Test-Time Training for Motor Control

Jon Chen jonchen25@
Preston Fu prestonfu@
Elvis Hsieh htelvis92@

Abstract

We present a practical and intuitive approach to resolve the domain shift issue between the pre-trained domain and the downstream domain in robotic motor control with Test-Time Training. We propose Test-Time Training for Motor Control (TTT-MC), which adapts to observation distribution by optimizing the vision encoder for each pixel observation using self-supervision. TTT-MC is conceptually simple and outperforms self-supervised vision methods in robotics at inference time.

1 Introduction

Modern machine learning has focused on learning self-supervised representations with large neural networks and fine-tuning networks on new tasks with domain-specific data. Areas such as vision and language have experienced significant growth through this paradigm [23, 1].

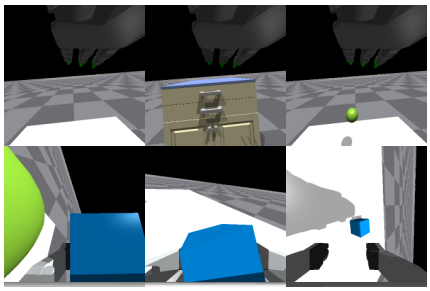
In parallel, however, supervised learning has demonstrated weak performance under distribution shifts. Recht et al. [15] shows that minor differences between training and testing distributions result in significant drops in performance of state-of-the-art models. In general, for supervised models in deployment, there is no guarantee that training and testing data are sampled from the same distribution. As a result, for a trained model to perform well under distribution shifts, it must be robust to any potential shifts that could occur in the testing data [11, 9].

Sun et al. [24] proposes Test-Time Training (TTT), which enables changes in the model parameters during test time with the insight that each test input reveals information about the test distribution. TTT creates a self-supervised learning problem in addition to the main task. During training, TTT optimizes both the main task and the self-supervised task; at test time, it adapts the model with the self-supervised task alone for each test input to make a prediction for the main task. Gandelsman et al. [8] proposes using a masked autoencoder (MAE) as the self-supervised task.

Xiao et al. [28] proposes Masked Visual Pretraining (MVP) and shows that self-supervised visual pre-training on real-world images are more effective than supervised methods for learning motor control tasks from pixels. Their network encodes the input image using a visual encoder and concatenates it with proprioceptive information to obtain an embedding. The only learnable parameters are in a lightweight controller, which accepts the combined embedding to produce actions. Their encoder is a Vision Transformer (ViT) [7], trained on images and videos from the Internet [6, 5, 10, 20]. This approach uses representations of natural images, without any task-specific fine-tuning. While conceptually appealing and computationally efficient, it does not account for the significant disparity in the distributions of training and test images. ImageNet contains classes of images including "mud turtle," "reflex camera," and "triumphal arch," whereas the Pixel Motor Control (PixMC) environments created by Xiao et al. [28] produces much simpler and more geometric images from Isaac Gym, a fast GPU-based simulator [12], as shown in Figure 1.

We propose Test-Time Training for Motor Control (TTT-MC), which uses masked reconstruction as the self-supervised objective. Like Xiao et al. [28], we use a pre-trained ViT for our encoder and do not perform any task-specific fine-tuning prior to the RL task. Following each training iteration, however, we unfreeze our encoder, enabling it to learn meaningful representations of the collected

(a) ImageNet sample [19]



(b) PixMC pixel sample [14]

Figure 1: Previous approaches mostly freeze the vision encoder pre-trained on ImageNet. However, there's obvious distribution shift from the pre-trained data (ImageNet) and the deployment environments (PixMC pixel observation).

on-policy image buffer. We show that our method can consistently outperform self-supervised methods at inference.

2 Background

2.1 Masked Visual Pretraining

MVP [28] proposes using self-supervised visual pre-training for motor control task. This method involves first training a visual encoder with real-world images through masked image modeling without task-specific fine-tuning, and then freezing the visual encoder. Unlike traditional methods, MVP leverages large-scale real-world image data for pre-training, avoiding the need for labeled data. These images provide a rich variety of visual scenarios, which equips the model witht he ability to generalize across different visual contexts. Neural network controllers are then trained on this encoder with RL to handle various motor control tasks.

Evaluating its performance, MVP consistently outperforms supervised encoders, sometimes even matching oracle state models, indicating its efficacy in learning motor tasks from visual inputs. Furthermore, MVP demonstrates effectiveness across different robotic configurations. Whether the robot has a simple mechanical structure or a complex one with multiple degrees of freedom, the MVP-trained model can adapt and control these various configurations effectively. MVP's ability to handle tasks in conditions or with objects not seen during training is an indicator of its strong generalization capabilities. This is essential for deploying robots in real-world situations where they must adapt to new and unforeseen challenges.

2.2 Distribution Shift

When doing pre-training, models are often trained on a set of data whose distribution differs from the actual distribution. This shift in distribution can cause even the state-of-the-art models to break [15]. One common method to address distribution shift is to train on a wider set of distribution in hopes that it is more similar to the actual test distribution. Another approach is through unsupervised domain adaptation, in which we pre-train our model using some unlabeled data from the test distribution. This method is sometimes successful since most of the distribution shift that occur is on the training inputs rather than the output labels. However, this method assume that we have access to the testing data at training time.

2.3 Test Time Training

Distribution shift is a key component of modern machine learning research, and one recent line of work to address distribution shift during test time is TTT [24]. TTT is an algorithmic framework with the core idea that each test instance is able to define its own learning problem, and is hence its own target of generalization. The main differentiation between Test Time Training and earlier proposed method is that TTT does not avoid distribution shift; rather, it attempts to learn during training time and adapts.

When a new test sample is received, TTT updates the model's parameters before making a prediction on that sample. This update is based on a self-supervised task specific to the test sample, allowing the model to adjust its parameters to better fit the test distribution. Then the new weights are discarded and reset to the original after each iteration. In general, test-time-training do not assume the test data are drawn from the same distribution. By adapting the model for each test sample or batch of samples, TTT can effectively handle changes in data distribution that occur over time, which is a significant challenge in many real-world applications.

2.4 Test Time Training with Masked Autoencoders

Building upon previous work, Gandelsman et al. [8] proposes that Masked Autoencoders (MAE) works well with TTT. Integrating MAE with TTT represents a significant shift in tackling generalization challenges under varying test distributions. This approach utilizes the self-supervised learning capabilities of MAEs to dynamically adapt models to new test inputs, potentially enhancing performance across diverse visual benchmarks. MAE, known for its proficiency in capturing spatial smoothness in natural images, is a robust choice for generating self-supervised signals for TTT.

Utilizing a MAE based on a ViT-Large encoder and a ViT-Base head, experiments demonstrate significant improvements in model performance when applying MAE with TTT, especially under conditions of distribution shifts. The results on ImageNet-C, a benchmark for object recognition under various corruptions, particularly highlight this enhancement, with MAE with TTT outperforming both the baseline and traditional rotation prediction methods. Further validations on the ImageNet-A, ImageNet-R, and Portraits datasets reiterate the method's robustness and adaptability, showing substantial gains over baseline models. This empirical evidence underscores the potential of MAE with TTT in improving model accuracy and generalization, especially in scenarios with varying and challenging distribution shifts.

3 Methods

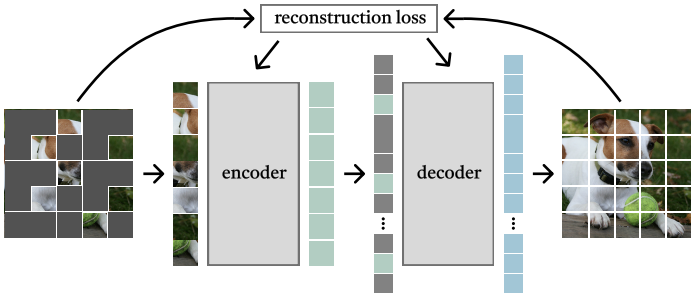
Our approach uses TTT for visual representation for motor control, particularly robotic manipulation. We show that it attenuates the distribution shift between pre-trained data distribution and the downstream data distribution.

3.1 Masked Visual Pre-training–Lite

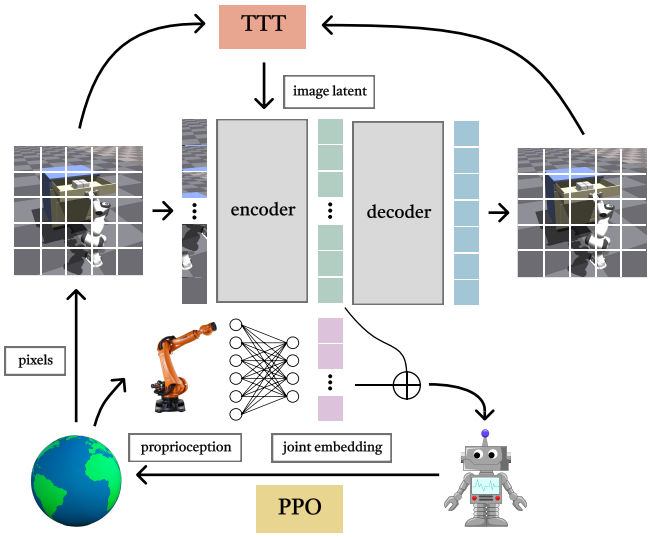
Training robotic agents is computationally intensive; unlike supervised and self-supervised settings, collecting samples in real time is costly, and running real-world experiments requires extensive real-world effort. As such, in the MVP setting, it is most practical to use lightweight ViTs, despite smaller models being relatively under-studied in the field of self-supervised learning. Wang et al. [27] presents a vanilla lightweight ViT pre-trained on an MAE objective that show comparable performance to larger SOTA networks and shows that they can perform well on downstream tasks.

Self-supervised objective. Xiao et al. [28] showed that the self-supervised approach consistently outperforms supervised objective for pre-trained encoder. Therefore, we use a masked autoencoder (MAE) as our self-supervision objective, as it outperforms other self-supervised pre-training methods [4, 13, 22]. In MAE training objective, it randomly masks different patches of input image and reconstructs masked patches with a ViT backbone. This approach enhances the efficiency of learning visual representations without relying on dataset or task-specific augmentations. Notably, MAE has emerged as a popular framework in self-supervised learning within the field of computer vision and has shown promising results in simulating motor control tasks [28, 14].

Model Architecture. Previous work uses a ViT-Large encoder, which contains 307M parameters [14]. Here, we employ ViT-Tiny [26] as our vision encoder, which only contains 5.7M parameters. The encoder consists of 12 transformer block with 12 attention heads, producing an image embedding of size 192. The decoder consists of 1 transformer block with 3 attention heads. We show that relatively small ViTs have sufficiently good representations to be robust to a wide variety of vision-based robotic tasks.



(a) Phase 1: Masked autoencoder is pre-trained on ImageNet.



(b) Phase 2: TTT learns a controller while updating the MAE encoder weights after each training iteration.

Figure 2: Two-part training pipeline.

3.2 Test-Time Training

Deep neural networks are prone to performance degradation under a domain shift between the source (pre-trained) data and target (environment) data. Inspired by Sun et al. [24], we turned use pixel observations as unlabeled test samples for the self-supervised task. Formally, we collect an observation $\mathbf{o}_t = [\mathbf{o}_t^{\text{pro}}, \mathbf{o}_t^{\text{pix}}]$. We have four networks: the masked autoencoder consisting of ViT-Tiny encoder $\mathcal{E}_\varphi$ and Transformer decoder $\mathcal{D}_{\varphi'}$, a feed-forward network f_ψ to produce joint embeddings, and a PPO controller $\pi_\theta(a|s)$. φ is initialized from the pre-trained model. Every N environment steps, the policy and joint embedder are updated,

$$\theta, \psi \leftarrow \arg\min_{\theta,\psi} L_{\text{PPO}}(\pi_\theta(a \mid [f_\psi(\mathbf{o}_t^{\text{pro}}), \mathcal{E}_\varphi(\mathbf{o}_t^{\text{pix}})])$$

Every K policy iterations, the vision encoder is updated (while the decoder remains frozen, as in [8],

$$\varphi \leftarrow \arg\min_{\varphi} \left\| \mathcal{D}_{\varphi'}(\mathcal{E}_\varphi(\text{mask}(\mathbf{o}^{\text{pix}}))) - \mathbf{o}^{\text{pix}} \right\|_2^2,$$

where $\mathbf{o}^{\text{pix}}$ is the batch of images that has just been collected. This architecture is depicted in Figure 2.

4 Experiments

In our experiments, we aim to study how the pre-trained representation can be re-used and adopted to different downstream robotic learning tasks.

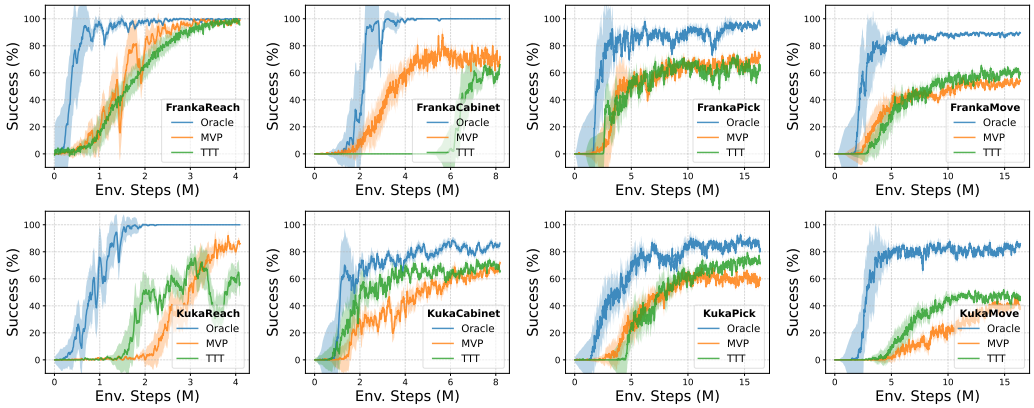


Figure 3: We plot the success rate as a function of environmental steps across eight different tasks within the PixMC benchmark. These tasks are executed using either the Franka arm, equipped with a parallel gripper, or the Kuka arm, featuring a multi-finger hand. Our results demonstrate that the Test-Time Training for Motor Control (TTT-MC) approach, leveraging a lightweight ViT architecture, exhibits a marked performance enhancement compared to MVP without TTT in 7 out of the 8 tasks. These results highlight the significant impact of our test-time training's effectiveness in substantially enhancing representation quality for a range of motor control tasks.

4.1 Setup

Data for pre-training. We consider ImageNet [17] as out main pre-training data. ImageNet-trained models have been shown to transfer to a variety of different tasks across language and vision fields[3, 21, 6].

Robots. PixMC consists of two robotic arms, each equipped with different gripping mechanisms. The first is "Franka," a widely used robot in research featuring a 7-degree-of-freedom (DoF) arm and a 2-DoF parallel jaw gripper. The second is "Kuka," which is a combination of a KUKA LBR IIWA arm with 7 DoFs and a multi-finger, 16 DoF Allegro hand.

Simulation environment. Radosavovic et al. [14] introduces several pixel-based motor control tasks. PixMC is based on the NVIDIA IsaacGym simulator [12], which performs parallelized simulation on a GPU. Rudin et al. [16] showed that the sim-to-real transfer is amenable based on the policies trained on IsaacGym.

PixMC provides high-resolution pixel observations for each robot, specifically the Franka and Kuka setups. Both setups utilize wrist-mounted cameras for tasks, and offer proprioceptive data along with hand-engineered states that include 3D poses and relevant object and goal relationships. Task-specific reward functions are designed for training RL policies, and reward-independent success metrics are used to quantify the distance from an agent or object to a goal location over a sufficient number of time steps.

Hyperparameters. Across all experiments, we use an actor and critic with hidden sizes $[256, 128, 64]$ and run gradient steps on the vision encoder every $K = 20$ iterations. We run ablations on learning rate in $\{0.0005, 0.001, 0.005\}$ on select experiments and find experimentally that 0.001 tends to perform best across all tasks but does not result in significant performance differences. For PPO, we use the default hyperparameters, i.e. clip range 0.1, a cosine learning rate schedule, $\gamma = 0.99$, and $\lambda = 0.95$.

4.2 Sample Complexity

After pre-training, the MAE encoder model is frozen and used as a perception module for the control policy. The policy is trained by Proximal Policy Optimization (PPO) [18]. We include success rates over training on 8 challenging tasks from PixMC. In our setting, we condsier oracle state model having all the infrmation of the state (i.e., position, orientation, and velocity of the object, goal and robot in world-coordinate system) as our upper bound in the environment. In Figure 3, TTT-MC achieve a higher success rate than the MVP in 4 out of 8 tasks. We also noticed that TTT-MC is

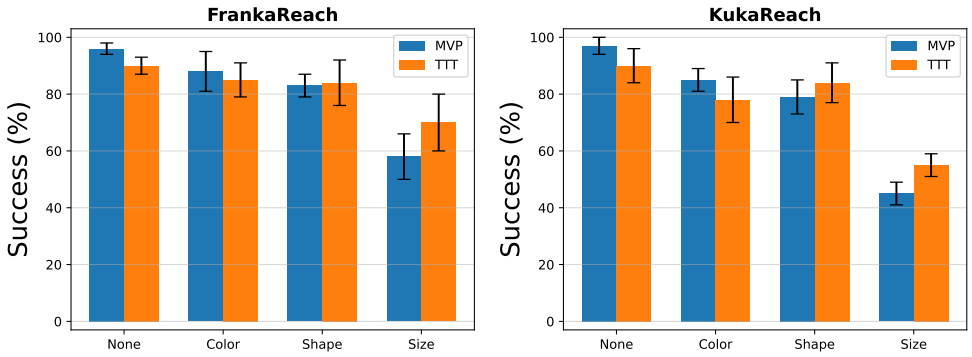


Figure 4: At test time, we add a distractor object differing from the training object in terms of color (blue vs. green), shape (cube vs. sphere), or size (4.5cm vs. 6cm) to test how well TTT-MC and MVP are able to generalize.

generally more stable than MVP, which suggests test-time training successfully adapts the model to the deployment distribution, allowing the model to capture useful visual features.

4.3 Robustness to distractors

In this section, we focus on evaluating the robustness of our pre-trained models against various distractors. In the FrankaPick and KukaPick tasks, we introduce distractors that varied in color, shape, and size during the test phase to assess the adaptability of the models.

Introducing Distractors. Initially, the robots were trained to pick up a specific object: a blue cube with a side length of 4.5cm. At test time, we introduced distractors that differed from the training object in one of three aspects: color (blue vs. green), shape (cube vs. sphere), or size (4.5cm vs. 6cm). These variations were chosen to test the model's generalization ability with and without test-time training.

Model Performance. The TTT-MC model, with test-time training, dynamically adapts its parameters to better interpret the available visual cues, even with limited depth information. This adaptability allows the model to develop a more nuanced understanding of size variations, despite the scale ambiguity presented by the camera setup. From the Figure 4, we observed that the TTT-MC performs better than MVP [28] in size and shape distractor tasks. The results further proved how the TTT process enables the model to refine its feature representations for each specific scenario, improving its performance in tasks where size discrimination is critical.

4.4 Generalization to objects of various geometries

Based on the original MVP paper [28], our experiments were designed to assess the model's adaptability to objects of varying geometrical shapes, specifically focusing on four distinct object types from the YCB dataset [2]: a box, a can, a mug, and a banana. We modified the existing tasks in the FrankaPick and KukaPick, allowing for a direct comparison of the model's performance across different geometrical contexts in Figure 5. The outcomes of these modified tasks revealed that our model demonstrated enhanced performance in object picking tasks across the varied geometrical shapes with non-TTT approach. Therefore, we attribute such performance to the test-time training, which evidently enhance the model's capability to generalize across different object geometries.‘

4.5 Computational Performance

All of our experiments are run on a single NVIDIA Quadro RTX 6000 GPU. Table 1 indicates the wall clock time for each of the four tasks (Franka and Kuka tasks have the same wall clock time to within 1%, so only the Franka times are displayed in the table). We find that, on average, TTT-MC tends to have runtimes around $3\times$ those of MVP for the same number of environment steps. This follows naturally from the vision encoder including an order of magnitude more parameters than the controller (roughly 200k across all tasks). Given that the

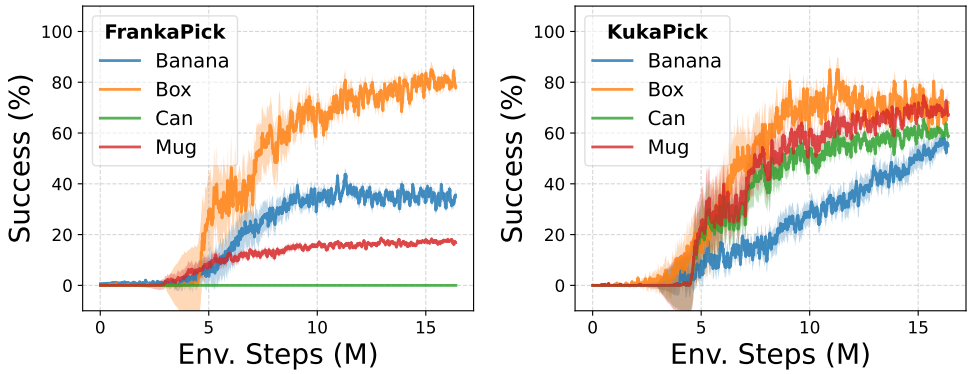


Figure 5: From the YCB datasets, we imported additional object and re-trained the controller for each. These object all feature different geometries. In the environment with the Kuka robot with the Allegro hand we were able to achieve over 50 success for all objects

Table 1: Wall Clock Time By Experiment

Task	Steps (M)	MVP Time (hr)	TTT Time (hr)
Reach	4.1	0.65	1.99
Cabinet	8.2	2.12	6.44
Pick	16.4	2.74	9.04
Move	16.4	2.69	9.24

5 Conclusion

In this paper, we show that Test-Time Training (TTT) is effective in mitigating the distribution shift between pre-trained data and deployment data for motor control. Our method dynamically adapts vision encoder parameters to the specific distribution of the deployment environment, ImageNet and PixMC respectively in our case. We further show that with lightweight ViT architecture as our the vision encoder, we are able to efficiently adapt to new environments without extensive computation in test-time training loop.

An avenue of future work is exploring the impact of different vision encoders. Radosavovic et al. [14] found that larger encoders do not improve performance and may require more data or different training regimes. Since collect image data in batches from a single task, we suspect that passing in image data sequentially as videos would achieve higher performance using a pretrained VideoMAE and temporal objective [25]. We do not have the required compute to pre-train our own MAE's from scratch, but in general scaling data and models is an exciting subject of future work.

Another line of further investigation is exploring other forms of real-time domain adaptation techniques in motor control and beyond, as well as self-supervised methods to refine useful visual representations.

6 Project Scope

The group split the coding work for this project evenly in synchronous sessions together via VS Code Live Share. Together, we migrated the TTT pipeline to the MVP codebase, which was challenging due to different data formats across each infrastructure, and various architectural decisions for incorporating compatibility with different formats for pre-trained architectures. In the paper, Preston focused mostly on experiments and ablations, Jon on relevant background, and Elvis on design decisions and analysis.

References

[1] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020.

[2] Berk Calli, Aaron Walsman, Arjun Singh, Siddhartha Srinivasa, Pieter Abbeel, and Aaron M. Dollar. Benchmarking in manipulation research: Using the yale-cmu-berkeley object and model set. *IEEE Robotics & Automation Magazine*, 22(3):36–52, September 2015. ISSN 1070-9932. doi: 10.1109/mra.2015.2448951. URL `http://dx.doi.org/10.1109/MRA.2015.2448951`.

[3] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime multi-person 2d pose estimation using part affinity fields, 2017.

[4] Xinlei Chen, Saining Xie, and Kaiming He. An empirical study of training self-supervised vision transformers, 2021.

[5] Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Sanja Fidler, Antonino Furnari, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, and Michael Wray. Scaling egocentric vision: The epic-kitchens dataset, 2018.

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. doi: 10.1109/CVPR.2009.5206848.

[7] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2021.

[8] Yossi Gandelsman, Yu Sun, Xinlei Chen, and Alexei A. Efros. Test-time training with masked autoencoders, 2022.

[9] Robert Geirhos, Carlos R. M. Temme, Jonas Rauber, Heiko H. Schütt, Matthias Bethge, and Felix A. Wichmann. Generalisation in humans and deep neural networks. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL `https://papers.nips.cc/paper_files/paper/2018/hash/0937fb5864ed06ffb59ae5f9b5ed67a9-Abstract.html`.

[10] Raghav Goyal, Samira Ebrahimi Kahou, Vincent Michalski, Joanna Materzyńska, Susanne Westphal, Heuna Kim, Valentin Haenel, Ingo Fruend, Peter Yianilos, Moritz Mueller-Freitag, Florian Hoppe, Christian Thurau, Ingo Bax, and Roland Memisevic. The "something something" video database for learning and evaluating visual common sense, 2017.

[11] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations, 2019.

[12] Viktor Makoviychuk, Lukasz Wawrzyniak, Yunrong Guo, Michelle Lu, Kier Storey, Miles Macklin, David Hoeller, Nikita Rudin, Arthur Allshire, Ankur Handa, and Gavriel State. Isaac gym: High performance gpu-based physics simulation for robot learning, 2021.

[13] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021.

[14] Ilija Radosavovic, Tete Xiao, Stephen James, Pieter Abbeel, Jitendra Malik, and Trevor Darrell. Real-world robot learning with masked visual pre-training. *CoRL*, 2022.

[15] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do cifar-10 classifiers generalize to cifar-10?, 2018.

[16] Nikita Rudin, David Hoeller, Philipp Reist, and Marco Hutter. Learning to walk in minutes using massively parallel deep reinforcement learning, 2022.

[17] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge, 2015.

[18] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017.

[19] Eli Schwartz. ImageNet sample images. `https://github.com/EliSchwartz/imagenet-sample-images`, 2023. Accessed: 2023-12-05.

[20] Dandan Shan, Jiaqi Geng, Michelle Shu, and David Fouhey. Understanding human hands in contact at internet scale. In *CVPR*, 2020.

[21] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2015.

[22] Aravind Srinivas, Michael Laskin, and Pieter Abbeel. Curl: Contrastive unsupervised representations for reinforcement learning, 2020.

[23] Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting unreasonable effectiveness of data in deep learning era, 2017.

[24] Yu Sun, Xiaolong Wang, Zhuang Liu, John Miller, Alexei A. Efros, and Moritz Hardt. Test-time training with self-supervision for generalization under distribution shifts, 2020.

[25] Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training, 2022.

[26] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention, 2021.

[27] Shaoru Wang, Jin Gao, Zeming Li, Xiaoqin Zhang, and Weiming Hu. A closer look at self-supervised lightweight vision transformers, 2023.

[28] Tete Xiao, Ilija Radosavovic, Trevor Darrell, and Jitendra Malik. Masked visual pre-training for motor control. *arXiv:2203.06173*, 2022.