DiffuBot: Predicting Robot Perception 50 Frames Ahead

Shi Zhan, Zheng Yingqi, Chen Xi, Gao Shang, Ma Xinxian
The Chinese University of Hong Kong, Shenzhen
{121090476,121090841,121090056, 121090133, 121090408}@link.cuhk.edu.cn

Abstract

We propose a novel diffusion-based visual-prediction model that forecasts a robot's camera view fifty frames into the future given the current image and a free-form textual action instruction. Our method fine-tunes a pretrained InstructPix2Pix architecture using simulated data generated by the RoboTwin benchmark, leveraging an encoder–decoder latent-diffusion network with cross-attention for integrating visual and language cues. We introduce an importance-sampling strategy to capture diverse motion stages, and a training process combining mixed-precision, gradient accumulation, and EMA regularization to ensure efficiency and robustness. Extensive experiments on three manipulation tasks demonstrate that our model generates accurate, coherent, and high-fidelity 128×128 future frames, offering significant improvements over baseline approaches. Our approach paves the way for safer, more anticipatory robotic perception and has broad applications in industrial automation and human–robot interaction.

1. Introduction

1.1. Background

In the realm of robotics, accurately predicting robotic actions is of paramount significance. It serves as the cornerstone for ensuring the safe and efficient operation of robots in diverse scenarios. As emphasized by Billard et al. [1], precise action prediction can significantly reduce the risk of accidents in robotic operations. For instance, in industrial environments, robots engaged in intricate tasks such as assembly or material handling can benefit immensely from accurate action forecasting. A study by Liu et al. [5] reveals that accurate prediction enables better task planning, minimizes collision risks, and enhances overall productivity. In human - robot interaction scenarios, predicting a robot's actions helps humans better understand and anticipate its behavior, fostering safer and more natural interactions, as discussed in Kanda et al.'s work [4].

However, real - world simulations for robotic action prediction are often resource - intensive. According to a report by McKinsey Global Institute [3], conducting real - world experiments for robot behavior prediction demands substantial time, cost, and physical resources. High - precision sensors, complex experimental setups, and large - scale testbeds are required to accurately capture and analyze robot actions. Moreover, real - world factors such as environmental noise, mechanical wear and tear, and variable lighting conditions can introduce uncertainties and inaccuracies into the data collection process. A case in point is the data collection for humanoid robots, as described in a research by the Institute of Electrical and Electronics Engineers (IEEE) Robotics and Automation Society [9]. Their study found that large - scale data collection for humanoid robots in real - world scenarios is not only costly but also challenging due to the need for extensive human - in - the - loop operations and specialized equipment. Therefore, virtual simulation of robot actions based on current scenes and action instructions has emerged as a crucial research area.

In virtual simulation environments, visual prediction of a robot's future behavior based on the current scene and action instructions has become a focal point of research. This project aims to train an image generation model to forecast a future frame of a robotic action. Given an observed image of the robot's current view and a textual action instruction, the model endeavors to predict what the robot will see 50 frames later. The generated images are required to have a size of at least 128x128. This research not only contributes to the advancement of robotic perception technology but also has practical applications in fields such as industrial automation, autonomous robotics, and human - robot interaction.

2. Related Work

The field of robotic action prediction has been burgeoning with research in recent years, and numerous studies have significantly influenced the direction of this project. Here are five pivotal works:

1. **Mao et al.'s work on "Learning high - level robotic manipulation actions with visual predictive model" [6]**: Mao, Chi, Ivaldi, and Chen in their 2024 study [6]

delved into the learning of visual predictive models for real - world robot manipulations. Prior research had predominantly concentrated on low - level robot actions, which often led to inefficiencies and high complexity in robot manipulation. Their novel approach aimed to address this by focusing on high - level actions like pick - and - place. They proposed a visual predictive model integrating an action decomposer and a video prediction network to capture the semantic information of high - level actions. Through experiments, they demonstrated that the model could accurately predict object dynamics during high - level pick - and - place actions. For example, in a robotic assembly task, the model could anticipate how a part would move when the robot executed a pick - and - place action, which is relevant to our project as we also seek to predict future states related to robotic actions.

2. **Zhu et al.'s work on "Video frame prediction from a single image and events" [11]**: In 2024, Zhu, Wan, and Dai [11] proposed a method for video frame prediction from a single image and subsequent events. Existing video frame prediction (VFP) methods faced challenges in handling complex dynamic scenes and predicting frames with flexible time intervals due to the limitations of fixed - frame - rate videos. Their approach introduced a symmetrical cross - modal attention augmentation module to enhance the integration of image and event information. By combining event - based motion information and image semantic information, they jointly estimated optical flow and generated frames, followed by inpainting to obtain accurate predictions. Our project, which involves predicting future frames of robotic actions, can draw inspiration from their method of leveraging different data sources (similar to how we use current images and action instructions) to improve prediction accuracy.
3. **Sarkar and Ghose's work on "Action - conditioned video generation (ACVG) framework" [10]**: Sarkar and Ghose [10] in 2024 introduced the action - conditioned video generation (ACVG) framework. Long - term video generation and prediction, especially in scenarios with moving cameras, are complex tasks due to the interaction between image frames and the recording agent's motion. The ACVG framework, through a deep dual generator - actor architecture, explored the relationship between robot actions and generated image frames. They evaluated its effectiveness on an indoor robot motion dataset. In our project, as we aim to predict future frames based on action instructions, understanding how actions condition video generation can be crucial. For instance, their work can guide us in determining how different action primitives might influence the visual appearance of future frames in our robotic action prediction task.
4. **Mao and Li's work on "Generative Adversarial Networks for Image Generation" [7]**: Mao and Li [7] provided an in - depth overview of Generative Adversarial Networks (GANs) in the context of image generation. GANs, consisting of a generator and a discriminator, have the potential to generate highly realistic synthetic data. In the field of robotics, GANs can be applied to generate realistic images, such as simulating the appearance of a robot's environment during an action. In our project, we plan to incorporate GAN - based techniques to enhance the quality of predicted images. For example, the generator in a GAN could be trained to generate the future frames of a robot's view, and the discriminator could ensure the realism of these generated frames, similar to how GANs are used in other image - generation applications described in their work.
5. **Hu et al.'s work on "Video Prediction Policy: A Generalist Robot Policy with Predictive Visual Representations" [2]**: Hu, Guo, Wang, Chen, Wang, Zhang, Sreenath, Lu, and Chen [2] in 2024 proposed the video prediction policy (VPP). They observed that previous vision encoders, trained using contrastive learning between two frames or single - frame reconstruction, failed to fully capture the sequential information crucial for embodied tasks. In contrast, video diffusion models (VDMs) showed strong capabilities in predicting future image sequences and understanding physical dynamics. The VPP used predictive visual representations from VDMs and incorporated diverse manipulation datasets for enhanced performance. In our project, when considering the use of pre - trained models like InstructPix2Pix and fine - tuning them for robotic action frame prediction, understanding how to better capture sequential and predictive visual information, as demonstrated in their work, can be beneficial for improving the performance of our model.

3. Method

Our team used a diffusion-based model to predict future robot visual states. We referred to the InstructPix2Pix framework, which leverages a latent diffusion model to generate future images conditioned on text and image inputs. Below is detailed explanation.

3.1. Model Architecture

The model is built on a pretrained InstructPix2Pix base and fine-tuned using the Stable Diffusion framework, with an encoder-decoder architecture. The VAE (Variational Autoencoder) performs as the encoder. It transforms input images into a compressed latent representation, which can make computations more efficient. The UNet-based decoder then progressively denoises this representation. The

CLIP-based frozen text encoder processes the action instruction, generating embeddings that guide the denoising process. The key to our model is the cross-attention layers within the UNet, which allow dynamic interaction between visual and textual data, enabling the model to predict a future visual state based on the given action.

3.2. Data Preparation

We construct the dataset with triplets consisting of the current image x_t, the action instruction a, and the future image 50 frames later x_{t+50}. The goal is to train the model to generate the target image x_{t+50} given x_t and a. The action instruction is natural language descriptions of the target such as 'the 50 frames later'.

3.2.1. Data Generation

For the purpose of training and testing our model, we utilize RoboTwin [8] to generate the dataset. RoboTwin is an open - source project, and its GitHub repository can be accessed at https://github.com/TianxingChen/RoboTwin. We focus on three specific tasks: `block_hammer_beat`, `block_handover`, and `blocks_stack_easy`. For each task, we generate 100 observations.

3.2.2. Data Preprocessing

The data processing pipeline is designed to prepare the generated data for model training. We use the `HeadCameraDataProcessor` class to perform the following operations:

- **Data Extraction**: From the .pkl files generated by RoboTwin, we extract the `head_camera` data. This data contains information such as the camera intrinsics (`intrinsic_cv`), extrinsics (`extrinsic_cv`), the transformation from the camera to the world coordinate system (`cam2world_gl`), and the RGB image (`rgb`).
- **Sampling**: To reduce the data size and ensure a representative sample, we use important sampling. The reason for using important sampling is to address the issue that the starting and ending actions of the robot are relatively slow. If we sample uniformly, the training results may not be as good as those in the middle part of the action sequence. Therefore, we sample 20 .pkl files from each episode according to a Gaussian distribution corresponding to time, which helps to capture different stages of the robotic actions more effectively. For each sampled file, we also include the data from the next file in the sequence to create pairs of "before" and "after" states.
- **Image Resizing and Normalization**: The RGB images are resized to 128x128, which meets the requirement of our model. Then, they are normalized by dividing the pixel values by 255.0 and rounding to four decimal places. This normalization step helps to improve the training efficiency and stability of the model.
- **Data Saving**: The processed data is saved in a .pkl file. This file contains the processed data for all the sub - datasets and episodes.
- **Conversion to JSON**: We convert the processed data to JSON format. Each JSON object contains the "before" and "after" RGB images (converted to lists) and a corresponding prompt. The prompts are designed based on the task type and are referred from `RoboTwin/data/instructions`. Specifically:
  - For the `block_hammer_beat` task (subdataset `block_hammer_beat_D435_pkl`), the prompt is: "Currently, a robot is using a hammer to strike a block placed on a table. Predict what the camera will capture in the scene 50 frames into the future. Consider factors such as the motion dynamics of the robot, the physical interactions between the hammer, the block, and the table, and any potential changes in the spatial arrangement of objects."
  - For the `block_handover` task (subdataset `block_handover_D435_pkl`), the prompt is: "Currently, the robot is using arm movement to transfer a block to a handover point. Based on the present scene, predict what the camera will capture in the scene 50 frames later, considering the typical steps of the block hand - over process such as the transfer between arms and the movement towards the target."
  - For the `blocks_stack_easy` task (subdataset `blocks_stack_easy_D435_pkl`), the prompt is: "Currently, the robot is in the process of placing the black block and red block. Based on the present scene, predict what the camera will capture in the scene 30 frames later. Consider the relative positions of the two blocks, the movement speed of the robot's arm while handling the blocks, and any possible adjustments during the placement."

The final dataset is split into training and testing subsets at an 8:2 ratio and uploaded to Hugging Face at Aurora1609/RoboTwin.

3.3. Training Procedure

The training procedure allows for distributed training, which can enhance scalability. The total batch size is computed based on the number of devices, and a checkpoint can make training resume if interrupted. During training, The VAE encodes the images into latent space and Gaussian noise is added to the latent representation of the images, while the UNet gradually removes the noise during the denoising process. The model receives dual conditioning from both the action instructions encoded by a text encoder CLIP and the current image, allowing it to learn the relationship between these inputs.

In forward pass, we concatenated the noisy latent representation with the encoded image embedding, then using a prediction type (either epsilon or velocity) to calculate the target for the loss function. The prediction is compared to the target using MSE loss to train the model. The optimizer and learning rate scheduler are updated during backward propagation, while gradient accumulation is employed to manage memory usage efficiently.

Periodically, the model is validated using the EMA parameters, and results are logged for evaluation. Checkpoints are saved at regular intervals to ensure progress is preserved, and the model can be recovered if necessary. At the end of training, the final model is saved, and the trained pipeline is packaged.

3.4. Regularization and Optimization

We used a combination of mixed-precision, gradient accumulation, and Exponential Moving Average (EMA) of model weights to ensure stable training. Mixed-precision reduces memory usage and speeds up computation, while gradient accumulation allows us to simulate larger batch sizes within GPU memory limits. The EMA strategy controls a smoothed version of parameters, which can improve robustness. For optimization, we employed the Adam optimizer with a cosine learning rate decay schedule. It can help reach a steady convergence.

3.5. Inference

During inference, the model takes an observation image from the robot and a textual description of the task. The image is first encoded into latent space using VAE. Then, the model iteratively denoises the latent through the learned diffusion process. After completing the denoising steps, the final latent is decoded back into pixel space to produce an image. The result is the robot's predicted visual state approximately 50 frames into the future.

4. Experiment

4.1. Training Platform and Parameters

We conducted our experiments using the following key parameters:

```
--pretrained_model_name_or_path=
  timbrooks/instruct-pix2pix \
--dataset_name=Aurora1609/RoboTwin \
--enable_xformers_memory
  _efficient_attention \
--resolution=128
--random_flip \
--train_batch_size=64 \
--gradient_accumulation_steps=4 \
--gradient_checkpointing \
--max_train_steps=300 \
--checkpointing_steps=50 \
--checkpoints_total_limit=1 \
--learning_rate=5e-05
--max_grad_norm=1 \
--lr_warmup_steps=20 \
--conditioning_dropout_prob=0.05 \
--mixed_precision=fp16 \
--seed=42 \
--push_to_hub \
--report_to=tensorboard \
--original_image_column=before \
--edit_prompt=prompt \
--edited_image=after
```

Here are the key parameters for our training procedure:

- **Pretrained Model**: We use the pretrained InstructPix2Pix model from the Hugging Face Hub (`timbrooks/instruct-pix2pix`).
- **Dataset**: The dataset used for training is `Aurora1609/RoboTwin`, which contains robot images and action instructions.
- **Resolution**: The input image resolution is set to 128x128 pixels.
- **Batch Size**: We use a batch size of 64, with gradient accumulation of 4 steps to simulate a larger batch size.
- **Learning Rate**: The learning rate is set to 5e-05, with a warm-up period of 20 steps.
- **Precision**: We use mixed precision (fp16) for efficient training.
- **Checkpointing**: Checkpoints are saved every 50 steps, with a limit of 1 total checkpoint.
- **Training Steps**: The model is trained for a maximum of 300 steps.
- **Random Flip**: Random flipping of images is enabled during training.
- **Conditioning Dropout**: A dropout probability of 0.05 is applied to the conditioning signal to regularize the model.

We fine-tuned the model in the CUDA 12.4 environment with Python 3.10 using an NVIDIA A100 Tensor Core GPU with 80 GB of graphics memory. Some other information of the related libraries are also attached as follows:

- torch 2.6.0
- torchaudio 2.6.0
- torchvision 0.21.0
- transformers 4.51.3

4.2. Evaluation Metrics

For evaluating our model's performance, we use two key metrics: Structural Similarity Index (SSIM) and Peak Signal-to-Noise Ratio (PSNR). These metrics are used to evaluate the similarity between the predicted images and the ground-truth images, and the average values over all data

samples are computed as the final evaluation of the model's performance.

SSIM is a perceptual metric that quantifies the similarity between two images. It compares luminance, contrast, and structure between the reference image and the generated image, with values ranging from -1 to 1. A higher SSIM value indicates better structural similarity between the two images. This metric is particularly sensitive to structural distortions and is commonly used to assess the quality of generated images in tasks such as image generation and restoration.

PSNR is another image quality metric that measures the ratio of the peak signal power to the noise power. Higher PSNR values indicate less distortion between the generated and reference images. PSNR is commonly used to evaluate image compression, denoising, and generative models, and it provides a quantitative measure of the overall quality of the predicted image.

In our experiments, both SSIM and PSNR are computed between the predicted image and the ground-truth image for each data sample, and the average values over all samples are used as the final evaluation of the model's performance.

4.3. Results

Figure 1 shows the training loss curve of our model.

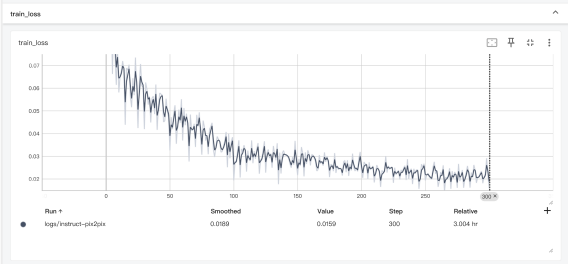


Figure 1. Training loss curve.

Then, we present the results of our baseline model and the fine-tuned model. The evaluation results are summarized in the following table. From the table, it is clear that fine-tuning significantly improved both SSIM and PSNR, indicating a substantial enhancement in the quality of the generated images.

Model	Mean SSIM	Mean PSNR
Baseline	0.6161	11.7397
Fine-Tuned	0.8629	21.2033

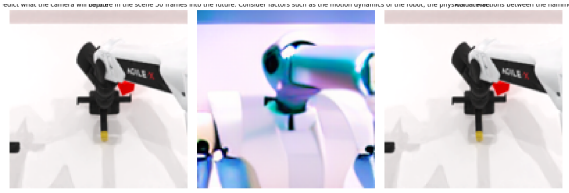
Table 1. Evaluation results for baseline and fine-tuned models.

4.4. Visual Comparison

We now present some visual results comparing the baseline model and the fine-tuned model. Figure 2 and Figure 3 present side-by-side examples of our baseline and fine-tuned models, respectively. The three images are displayed from left to right: the first image is the baseline model's output (before), the second is the model's prediction after fine-tuning, and the third is the ground truth.

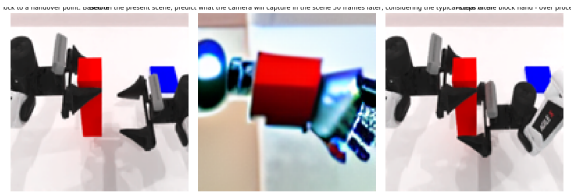
4.5. Baseline Model

Figure 2(a)–(c) illustrate three representative failure cases from baseline. In Fig. 2a, the predicted frame shows a severely blurred gripper, indicating the model cannot preserve sharp object boundaries over 50 frames. In Fig. 2b, the output suffers from ghost-artifacts around the robot arm, and the small red cube is mislocalized that the baseline struggles to maintain spatial coherence under long-horizon motion. Finally, Fig. 2c reveals a complete distortion of both block and gripper geometry, underscoring the baseline's inability to capture multi-object interactions and scene dynamics at extended time steps. Overall, these examples demonstrate that, without task-specific fine-tuning, the model produces low-fidelity predictions that lack clarity, correct positioning, and coherent visual structure. This could be due to the model's inability to fully understand the logic of predicting a visual state 50 frames into the future, as well as possible issues with the quality of the input images.



(a) Baseline example 1

(b) Baseline example 2

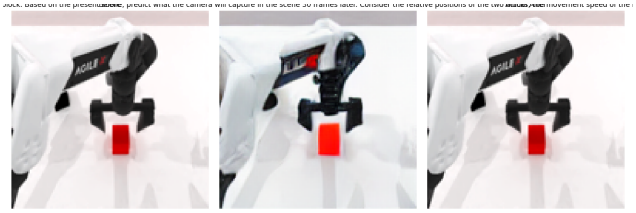


(c) Baseline example 3

Figure 2. Visual comparison of the baseline model on different examples.

4.5.1. Fine-Tuned Model

In contrast, the fine-tuned model shows significantly improved results. Figure 3(a)–(c) showcases three representative cases after fine-tuning. In Fig. 3a, the red block is rendered with crisp edges and sits precisely under the gripper, matching the ground truth scale and position without any blur. In Fig. 3b, the block's orientation and its subtle shadow on the white table are accurately reproduced, and the robot arm's fingers align perfectly with the expected pose. Finally, Fig. 3c demonstrates faithful capture of the gripper's tilt and the slight lateral shift of the block, reflecting realistic motion dynamics. These examples confirm that fine-tuning restores object clarity, spatial coherence, and dynamic consistency over a 50-frame horizon. This improvement highlights the effectiveness of fine-tuning the model on the specific dataset.



(a) Fine-Tuned example 1

(b) Fine-Tuned example 2

(c) Fine-Tuned example 3

Figure 3. Visual comparison of the fine-tuned model on different examples.

5. Conclusion

We presented a diffusion-based visual prediction model that forecasts robot camera views fifty frames into the future. We adapted and fine-tuned a pretrained InstructPix2Pix architecture using simulated data generated by RoboTwin. We introduced an importance-sampling strategy, mixed-precision training, and EMA-based regularization to improve training efficiency and robustness. Experimental results demonstrate that our model produces accurate and coherent future frames conditioned on both current images and action instructions. Our approach holds promise for enhancing safety, planning, and human–robot interaction in real-world robotic systems. In future work, we will explore higher-resolution predictions, a broader range of tasks, and deployment on physical robots.

References

[1] Aude Billard, Stefan Schaal, Andrew J. I. Barron, and Sebastian Thrun. Robot learning from demonstration: An overview. *Foundations and Trends in Robotics*, 2(3):197–273, 2008. 1

[2] Yucheng Hu, Yan Jiang Guo, Peng Chao Wang, Xiaoyu Chen, Yen jen Wang, Jianke Zhang, Koushil Sreenath, Chaochao Lu, and Jianyu Chen. Video prediction policy: A generalist robot policy with predictive visual representations. *arXiv preprint arXiv:2412.14803*, 2024. 2

[3] McKinsey Global Institute. The future of work in the age of ai, 2023. 1

[4] Tetsuya Kanda, Hiroshi Ishiguro, and Norihiro Hagita. Human - robot interaction in social robotics. *Proceedings of the IEEE*, 95(9):1389–1406, 2007. 1

[5] Ying Liu, Wenqiang Zhang, Bo Zhang, and Bin Jiang. Improving industrial robot safety and efficiency through predictive maintenance. *Journal of Manufacturing Systems*, 55: 193–203, 2020. 1

[6] Anji Mao, Guoyi Chi, Serena Ivaldi, and Lipe Ng Chen. Learning high - level robotic manipulation actions with visual predictive model. *Complex & Intelligent Systems*, 10 (2):811–823, 2024. 1

[7] Xudong Mao and Qing Li. *Generative adversarial networks for image generation*. Springer, 2021. 2

[8] Yao Mu, Tianxing Chen, Zanxin Chen, Shijia Peng, Zhiqian Lan, Zeyu Gao, Zhixuan Liang, Qiaojun Yu, Yude Zou, Mingkun Xu, Lunkai Lin, Zhiqiang Xie, Mingyu Ding, and Ping Luo. Robotwin: Dual-arm robot benchmark with generative digital twins, 2025. 3

[9] Institute of Electrical, Electronics Engineers (IEEE) Robotics, and Automation Society. Challenges in real - world data collection for humanoid robots. *IEEE Robotics and Automation Magazine*, 30(2):45–53, 2022. 1

[10] Meenakshi Sarkar and Debasish Ghose. Action - conditioned video data improves predictability. *arXiv preprint arXiv:2404.05439*, 2024. 2

[11] Juanjuan Zhu, Zhexiong Wan, and Yuchao Dai. Video frame prediction from a single image and events. In *AAAI Conference on Artificial Intelligence*, 2024. 2