Economical Precise Manipulation and Auto Eye-Hand Coordination with Binocular Visual Reinforcement Learning

Yiwen Chen[1,*], Sheng Guo[1,*], Zhiyang Liu[1], Lei Zhou[1],
Zhengshen Zhang[1], Ruiteng Zhao[1], Marcelo H. Ang, Jr.[1]
[1] Advanced Robotics Center, National University of Singapore

***Abstract*—Precision robotic manipulation tasks (insertion, screwing, precise pick, precise place) are required in many scenarios. Previous methods achieved good performance on such manipulation tasks. However, such methods typically require tedious calibration or expensive sensors. 3D/RGB-D cameras and torque/force sensors add to the cost of the robotic application and may not always be economical. In this work, we aim to address these issues by only 2 low-cost webcams, and minimize the reliance on manual eye-hand calibration. We propose Binocular Alignment Learning (BAL), which could automatically learn the eye-hand coordination and points alignment capabilities to solve the four tasks. Our work focuses on working with unknown eye-hand coordination and proposes different ways of performing eye-in-hand camera calibration automatically. The algorithm was trained in simulation and used a practical pipeline to achieve sim2real and test it on the real robot. Our method achieves a competitively good result with minimal cost on the four tasks. The video link of our work: https://youtu.be/gfcsGtT8IoU**

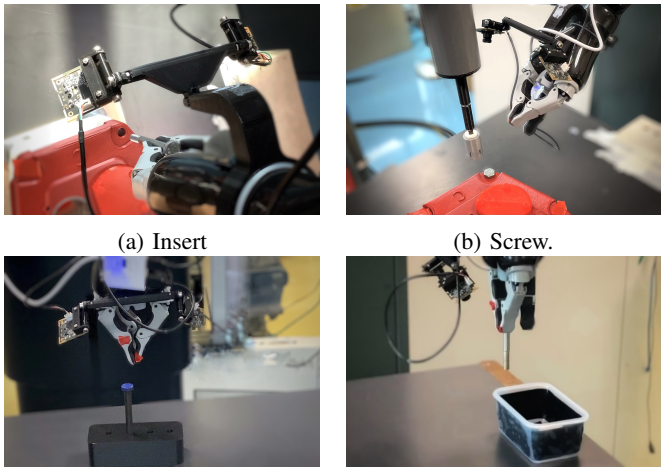
I. Introduction

Precise manipulation is a long-term challenge in robotics area, and it has been seen in a lot of scenarios like gearbox assembly [1], insertion [2] or small item pick-and-place [3]. However, such solutions generally require complex setup, such as the top-down view camera[4], [5], high-precision 3D vision camera with dedicate tuning position[6], which also lead to higher setup costs.

We also observe that the human doesn't need a high precision visual system to perform tasks, with our eyes analogous to two RGB cameras. Therefore, in this work, we argue that high precise manipulation tasks can also be done with only visual inputs.

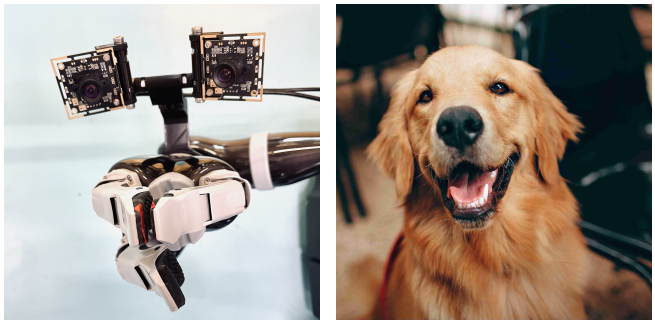
In the previous influential works, such as TossingBot [4] and 6-DOF GraspNet [7], visual inputs requires a well calibration. Eye-hand calibration helps increasing task precision. However, over-reliance on eye-hand calibration can lead to a fragile system due to camera position disturbance, camera setup offset, field-of-view changes, setup error, camera

Sheng and Yiwen are equally contributed, corresponding author is Sheng Guo (E0576004@u.nus.edu). This research is supported by the Agency for Science, Technology and Research (A*STAR), Singapore, under its AME Programmatic Funding Scheme (Project #A18A2b0046). The computational work for this article was partially performed on resources of the National Supercomputing Centre, Singapore (https://www.nscc.sg).



(a) Insert (b) Screw. (c) Pick-small-item (d) Place-small-box

Fig. 1: Task setup. KINOVA MOVO robot is used with binocular camera set on the last joint of the right arm. *For the screwing task, the right gripper holds the camera, and the left manipulator which holds screw driver performs the task.



(a) Our Eye-hand System (b) Bio. Eye-Gripper(mouth)

Fig. 2: Binocular-gripper setup is inspired from biological dual-eyes-one-mouth system.

support degradation, and so on. The most related paper [8] also investigate this issue with a learning-based method. Here we address the importance of reducing eye-hand calibration effort and propose our method to challenge performing a precise task under a system that is exempted from manual eye-hand calibration. To intentionally introduce eye-hand error, we perform all tasks with the adjustable camera frame.

Our contribution in this work is summarised as follow:

1. Compared with benchmarks, our proposed binocular alignment learning method shows a competitive success rate in insertion (95%), screwing (95%), pick-small-item (100%), place-small-box (100%) with the lowest effort and economic cost.

2. We address the eye-hand calibration issue and propose different auto self-calibration methods SAMLs. We give detailed ablation studies on each SAML method. This solves the unknown eye-hand coordination issue in the tasks.

3. We propose the camera pose randomization training and successfully adapt the learned policy from simulation to the real task (Sim2Real) using domain randomization and feature disentanglement.

4. We propose a novel dual-arm working setup in the screwing task, in which the right arm holds the camera and the left arm perform the task.

II. Related Work

For precision manipulation tasks, there are visual-based methods [9][10][5], force-based methods [11] and hybrid methods [12]. In this work, we only look into learning-based and visual-based methods, and select insertion as the key task to research.

[12][9] proposes novel peg-insertion methods using visual DeepRL method. [13][10] proposes an novel pure visual-based solution. However, some additional sensors, information, efforts or costs are required by these methods, such as well camera calibration [12][13], high quality visual system[13], torque/force sensors [12], hard-to-get information like goal-pose [6][2] and goal-pose-image [9]. [8] proposes a novel way to learn eye-hand calibration using CNN, yet not support camera out of calibrated position.

Reinforcement Learning (RL) has been widely used to solve robotics tasks, such as solving a Rubik's cubic with an anthropomorphic robotic hand [14], tossing novel objects [4], performing peg-in-hole insertions [12][9][6]. In this work, we also follow Proximal Policy Optimization [15] to learn the control policy and overcome the camera position disturbance. For the sim2real, there proposed novel methods as domain randomization and augmentation [14].

Therefore, in this work, we target to propose a learning-based method to achieve high precise tasks, at the same time minimize the economical cost and calibration efforts and perform sim2real.

III. Preliminaries

In this work, we model the four tasks insertion, screwing, pick, and place as Points Alignment tasks. In these tasks,

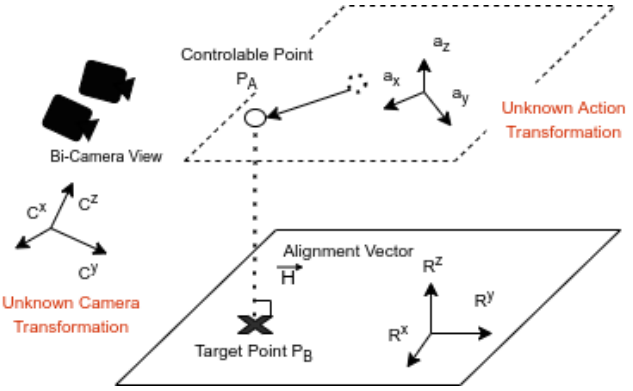
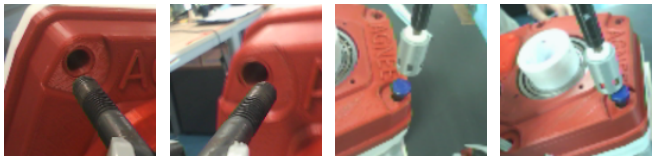


Fig. 3: Task Definition



(a) Insert(L) (b) Insert(R) (c) Screw(L) (d) Screw(R)

Fig. 4: Binocular Camera View. L, left camera view. R, right camera view. Pick and Place are omitted. Agent perform the task with this as observation.

there are controlled point P_A, target point P_B and target vector $\overrightarrow{H}$. The agent should control the position of controllable point P_A to align the two points P_A and P_B with the $\overrightarrow{H}$, seeing Fig.3.

High precision is always required in these tasks. Otherwise, the task will fail. And the setup details can be found in Sec.V and Fig.1. To address the eye-hand calibration issue, we assume eye-hand transformation is unknown. To generate unpredictable camera pose errors, we design our camera frame pose adjustable.

We formulate this visual servoing process as a partially observable Markov decision process (POMDP). **Action space** $\overrightarrow{a} \in A^x \times A^y$ (robot's base Cartesian frame) is given by the position control for the controlled point P_A. This process is rolling out under discrete time step $t \in [1;T]$, T is the max steps in each episode. Observation space is giving by two low-resolution RGB images. The agent detects the controlled point P_A and target point P_B in the raw images as the **observation space**. **Reward** r_t is a scalar variable only given in simulation. Transition process is described as $p\left(o_t, r_t | a_{t-1}, o_{t-1}\right)$. The agent in the simulation should be trained to maximize the total rewards $\sum r_t$ in the whole process.

IV. Approach

A. Approach Overview

This learning part introduces the Self Action Mapping Learning (SAML) to correct eye-hand coordination and Points Alignment Learning (PAL) to perform task policy.

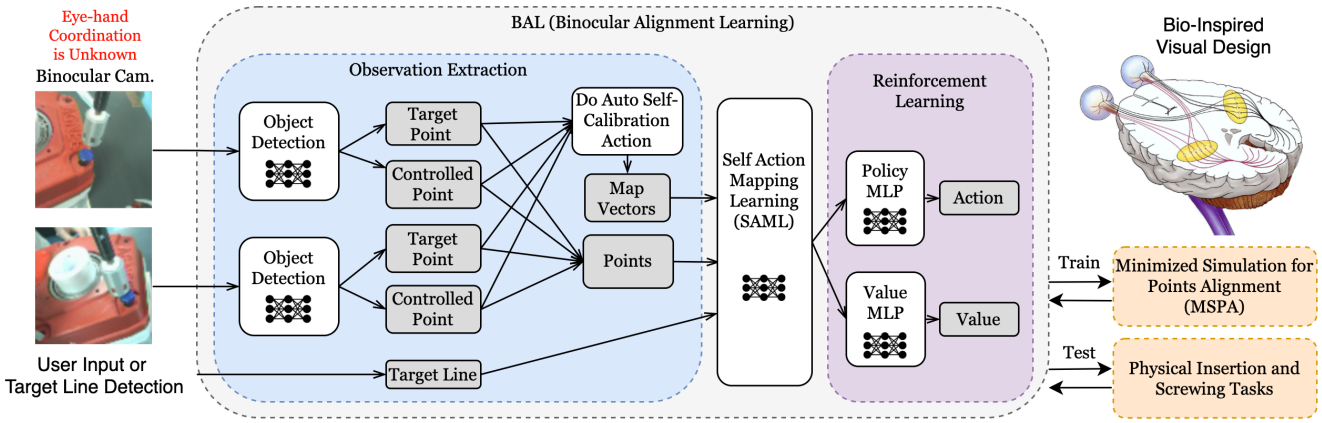


Fig. 5: Binocular Alignment Learning (BAL) Architecture. A pre-trained YOLO network is used to detect key points. Details about auto self-calibration action can refer to Sec.IV-B. Target line detection will be discussed in future work, this work we use user input target line.

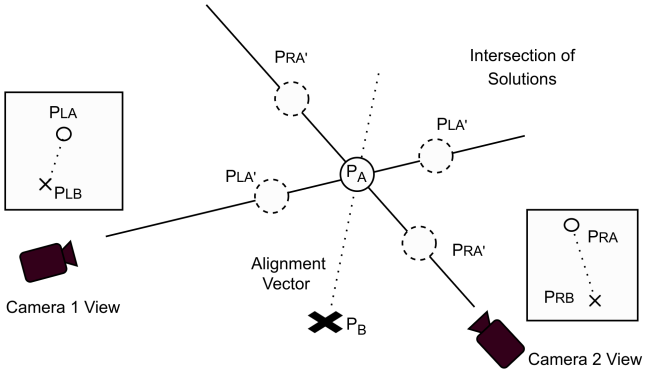


Fig. 6: Intersection of Solutions in Binocular Vision

Minimal Simulation for Points Alignment (MSPA, Sec.IV-C) is used to efficiently learn the policy in a few minutes training. The general approach is described in Fig.5, Fig.8 and Algorithm.1. Fig.7 gives network designs for SAML methods.

The inputs to the model are two images. The model is allowed to use the object detector (we use YOLO) to capture the key points in the images, annotated as $P_{ij} \in \{P_{LA}, P_{LB}, P_{RA}, P_{RB}\}$, L, R stand for the left and right cameras selection, A, B represents the controllable point and target point. Using the self action mapping learning (SAML) methods, the robot generates the self-calibration vector (SCV) V_{ik}, $i \in \{L, R\}$ and $k \in \{1, 2\}$. Using the SCV and P_{ij}, the model learns a camera-pose adaptive control policy using Self Action Mapping Learning (SAML). While testing on the real robot, the RL agent and YOLO works at $1 \sim 2Hz$, the robot controlled by ROS Moveit at $20Hz$ level, camera $30Hz$.

Algorithm 1: Binocular Alignment Learning and Sim2Real

Init: RL Agent $\Psi(obs)$, Trained Object Detector $Y(Image)$, min error ε
Train in Simulation (MSPA)
  while *Train* **do**
    foreach *Episode* **do**
      $P_A, P_{camera} \leftarrow$ Random();
      $env \leftarrow$CreateEnv(P_A, P_{camera});
      Interact(π_Ψ, env) until done ;
    end
    if *Update* **then**
      Gradient Update Ψ (PPO Convention)
    end
  end
end
Test in Real
  Input: Target Alignment Line $\overrightarrow{H}$
  $SCV_{1,2} \leftarrow$ do auto calibration action, collect;
  /* Alignment */
  while *err*$> \varepsilon$ **do**
    $P_i \leftarrow Y(Images)$;
    obs$\leftarrow$Preprocess($P_i, \overrightarrow{H}, SCV$);
    $a \leftarrow \pi_\Psi(obs)$;
    robot input (a);
    err$\leftarrow$GetError($\overrightarrow{H}, P_i$);
  end
  /* Alignment Finished */
  Do action$\in$\{insert, screw, pick, place\}
end

B. Self Action Mapping Learning (SAML)

To address the eye-hand coordination problem, we propose methods to achieve the self action mapping learning. We designed the PML, MML, IML and MonoIML. In those approaches the agent need to perform a self-calibration action $a_{d1} = [1, 0]$ (move towards A_x direction for 1 unit), come back to initial position and perform $a_{d2} = [0, 1]$ (move

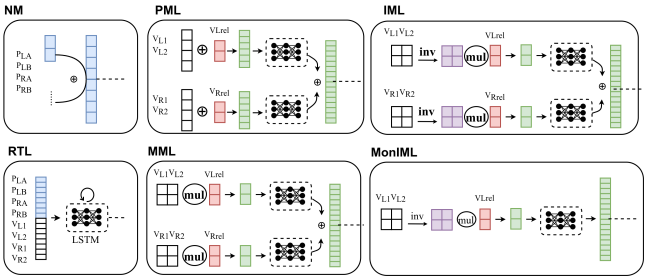


Fig. 7: Self Action Mapping Learning (SAML) methods

towards A_y direction for 1 unit) in sequence to collect the corresponding position changes in the observation. They are annotated as self-calibration vector (SCV) V_{ik} with $i \in \{L, R\}, k \in \{1, 2\}$ representing the position translation of P_{iA} as the result of action a_{d1} and a_{d2}. Target related vector (TRV) V_{irel} represents the relative position in the camera observation of $P_{iA}, P_{iB}, i \in \{L, R\}$. L_Θ is the MLP (multilayer perceptron) block for information extraction out of vectors. F is the flatten layer. Given H_L, H_R as the alignment target vector by user. All the pipelines are described in the Fig.7.

$$V_{L1}, V_{R1} \hookleftarrow a_{d1}$$

$$V_{L2}, V_{R2} \hookleftarrow a_{d2}$$

$$V_{Lrel}, V_{Rrel} = V_{P_{LA}P_{LB}} - H_L, V_{P_{RA}P_{LB}} - H_R$$

None-Maping (NM) has no action-mapping learning. Hence the robot only observe the object detection results from the last layer. With the random noise given to the camera position, this approach should perform the worst. This approach serves as baselines to be compared. **Monocular Mapping Learning (MonIML)** utilizes only one camera observation in IML. $o = L_\Theta(matmul((V_{i1}; V_{i2})^{-1}, V_{irel})), i \in \{L\}$

Parral Mapping Learning (PML) concatenates SCV with TRV and flattens them into a 1D tensor. $h_i = L_\Theta((V_{i1}; V_{i2}), V_{irel}), i \in \{L, R\}; o = F(h_L; h_R)$ However, since SCV and TRV are from a different domain, it can be difficult for the model to learn the relation between SCV and TRV.

Recurrent Time-based Learning (RTL) recurrently process TRV, $h, o = LSTM(h, [V_{lrel}; V_{rrel}])$ to learn the action-observation coordination.

Mat-mul Mapping Learning (MML) multiplies the SCV and TRV, $h_i = L_\Theta(matmul((V_{i1}; V_{i2}), V_{irel})), i \in \{L, R\}$; $o = [h_L; h_R]$.

Inverse Mapping Learning (IML) multiplies the inverse of SCV (V_{ik})to the TRV. It inverses the SCV into an easy learning domain U (details of experiments are in Sec.V). $h_i = L_\Theta(matmul((V_{i1}; V_{i2})^{-1}, V_{irel})), i \in \{L, R\}$; $o = [h_L; h_R]$

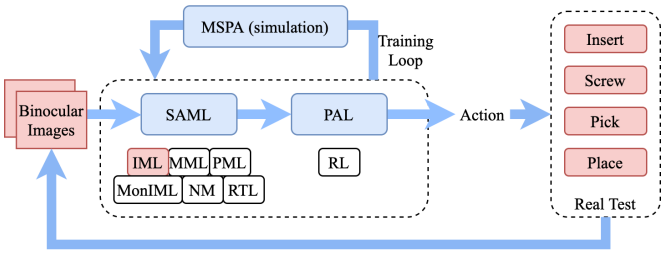


Fig. 8: Components Workflow. SAML:Self Action Mapping Learning (in Sec.IV-B); PAL:Points Alignment Learning (in Sec.IV-C); MSPA: Minimal Simulation for Points Alignment (in Sec.IV-C)

.

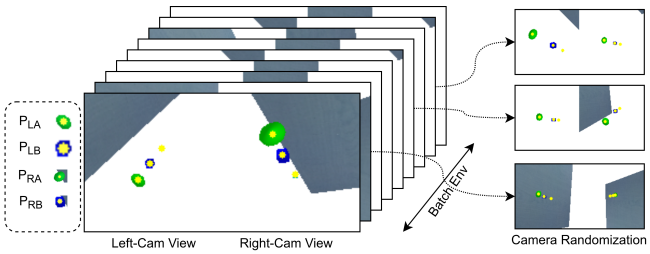


Fig. 9: Simulation. This is the Minimal simulation for Points Alignment (MSPA), which only provides points to train policy, not the image.

C. Points Alignment Learning (PAL) and Minimal Simulation (MSPA)

Points Alignment task learning is described as a Markov Decision optimization problem introduced in Sec.III. The Proximal Policy Optimization (PPO) approach is used to train the policy for action generation. The rewards are defined as $r = clip(-80 * D + 1, -10, 1)$ to help the agent learn to approach the perform alignment task. D is the distance between the current controllable point position and target position. D is only applicable in the simulation, while in the real inference there is no D.

In the simulation, the camera position is randomized to help learn a camera position adaptive strategy. The key points are given in the simulation as the green point P_B and the blue point P_A. The third yellow point is a random dot located on the given target alignment vector H (Sec.III). For learning-based methods, training in the simulation then testing in the physical world is much more efficient and less dangerous than directly training in the physical environment. With feature disentanglement and domain randomization, RL policy optimization (PPO convention) the policy can successfully adapt to the real test.

V. Experiment and Discussion

We conduct the real experiments by setting up the MOVO robot to perform four kinds of points alignment tasks(insertion, screwing, pick, and place). The whole setup is shown in Fig.1. For each task, 20 attempts are conducted

TABLE I: Ablation study of Self Action Mapping Learning (SAML) and Camera Randomized Training. FC use fixed camera pose. RC use randomized camera pose. Different methods are introduced in Sec.IV-B. . Results are collected from simulation.

Sim. Test	FC Train FC Test	FC Train RC Test	**RC Train FC Test**	RC Train RC Test
PML	2%	9%	4%	20%
MML	3%	0%	91%	59%
NM	6%	6%	1%	11%
MonIML	3%	15%	36%	26%
RTL	2%	4%	3%	61%
IML	**11%**	**37%**	**96%**	**66%**

TABLE II: Success rate on real robot using policy BAL+IML. Test setup is shown in Fig.1

Real Test	Insert	Screw	Pick	Place
Success Rate	95%	95%	100%	100%

for the purpose of evaluating the success rate, which is shown in Table.II

A. Benchmark and Eye-hand Coordination Learning

Table.III compares our method performance with recent years' learning-based methods in high precision tasks, especially in the task of insertion. Clearly, the performance of our method is highly competitive, achieving a considerably high task success rate solely relying on visual detection and eliminating the need for manual eye-hand calibration. In contrast, other methods in comparison are either still in the simulation stage or depend on manual hand-eye calibration, significantly increasing the system's complexity and workload.

Ablation study of different SAML methods is given in the Table.I. To overcome eye-hand calibration, BAL(IML) has the best success rate of 96% with a camera position adaptive policy. Random camera poses training benefits from domain randomization and improves the performance from 11% to 96%. It also shows fixed camera training can not solve an unseen eye-hand coordination situation. Table.II shows BAL(IML) can solve tasks with a success rate of $95\%-100\%$ in a real robot. Results also show using a inverse method, the IML improved the performce from MML(91%) to IML(96%).

TABLE III: The comparison of our method with other learning-based approach. Manual Calib. indicates if this method requires manual eye-hand coordination calibration

Method	Vision Based	Manual Calib.	Success Rate
M. V. et al[12]	✓	Need	97%
F. B. et al[10]	✓	No Need	100% in Simulator
J. C. et al[16]	✓	Need	100%
G. S. et al[9]	✓	Need	84%-100%
B. C. et al[2]	×	Need	65%-100%
Ours(BAL)	✓	No Need	95%-100%

B. Binocular is better than Monocular

To show the necessity of a binocular camera system compared with a monocular camera, we give a baseline using MonIML (Monocular-based IML). Table.I shows, under RC training and RC testing setup, IML (96%) successes much more than MonIML (36%). Using FC training, IML (11% in FC test and 37% in RC test) also shows much better performance than MonIML (3% in FC test and 15% in RC test). Therefore, binocular-based methods (i.e., IML) learns a better policy in solving target tasks, and the policy is adaptive to camera position.

C. High Benefits with Low Cost

Modern industrial tasks strive to achieve high productivity while reducing costs. This work exclusively relies on two rough RGB cameras (20 SGD) to execute four types of robotic tasks, achieving a high success rate. In comparison to other methods mentioned in Table 3, it effectively reduces costs, promoting applications in the industrial sector. For example, [12] requires not only RGB cameras, but also relies on information from T/F sensors, with each force sensor potentially incurring a cost exceeding 500 SGD.

D. Limitation and Future Work

However, our work also remains minor limitations.

1.We didn't discuss the camera distortion and give an ablation study on the camera distortion with SAML.

2. As an example, considering the bolt insertion task, our study assumes the bolt is oriented vertically to the working plane at initial stage. The alignment task focuses on the alignment of the bolt head and hole within the 2D plane, without taking into consideration errors along the Z-axis. We will proceed to realize 3D alignment in the following research in the future.

VI. Conclusion

Precision manipulation is a long-term challenge in robotics. In this work, we propose BAL to successfully solve insertion, screwing, pick-small-item, and place-small-box with success rate of $95\% - 100\%$. Additionally, we also reduced the cost of the setup, making it economically efficient. We addressed the importance of adaptability under poor eye-hand coordination and proposed SAML methods to solve it with a detailed ablation study. We proposed a practical sim2real pipeline and successfully adapt it to real robot test.

References

[1] T. Inoue, G. De Magistris, A. Munawar, T. Yokoya, and R. Tachibana, "Deep reinforcement learning for high precision assembly tasks," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 819–825.

[2] C. C. Beltran-Hernandez, D. Petit, I. G. Ramirez-Alpizar, and K. Harada, "Variable compliance control for robotic peg-in-hole assembly: A deep-reinforcement-learning approach," *Applied Sciences*, vol. 10, no. 19, 2020, ISSN: 2076-3417.

[3] H.-S. Fang, C. Wang, M. Gou, and C. Lu, "Graspnet-1billion: A large-scale benchmark for general object grasping," in *CVPR*, Jun. 2020.

[4] A. Zeng, S. Song, J. Lee, A. Rodriguez, and T. Funkhouser, "Tossingbot: Learning to throw arbitrary objects with residual physics," 2019.

[5] A. Hundt, B. Killeen, H. Kwon, C. Paxton, and G. Hager, ""'good robot!": Efficient reinforcement learning for multi-step visual tasks via reward shaping," Sep. 2019.

[6] G. Schoettler, A. Nair, J. A. Ojea, S. Levine, and E. Solowjow, *Meta-reinforcement learning for robotic industrial insertion tasks*, 2020.

[7] A. Mousavian, C. Eppner, and D. Fox, "6-dof graspnet: Variational grasp generation for object manipulation," Oct. 2019, pp. 2901–2910. DOI: 10.1109/ICCV.2019.00299.

[8] H. Cheng, Z. Zhang, and W. Li, "Efficient hand eye calibration method for a delta robot pick-and-place system," in *2015 IEEE CYBER*, 2015, pp. 175–180. DOI: 10.1109/CYBER.2015.7287930.

[9] G. Schoettler, A. Nair, J. Luo, *et al.*, *Deep reinforcement learning for industrial insertion tasks with visual inputs and natural rewards*, 2019.

[10] F. de La Bourdonnaye, C. Teulière, J. Triesch, and T. Chateau, "Learning to touch objects through stage-wise deep reinforcement learning," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 1–9. DOI: 10.1109/IROS.2018.8593362.

[11] *A Learning-Based Framework for Robot Peg-Hole-Insertion*, Dynamic Systems and Control Conference, V002T27A002, Oct. 2015. DOI: 10.1115/DSCC2015-9703.

[12] M. Vecerik, O. Sushkov, D. Barker, T. Rothörl, T. Hester, and J. Scholz, "A practical approach to insertion with variable socket position using deep reinforcement learning," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 754–760. DOI: 10.1109/ICRA.2019.8794074.

[13] G. Thomas, M. Chien, A. Tamar, J. A. Ojea, and P. Abbeel, "Learning robotic assembly from cad," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 3524–3531. DOI: 10.1109/ICRA.2018.8460696.

[14] OpenAI, I. Akkaya, M. Andrychowicz, *et al.*, *Solving rubik's cube with a robot hand*, 2019.

[15] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, *Proximal policy optimization algorithms*, 2017.

[16] J. Triyonoputro, W. Wan, and K. Harada, *Quickly inserting pegs into uncertain holes using multi-view images and deep network trained on synthetic data*, Feb. 2019.