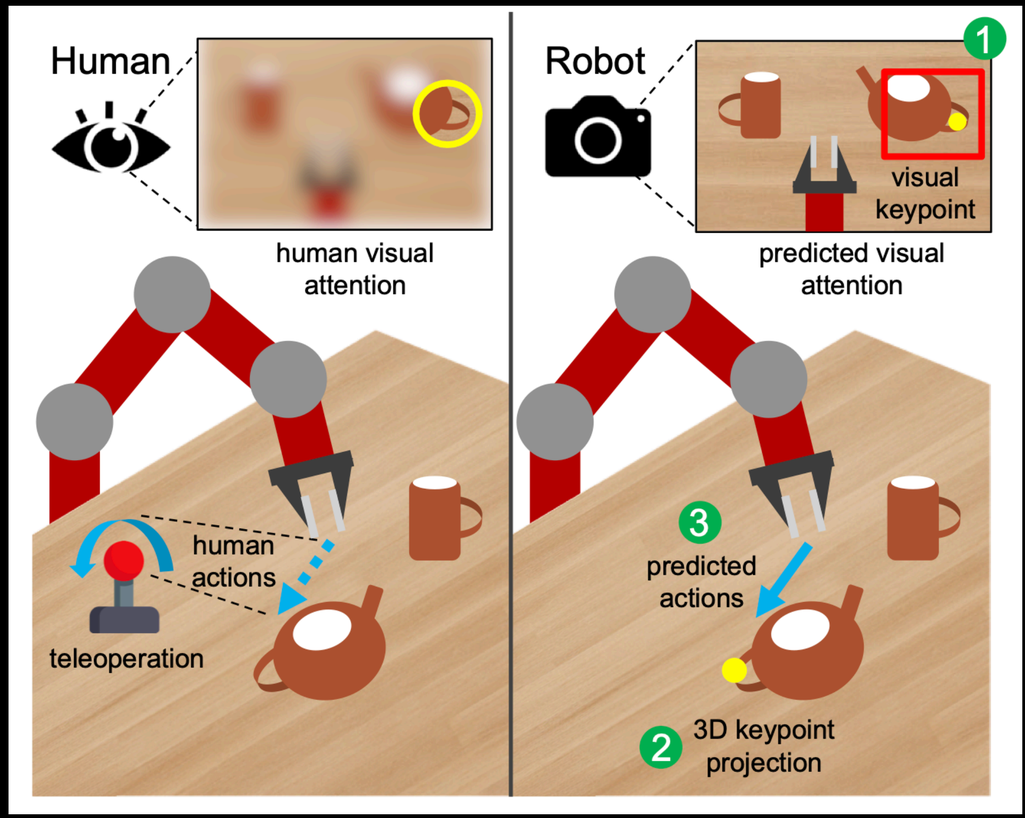
- Robot may learn to focus on spurious correlations btw the pixels and the demonstrated actions

Imitate “Human can manipulate object on Hand-eye coordinate”

➡️ Approximate human’s hand-eye coordination behaviors

➡️ Can approximate the hand-eye coordination and visual attention behaviors from a dataset of hum-controlled robot action and the corresponding image observations?

➡️ Learnable action space, Hand-eye Action Networks(HAN)



Hand-eye coordination : the cognitive ability of coordinating visual attentions and hand movements

switch visual attention ➡️ directly their movements at task-relevant objects and be invariant to the objects’ absolute spatial locations

Method

Goal: Approximate human's hand-eye coordination behaviors

1. Generate a visual Keypoint-based attention from the image observation

    ➡️ 3D Visual Attention Network

2. Generate a single 3D location to guide the robot's next action

    ➡️ Attention Switching Network

3. Generates the robot end effector actions based on the attended 3D spatial locations

    ➡️ Action Target Network

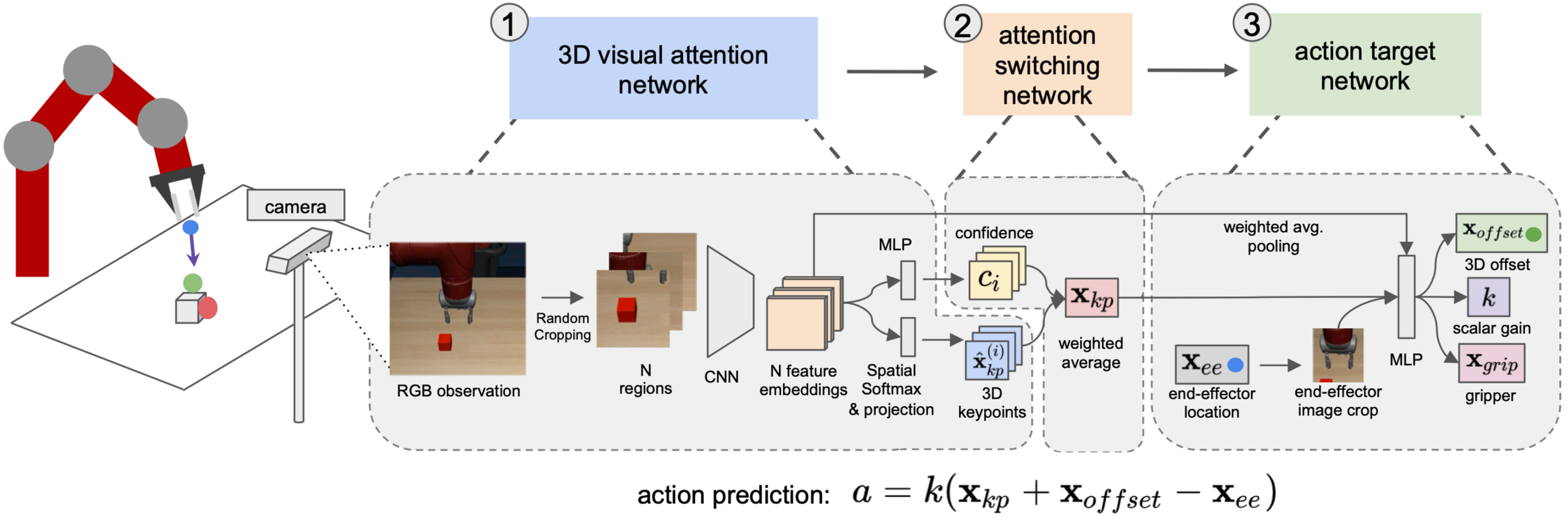
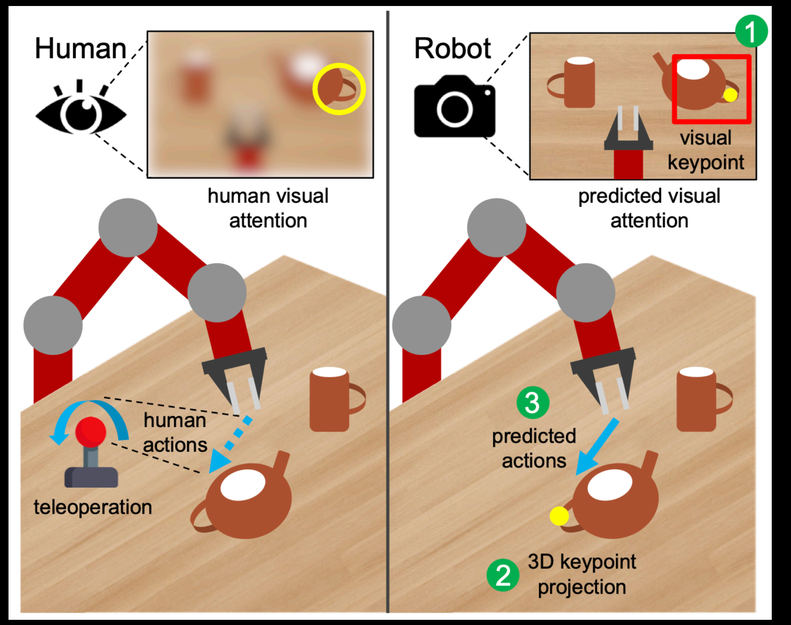


Fig. 2. **Architecture Overview.** HAN has three main components: (1) N regions are randomly sample from the input image. Then a 3D visual attention network localizes a 3D keypoint $\hat{\mathbf{x}}_{kp}$ for each region. (Sec. IV-A) (2) An attention switching network generates a single target keypoint $\mathbf{x}_{kp}$ by aggregating the candidate keypoints through confidence-weighted sum. (Sec. IV-B) (3) The final local action target is set by moving the predict keypoint $\mathbf{x}_{kp}$ by a learned offset $\mathbf{x}_{offset}$. The network predicts offset $\mathbf{x}_{offset}$, control gain k, and the binary gripper open/close command $\mathbf{x}_{grip}$. The final output action a is then calculated through the function on the bottom (Sec. IV-C).

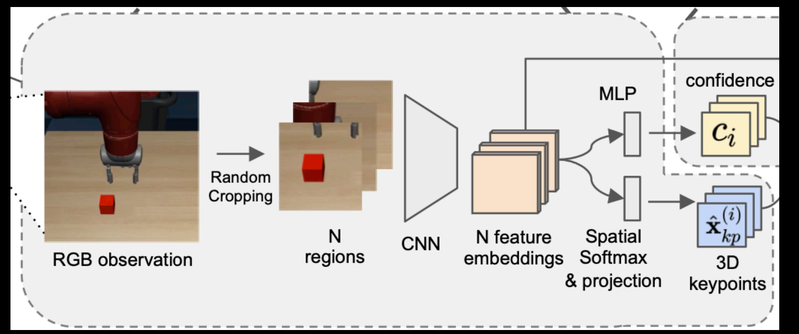
3D Visual Attention Network

- Generate a visual Keypoint-based attention from the image observation

- Input: RGB image
- Output: set of 3D key points relative to robot's base coordinate frame



1. 2D region proposal for coarse-level attention
    - Select the RoI that are likely to contain task-relevant objects (RPN in FasterRCNN)
    - Input: RGB Image => Output: N feature embeddings
2. 3D Keypoint Detection for refined attention
    - 3D keypoint detector to extract a 3D keypoint from each 2D generated by the RPN layer
    - Input: N feature embeddings => Output: 3D key points (ideal 3D keypoint candidates)



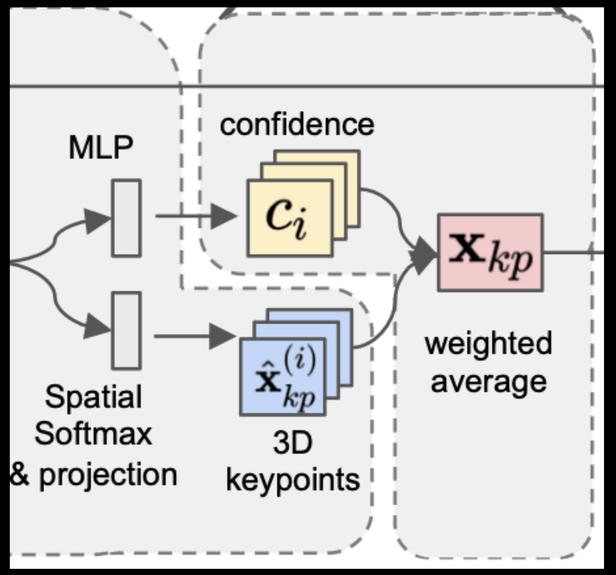
$$\{(u_i, v_i, d_i)\}_{i=1}^{N} \longrightarrow \widehat{x_{kp}} = \{(x_i, y_i, z_i)\}_{i=1}^{N}$$

2D region — 3D candidate keypoint

Attention Switching Network

- Generate a single 3D location to guide the robot's next action

- Input: set of 3D key points $\hat{x}_{kp}$ relative to robot's base coordinate frame
- Output: a single 3D location x_{kp}



- Take a weighted average of the candidates
  - c_i: normalized confidence of each candidate keypoint
    - Weights computed by passing the convolutional feature maps for each region through a shallow multilayer perceptron(MLP) and softmax normalization

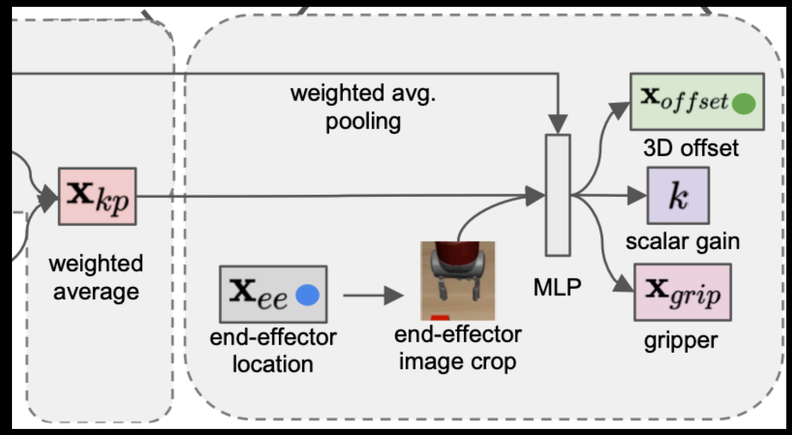
$$\hat{x}_{kp} = \{(x_i, y_i, z_i)\}_{i=1}^{N} \longrightarrow x_{kp} = \sum_{i=1}^{N} c_i \hat{x}_{kp}^{(i)}$$

3D candidate keypoint

3D keypoint

Action Target Network

- Generates the robot end effector actions based on the attended 3D spatial locations



- Input: a single 3D location x_{kp}
- Output: Local end effector actions $a = \Delta(x, y, z)$

1. Mapping
   - Fit the highly nonlinear relationship btw the target and the end effector (check noises and suboptimal actions)
   - 3D spatial offset vector x_{offset} , action target relative to the end effector position x_{ee}

     $$x_{target} = x_{kp} + x_{offset} - x_{ee}$$

2. Generate the local end effector action (use MLP predictions)
   - Flexible function approximator MLP: Network that simply learns to ignore the target keypoint and use other input information

     $$a = k(x_{kp} + x_{offset} - x_{ee})$$

- Loss function
  - L2 loss + cosine similarity loss: $L = \| a_t - a_t^* \|^2 + \lambda arccos(\frac{a_t^T a_t^*}{\| a_t \| \| a_t^* \|})$

- Results

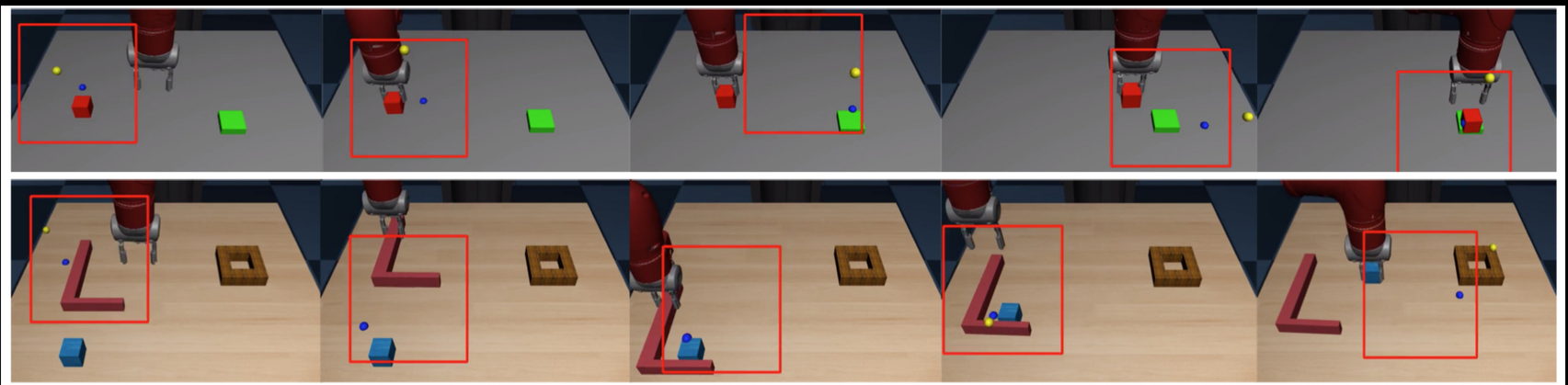
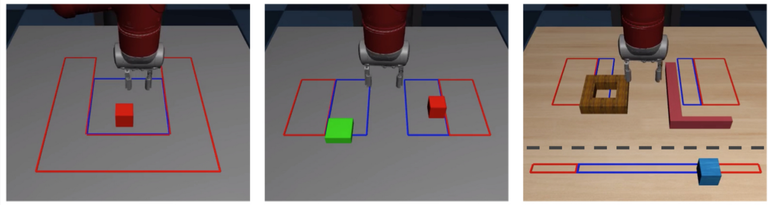


Fig. 4. Key snapshots along the rollout of the *Stacking* (top) and *Tool-using* (bottom) domains, respectively. The blue spheres are the predicted keypoints. The yellow sphere indicates the location of the action target generated by the action target network (Sec. IV-C). Some of these action targets can be occluded by objects in the scene. The red boxes are the 2D bounding box with highest confidence scores (Sec. IV-B).



(a) Lifting (b) Stacking (c) Tool-using

Fig. 3. Initial states for the three manipulation tasks we evaluate on. During data collection, objects are initialized in the *interpolation* regions, which are delineated by the red lines. We evaluate all methods with tasks that are initialized with both interpolation and *extrapolation* regions, which are delineated by red lines. The dashed line in *Tool-using* environment indicates the boundary beyond which objects are deemed unreachable by the robot, which requires the assistance of the tool.

TABLE III

QUANTITATIVE EVALUATION IN THE *Tool Using* ENVIRONMENT

Demo type	100-Expert		200-Expert	
Eval region	Int.	Ext.	Int.	Ext.
BC-states	0.13	0.03	0.23	0.0
BC-image[9]	0.3	0.03	0.33	0.07
HAN(mlp-ATN)	0.53	0.27	0.87	0.33
HAN(no-ROI)	0.0	0.0	0.0	0.0
HAN	**0.73**	**0.43**	**0.9**	**0.6**