FLIP: Flow-Centric Generative Planning for General-Purpose Manipulation Tasks

Chongkai Gao
National University of Singapore
gaochongkai@u.nus.edu

Haozhuo Zhang
Peking University
2100013132@stu.pku.edu.cn

Zhixuan Xu
National University of Singapore
zhixuanxu@u.nus.edu

Zhehao Cai
National University of Singapore
e1373791@u.nus.edu

Lin Shao
National University of Singapore
linshao@u.nus.edu

Abstract: We aim to develop a model-based planning framework for world models that can be scaled with increasing model and data budgets for general-purpose manipulation tasks with only language and vision inputs. To this end, we present FLow-CentrIc generative Planning (FLIP), a model-based planning algorithm on visual space that features three key modules: 1) a multi-modal flow generation model as the general-purpose action proposal module; 2) a flow-conditioned video generation model as the dynamics module; and 3) a vision-language representation learning model as the value module. Given an initial image and language instruction as the goal, FLIP can progressively search for long-horizon flow and video plans that maximize the discounted return to accomplish the task. FLIP is able to synthesize long-horizon plans across objects, robots, and tasks with image flows as the general action representation, and the dense flow information also provides rich guidance for long-horizon video generation. In addition, the synthesized flow and video plans can guide the training of low-level control policies for robot execution. Experiments on diverse benchmarks demonstrate that FLIP can improve both the success rates and quality of long-horizon video plan synthesis and has the interactive world model property, opening up wider applications for future works. Video demos are on our website: https://flow-planning.github.io/.

Keywords: World Model, Planning, Video Generation, Manipulation

1 Introduction

World models refer to neural network-based representations or models that learn to simulate the environment [1, 2]. With world models, agents can imagine, reason, and plan inside world models to solve tasks more safely and efficiently. Recent advancements in generative models, especially in the area of video generation [3, 4, 5], have demonstrated the application of generating high-quality videos as world simulators with internet-scale training data. World models have opened new avenues across various fields, particularly in the domain of robotic manipulation [5, 6, 7], which is the focus of this paper.

The intelligence of generalist robots involves two levels of abilities [8, 9]: 1) high-level planning of the abstraction sequence of the task with multi-modal inputs, and 2) low-level execution of the plan by interacting with the real world. A well-designed world model could serve as an ideal way to realize the first function, for which it should enable model-based planning. This requires the world model to be interactive, i.e., can simulate the world according to some given actions. The core of this

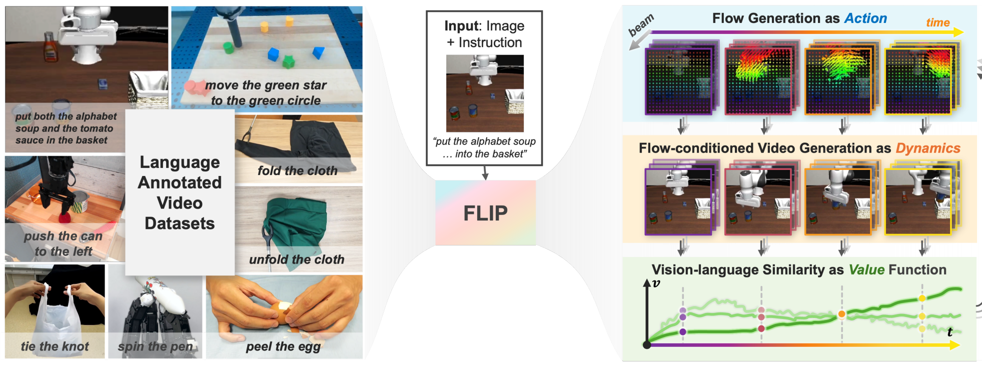

put both the alphabet soup and the tomato sauce in the basket
move the green star to the green circle
Language Annotated Video Datasets
fold the cloth
push the can to the left
unfold the cloth
tie the knot
spin the pen
peel the egg
Input: Image + Instruction
"put the alphabet soup … into the basket"
FLIP
Flow Generation as Action
beam
time
Flow-conditioned Video Generation as Dynamics
Vision-language Similarity as Value Function
v
t
Iterative Planning

Figure 1: Overview of our method. Left: FLIP is trained on video datasets across different tasks, objects, and robots, with only one language description for each video as the goal. Right: we train an interactive world model consisting of an action module for flow generation, a dynamics module for video generation, and a value module for assigning value at each step. These modules can perform flow-centric model-based planning for manipulation tasks on the flow and video space.

framework is to find a *scalable action representation* that serves as the connection between high-level planning and low-level control. This representation should: 1) be able to represent various kinds of movements across diverse objects, robots, and tasks in the whole scene; 2) be easy to obtain or label a large amount of training data for scaling up. Regarding this, Yang et al. [5], Du et al. [10], Zhou et al. [11] use languages from VLMs [12] as high-level actions, while Wu et al. [13] directly use low-level robot actions to interact with the world model. However, they either require extra datasets or task-specific high-level action labeling processes for training the interactive world model, or their representations cannot describe sophisticated subtle movements in the whole scene. For example, they cannot describe the detailed movements of a dexterous hand spinning a pen. These limit their application as a scalable interactive world model and inspire us to find other action representations.

Image flow, a dynamic representation of pixel-level changes over time, is a concise yet general representation of all kinds of movements in images for different robots and objects and can describe more subtle changes than language. More importantly, image flow can be completely obtained by off-the-shelf trackers [14] from pure video datasets. Meanwhile, recent works also show that flows are effective representations for training low-level manipulation policies [15, 16, 17]. These make flow a good choice for action representation of world models. However, it remains unclear how to leverage flows for planning on manipulation tasks.

In this work, we present Flow-Centric General Planning (FLIP) for general-purpose robot manipulation tasks. As shown in Figure 1, we train a flow-centric world model purely from language-annotated video data from diverse tasks. This world model contains three modules: 1) a flow generation network as the action module, 2) a flow-conditioned video generation model as the dynamics module, and 3) a visual-language representation learning model as the value module. Specifically, we design our action and dynamics module based on CVAE [18] and DiT [19] architectures respectively and propose a new training mechanism for leveraging LIV [20] as our value module. The trained modules enable model-based planning by progressively searching successful long-horizon plans on the flow and video spaces: given an initial image and language instruction as the goal, the action module will propose several flow candidates, and the dynamics module will generate the short-horizon future videos. The value module will access the favorability of generated videos that maximize the discounted returns and perform tree search [21] to synthesize long-horizon plans for solving the task.

In experiments, we show that FLIP can perform model-based planning to solve tasks for both simulation manipulation tasks (LIBERO [22]) and real-world tasks (including FMB [23], cloth folding, unfolding, and Bridge-V2 [24]). We also show that FLIP can generate high-quality long-horizon videos for these tasks. Meanwhile, the generated flow and video plans can guide the training of low-level policies. We also show that the three modules of FLIP are superior to their respective baselines [15, 25, 20]. We quantitatively show that FLIP can simulate diverse complex manipulation tasks across objects and robots. The trained world model also demonstrates interactive properties, zero-shot transfer ability, and scalability. In summary, our contributions are:

- We propose flow-centric generative planning (FLIP) as an interactive world model for general-purpose model-based planning for manipulation tasks.
- We design a new flow generation network, a new flow-conditioned video generation network, and a new training method for an existing vision-language representation learning network as the three key modules of FLIP.
- In our experiments, we show FLIP can perform general-purpose model-based planning, synthesize long-horizon videos, guide the training of low-level policy, and other promising properties, as well as the superiority of the three modules of FLIP compared to baselines.

2 Related Works

2.1 World Models for Decision Making

Early works of world models learn system dynamics in low dimensional state space [26, 27], perform planning in latent space [28], or train networks to predict the future observations [29] and actions [30]. Modern model-based reinforcement learning methods [31, 32, 33, 34, 35] focus on latent space imagination with coupled dynamics and action modules. Recent works leverage powerful scalable video generation architectures like Diffusion Transformer [19] and large-scale training data [36] to develop video generation networks to simulate an interactive environment [5, 37, 38, 39, 13, 25, 13]. In this work, we build a world model with separate flow-centric action and dynamics modules as well as a vision-language value model for model-based planning for robot manipulation tasks.

2.2 Flow and Video Models for Manipulation

Flows are the future trajectories of query points on images or point clouds. They are universal descriptors for motions in the video, while video data contains rich knowledge of behaviors, physics, and semantics, and have unparalleled scalability in terms of both content diversity and data acquisition. For robotics, people have been trying to use flows as policy guidance [15, 40], learn dense correspondence [41], tool using [42], or cross-embodiment representations [16, 25, 43]. Videos are usually used for learning inverse dynamics [44, 29, 45, 46], rewards [47, 20, 48, 49], transferrable visual representations such as latent embeddings [50, 48, 22], key points [51, 52], affordance [53, 54], flows [15, 16, 40], scene graphs [55, 56, 57], or acquire similar manipulation knowledge from human videos [58, 6, 59, 60]. Recent works also use video generation techniques as visual simulation [5, 61]. In this work, we build our action, dynamics, and value modules all based on video and language inputs, enabling the scalability of our framework.

3 Three Fundamental Modules of FLIP

3.1 Problem Formulation

We model a manipulation task $\mathcal{T}$ as a goal-conditioned Partially Observable Markov Decision Process (POMDP) parameterized by $(\mathcal{S}, \mathcal{O}, \phi, \mathcal{A}, P, R, \gamma, g)$ where $\mathcal{S}, \mathcal{A}, \mathcal{O}$ are state, action, and observation spaces, $\phi : \mathcal{S} \to \mathcal{O}$ is the state-observation mapping function, $P : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ is the transition function, $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, γ is the discount factor, and g is the goal state. In this work, the observation space is the image space: $\mathcal{O} = \mathbb{R}^{H \times W \times 3}$, where H and W are the height and width of the image, and $R(s, g) = \mathbb{I}(s == g) - 1$ is a goal-conditioned sparse reward. The task is solved if the agent maximizes the return $\sum_{t=0}^{T} \gamma^t R(s_t, g)$.

We aim to solve this problem by learning a world model and a low-level policy. The world model performs model-based planning on image and flow spaces to maximize the return, synthesizing long-horizon plans, and the low-level policy executes the plan in the real environment. We aim to train

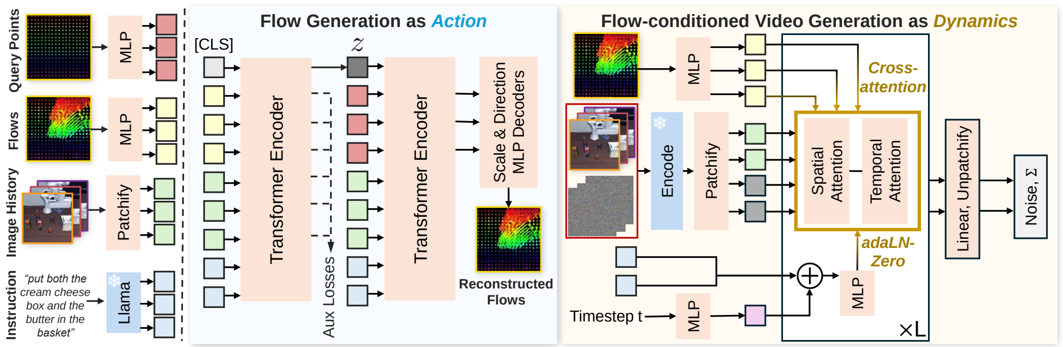


Figure 2: The action module and dynamics module of FLIP. Left: the tokenizing process of different modalities in training data. Middle: we use a Conditional VAE to generate flows as actions. It separately generates the delta scale and directions on each query point for flow reconstruction. Right: we use a DiT model with the spatial-temporal attention mechanism for flow-conditioned video generation. Flows (and observation history) are conditioned with cross attention, while languages and timestep are conditioned with AdaLN-zero.

the world model only on language-annotated video datasets to make it general-purpose and scalable, and train the low-level policy on a few action-labeled datasets. To enable model-based planning, our world model contains three key modules, as introduced in the following sections.

3.2 Flow Generation Network as Action Module

Overview. The action module of FLIP is a flow generation network π_f that generates image flows (future trajectories on query points) as *actions* for planning. The reason why we use a generation model rather than a predictive model is that we are doing model-based planning, where the action module should give different action proposals for sampling-based planning. Formally, given h step image observation history $o_{t-h:t}$ at timestep t, the language goal g, and a set of 2D query points coordinates $\mathbf{p}_t = \{p_t^k\}_{k=1}^K$, where $p_{t,k} = (u, v)$ is the k-th query point coordinate on o_t, the flow generation network π_f generates coordinates of query points in future L timesteps (including the current step): $\mathbf{p}_{t:t+L} = \pi_f(o_{t-h:t}, \mathbf{p}_t, g) \in \mathbb{R}^{L \times K \times 2}$.

Training Data Annotation. The flows of query points can be extracted from pure video data by the off-the-shelf point tracking models. The problem is how to select query points. Previous works either use SAM [62] to select query points on the region of interest or select query points on active/stationary regions with a predefined ratio [15]. These methods face two problems: 1) for diverse kinds of videos with complex scenes, it is hard for modern segmentation models [62] to segment perfect regions of interest with no human assistance; 2) for long-horizon videos, there may be objects appearing/disappearing in the video, and using query points only from the initial frame become problematic. To this end, in this work, we uniformly sample dense grid query points for the whole image (for the first problem) at each timestep, and track them for only a short-horizon video clip, i.e., tracking on video clips starting from *every* frame of the long-horizon video (for the second problem). This can mitigate the second problem because even if some objects appear/disappear, their influences are restricted in a short horizon. Formally, for each frame in the dataset, we uniformly sample a grid of N_q points, then use Co-Tracker [14] to generate the flows $\{p_{t:t+L}^k\}_{k=1}^{N_q}$ within a future video clip of L steps.

Model Design. We design a Conditional VAE [18] with transformer [63] architecture for flow generation, as illustrated in Figure 2. As opposed to previous flow prediction works [15, 16, 40], we observe enhanced performance when predicting relative displacements rather than absolute coordinates, i.e., we predict $\Delta p_t^k = p_{t+1}^k - p_t^k$ for the k-th point at each time step.

For the VAE encoder, we encode ground truth flow $\{\mathbf{p}_t\}_{t=1}^L$, patchify observation history $o_{t-h:t}$, and encode language embedding from Llama 3.1 8B [64] to tokens, concatenate them with a *CLS* token for gathering the information, and then send them to a transformer encoder to extract the output at the *CLS* token position as the latent variable of VAE. For the VAE decoder, we first encode the query points $\{p_t^k\}_{k=1}^{N_q}$ at only timestep t to query tokens, concatenate them with image and language tokens as well as the sampled latent variable z from reparameterization, and send them to another transformer encoder. We extract the output at the query tokens and use two MLPs to predict the delta scale $\delta_s \in \mathbb{R}_{\geq 0}^{L \times K}$and delta direction $\vec{\delta_d} \in \mathbb{R}^{2L \times K}$ for L future horizons. Thus we can get $\Delta p_t^k = \delta_s^{tk} \vec{\delta_d^{tk}}$, and the whole future flow can be reconstructed step by step. We also decode the output at the image token positions as an auxiliary image reconstruction task [15, 65], which we find useful for improving the training accuracy.

3.3 Flow-Conditioned Video Generation Network as Dynamics Module

Overview. The flow-conditioned video generation network $\mathcal{D}$ generates the following L frames based on current image observation history $o_{t-h:t}$, the language goal g, and the predicted flow $\mathbf{p}_{t:t+L}$ to enable iterative planning for the next planning step: $\hat{o}_{t+1:t+L} = \mathcal{D}(o_{t-h:t}, g, \mathbf{p}_{t:t+L})$.

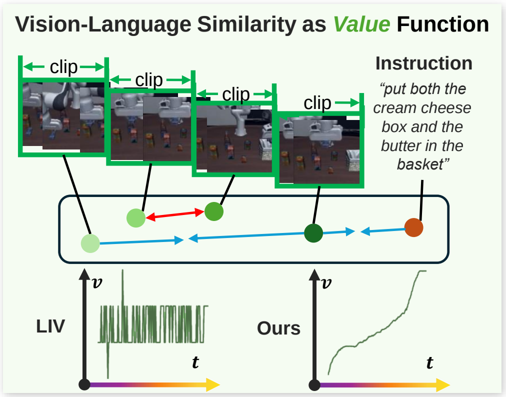
Model Design. We design a new latent video diffusion model that can effectively take as input different kinds of conditions such as images, flows, and language. This model is built on the DiT [19] architecture with spatial-temporal attention mechanism [66, 37, 25]. The background knowledge of latent video diffusion models is in Appendix A.1. Here we introduce the design of the multi-modal condition mechanism.

In the original DiT [19] and previous trajectory-conditional video diffusion paper [25], they use adaptive layer norm zero (AdaLN-Zero) blocks to process conditional inputs (such as diffusion timestep and class labels), which regress the scale and shift parameters of the layer norm layers from all conditions with a zero-initialized MLP. However, AdaLN will compress all conditional information to scalars, and cannot enable fine-grained interaction between different parts of conditions with the inputs. Thus, this mechanism is not suitable for complex conditions such as image and flow [67, 68]. To this end, we propose a mixed conditioning mechanism for multi-modal conditional generation. We use cross attention for fine-grained interactions between flow conditions (tokenized as N_q tokens) and observation conditions and noisy frames. For image history conditions, we concatenate them on the Gaussian noise frames. We use AdaLN-Zero to process the global conditions including the diffusion timestep and language instruction, as shown in Figure 2. To keep the observation condition clean, we do not add noise to $o_{t-h:t}$ during the diffusion process and do not perform denoising on them either.

3.4 Vision-Language Representation Learning as Value Module

Overview. The value module $\mathcal{V}$ assigns an estimated value $\hat{V}_t$ for each frame o_t to enable model-based planning on the image space, based on the language goal g: $\hat{V}_t = \mathcal{V}(o_t, g)$. In this work, we adopt LIV [20] to instantiate the value function. LIV first learns a shared language-vision representation from action-free videos with language annotations. It then computes the similarity between current frame o_t and g as the value for timestep t: $\hat{V}_t = \mathcal{S}(\psi_I(o_t), \psi_L(g)) = \frac{1}{1-\gamma} cos(\psi_I(o_t), \psi_L(g))$, where ψ_I and ψ_L are the encoding network for image and language respectively, and $\mathcal{S}$ is the γ-weighted cosine similarity metric.

The pretrained LIV model needs to be fine-tuned to give good value representation on new tasks [20]. The original fine-tuning


Vision-Language Similarity as Value Function
clip
clip
clip
clip
Instruction
"put both the cream cheese box and the butter in the basket"
LIV
Ours
v
t
v
t
Figure 3: Top: The value module of FLIP. We follow the idea of [20] and use time-contrastive learning for the visual-language representation, but we treat each video clip (rather than each frame) as a state. Bottom: the fine-tuned value curves of Ma et al. [20] and ours.

Algorithm 1 Flow-Centric Generative Planning

```
1: Input: Current observation history $o_{0-h:0}$, language goal $g$, query points $\mathbf{p}$, flow prediction
   network $\pi_f$, flow-conditioned video generation network $\mathcal{D}$, vision-language value module $\mathcal{V}$.
2: Hyperparameters: Flow candidates number $A$, planning Beams $B$, planning horizon $H$.
3: Initialization: Flow plans $f_p \leftarrow [\mathbf{p}^i_{0-h:0}, i \in 1 \dots B]$, video plans $v_p \leftarrow [o^i_{0-h:0}, i \in 1 \dots B]$.
4: for $h = 1 \dots H$ do
5:     for $b = 1 \dots B$ do
6:         $o \leftarrow v_p[b][-h:]$                              ▷ Get the Latest Observation History in the Plan Beam
7:         $a_{1:A} \leftarrow \pi(o, \mathbf{P}, g)$                          ▷ Generate A Different Flow Actions
8:         $v_{1:A} \leftarrow \mathcal{D}(o, a_i, g)$ for $i$ in$(1 \dots A)$   ▷ Generate Corresponding Different Video Clips
9:         $id \leftarrow \text{argmax}\, \mathcal{R}(v_i, g)$ for i in (1 ... A)
10:        $f_p[b].\text{append}(a_{id}), v_p[b].\text{append}(v_{id})$          ▷ Add Plans with Highest Value
11:    end for
12:    $max\_idx, min\_idx \leftarrow \text{argmax}(v_p, \mathcal{V}), \text{argmin}(v_p, \mathcal{V})$
13:    $f_p[min\_idx] \leftarrow f_p[max\_idx], v_p[min\_idx] \leftarrow v_p[max\_idx]$        ▷ Periodically Replace
14: end for
15: $f \leftarrow f_p[\text{argmax}(v_p, \mathcal{V})], v \leftarrow v_p[\text{argmax}(v_p, \mathcal{V})]$        ▷ Return Highest Value Plan
```

loss $\mathcal{L}_{LIV} = \mathcal{L}_I(\psi_I) + \mathcal{L}_L(\psi_I, \psi_L)$ is calculated on sampled sub-trajectory batch data $\left\{o^i_s, \dots, o^i_k, o^i_{k+1}, \dots, o^i_T, g^i\right\}_{i=1}^{B}$ from each task $\mathcal{T}_i$, where $s \in [0, T_i - 1], s \leq k < T_i$. For $\forall$ task i, $\mathcal{L}_I(\psi_I)$ will use time contrastive learning to increase the similarity $\mathcal{S}(o^i_s, o^i_T)$ between the sampled start frame and the end frame and keep the embedding distance between two adjacent frames as (γ-discounted) 1, and $\mathcal{L}_L$ encourages the image goal o^i_T and language goal g^i have the same embedding for the same task i. Details of this process can be found in Appendix A.2.

Finetuning LIV on Long-Horizon Imperfect Videos. Finetuning LIV with the original training objective works well on short-horizon perfect videos (about 50 frames in their original papers [47, 20]). However, we find that it does not work well for our long-horizon imperfect videos, as shown in Figure 3, where the fine-tuned value curve exhibits numerous jagged peaks. This is disastrous for sampling-based planning algorithms since most planning algorithms expect a smoothing value curve to be effective [21, 20].

We point out that this problem is caused by imperfect long-horizon videos, where the task does not necessarily progress smoothly as the video progresses. For example, the robot arm may hesitate in the air during the task. To mitigate this problem, we replace the concept of *adjacent frames* in the original loss to *adjacent states*, where we define states as short-horizon video clips. Formally, we divide a long-horizon video into small segments of fixed length and treat each clip s^{clip} as the smallest unit of the video. The original o_s, o_T, o_k, o_{k+1} are seamlessly replaced by $s^{clip}_s, s^{clip}_T, s^{clip}_k, s^{clip}_{k+1}$ respectively. As shown in Figure 3, this simple strategy is surprisingly useful and makes the fine-tuned vale curve much smoother than the originally fine-tuned ones.

4 Flow-Centric Generative Planning

4.1 Model-based Planning with Flows, Videos, and Value Functions

Directly generating long-horizon videos autoregressively is usually not accurate [15, 5, 44] due to compounding errors. In this work, we use model-based planning to search for a sequence of flow actions and video plans that maximizes the discounted return:

$$o^*_{0:L} = \underset{o_{0:L} \sim \pi_f, \mathcal{D}}{\arg\max} \sum_{i=0}^{L} \gamma^i R(o_i, g). \tag{1}$$

According to Bellman Equation [69], this equals stepping towards the next state that maximizes $V^*(s_{t+1}, g)$ at each time step given an optimal value function V^*. In our problem, we learn the optimal value function $\mathcal{V} = V^*$ through our proposed method, thus our problem is simplified to

	LIBERO-LONG [22]	FMB-S [23]	FMB-M	Folding	Unfolding
UniPi [44]	2%	0%	0%	20%	10%
FLIP-NV	78%	52%	40%	**100%**	70%
FLIP(Ours)	**100%**	**86%**	**78%**	**100%**	**90%**

Table 1: Success rates of model-based planning on long-horizon tasks.

	LIBERO-LONG [22]			FMB [23]			Bridge-V2 [24]		
	Latent L2 ↓	FVD ↓	PSNR ↑	Latent L2 ↓	FVD ↓	PSNR ↑	Latent L2 ↓	FVD ↓	PSNR ↑
LVDM [70]	0.566	610.98	10.852	0.484	358.22	12.349	0.373	153.41	16.481
IRASim [25]	0.407	206.28	12.205	0.395	172.45	13.157	0.325	138.97	16.796
FLIP(Ours)	**0.217**	**35.62**	**26.452**	**0.264**	**43.712**	**25.531**	**0.173**	**36.15**	**33.485**

Table 2: Quantitative results on long-horizon video generation.

search for the next state that maximizes $\mathcal{V}$ at each time step. Note this reward design also encourages finding the shortest plan. We use hill climbing [21] to solve this problem. It initializes B plan beams. At each timestep t, given current image history $o_{t-h:t}$ and the language goal g, it employs π_f to generate multiple flow actions $\mathbf{p}_{t+1:t+L} = \pi_f(o_{t-h:t}, \mathbf{p}_t, g)$ on uniformly sampled query points as candidates for tree search, then use $\mathcal{D}$ to generate corresponding short-horizon videos $o_{t+\hat{1}:t+L} = \mathcal{D}(o_{t-h:t}, g, \mathbf{p}_{t+1:t+L})$. The value module $\mathcal{V}$ is then used to select the generated video with the highest reward among A videos to enable the next iteration of generation for each beam. In order to prevent exploitative planning routes that over-exploit on an irregular state, we periodically replace the lowest value plan among the beams with the beam with the highest value. The algorithm is summarized in Algorithm 1.

4.2 Plan-Conditioned Low-Level Policy

The low-level policy π_L takes as input the image observation history $o_{t-h:t}$, the language goal g, and the predicted flow plan $\mathbf{p}_{t:t+L}$ as well as the video plan $o_{t+\hat{1}:t+L} = \mathcal{D}(o_{t-h:t}, g, \mathbf{p}_{t+1:t+L})$ to predict the low-level robot action $a_{t:t+L}$ that actually drive the robot to operate in the environment. We train this policy through imitation learning on a few demonstrations with action labels. We employ a spatial-temporal transformer to instantiate π_L with a similar structure to Wen et al. [15]. Details can be found in Appendix A.3.

5 Experiments

In this section, we first demonstrate that FLIP can: 1) perform model-based planning for different manipulation tasks; 2) synthesize long-horizon videos ($\geq$ 200 frames); and 3) can guide the low-level policy for executing the plan. We also evaluate the action, dynamics, and value modules separately compared to corresponding baselines and show the interactive, zero-shot, scalability properties of FLIP. More details of experiment settings and results can be found in Appendix C and our website.

5.1 Model-Based Planning for Manipulation Tasks

Setup. In this section, we train FLIP on four benchmarks to show its model-based planning ability. The model is given an initial image and a language instruction, and it is required to search the flow and video spaces to synthesize the plan for this task. The first one is LIBERO-LONG [22], a long-horizon table-top manipulation benchmark of 10 tasks in simulation. We train FLIP on 50×10 long-horizon videos with a resolution of $128 \times 128 \times 3$ and test on 50×10 new random initializations. The second one is the FMB benchmark [23], a long-horizon object manipulation and assembly benchmark with varying object shapes and appearances. We train FLIP on 1K single-object multi-stage videos and 100 multi-object multi-stage videos with a resolution of $128 \times 128 \times 3$ and test on 50 new initialization for each. The third and fourth suites are cloth folding and cloth unfolding. These two datasets are collected by ourselves. We train each suite on 40 videos with varying viewpoints and test on 10 new viewpoints for each with a resolution of $96 \times 128 \times 3$.

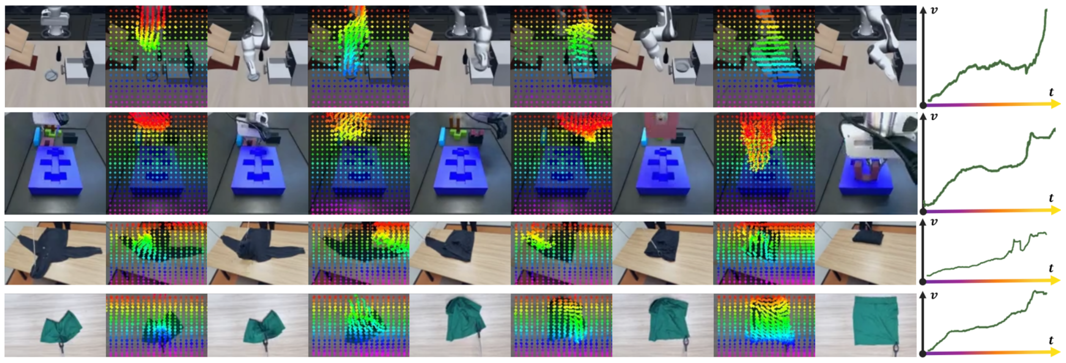


Figure 4: Model-based planning results on LIBERO-LONG, FMB, cloth folding, and cloth unfolding. All of the flows, images, and values shown are generated by FLIP.

We follow previous works[10, 25] and evaluate our model-based planning results by human evaluating the correctness of generated video plans. That is, we visually assess the percentage of time the video successfully solved the given task. We compare FLIP to two baselines: 1) UniPi [44], a text-to-video generation method with long-horizon text goals. 2) FLIP-NV, an ablation of FLIP that performs the same beam search but with no value module as guidance.

Results. Table 1 shows the results. We can see that UniPi achieves low success rates across all tasks, which shows that directly synthesizing long-horizon videos is difficult. FLIP-NV achieves better results than UniPi. This shows that with dense flow information as guidance, the performance of the video generation model is improved. FLIP outperforms all baselines, pointing out the effectiveness of using value functions for model-based planning. This can eliminate incorrect search routes during planning. We show such incorrect search routes on our website.

5.2 Long-Horizon Video Generation Evaluation

Setup. In this section, we quantitatively evaluate the long-horizon video generation quality of FLIP compared to other video generation models. We choose the same datasets as in Section 5.1 as well as Bridge-V2 [24] as the evaluation benchmarks. Here all videos are longer than 200 frames except for Bridge-V2. For Bridge-V2, we train on 10k videos and test on 256 videos with a resolution of $96 \times 128 \times 3$. We choose two baselines: 1) LVDM [70], a state-of-the-art text-to-video method for video generation; 2) IRASim [25], a conditional video generation method with the end-effector trajectories as the condition. We use SAM2 [62] to label the end-effector trajectory for IRASim. We choose model-based metrics including Latent L2 loss and FVD [71] as well as a computation-based metric PSNR [72]. Latent L2 loss and PSNR measure the L2 distance between the predicted video and the ground-truth video in the latent space and pixel space, and FVD assess video quality by analyzing the similarity of video feature distributions

Results. Table 2 shows the results. The analysis is in Appendix C. More results about Plan-Guided Low-Level Policy, Fundamental Module, Applications, and Scaling can be found in Appendix C.

6 Conclusion and Limitation

In this work, we present FLIP, a flow-centric generative planning method for general-purpose manipulation tasks. FLIP is trained on only video and language data, can perform model-based planning on the trained world model to synthesize long-horizon plans, and can guide low-level policy learning. FLIP has the potential to scale up with increasing data and computation budgets in the future.

A major limitation of FLIP is the slow speed of planning, which is restricted by extensive video generation processes during the planning phase. This restricts our method on quasi-static manipulation tasks. Another limitation is that FLIP does not use physical properties and 3D information of the scene. Future works can develop physical 3D world models and extend FLIP to 3D scenarios.

Acknowledgments

We would like to thank Tianren Zhang for his helpful discussions.

References

[1] Y. LeCun. A path towards autonomous machine intelligence. 2022. *URL https://openreview.net/pdf*, 2024.

[2] D. Ha and J. Schmidhuber. Recurrent world models facilitate policy evolution. *Advances in neural information processing systems*, 31, 2018.

[3] T. Brooks, B. Peebles, C. Holmes, W. DePue, Y. Guo, L. Jing, D. Schnurr, J. Taylor, T. Luhman, E. Luhman, C. Ng, R. Wang, and A. Ramesh. Video generation models as world simulators. 2024. URL https://openai.com/research/video-generation-models-as-world-simulators.

[4] A. Blattmann, T. Dockhorn, S. Kulal, D. Mendelevitch, M. Kilian, D. Lorenz, Y. Levi, Z. English, V. Voleti, A. Letts, et al. Stable video diffusion: Scaling latent video diffusion models to large datasets. *arXiv preprint arXiv:2311.15127*, 2023.

[5] M. Yang, Y. Du, K. Ghasemipour, J. Tompson, D. Schuurmans, and P. Abbeel. Learning interactive real-world simulators. *arXiv preprint arXiv:2310.06114*, 2023.

[6] R. Mendonca, S. Bahl, and D. Pathak. Structured world models from human videos. *arXiv preprint arXiv:2308.10901*, 2023.

[7] Y. Seo, J. Kim, S. James, K. Lee, J. Shin, and P. Abbeel. Multi-view masked world models for visual robotic manipulation. In *International Conference on Machine Learning*, pages 30613–30632. PMLR, 2023.

[8] C. Caucheteux and J.-R. King. Brains and algorithms partially converge in natural language processing. communications biology, 5 (1), 134, 2022.

[9] M. Manto, J. M. Bower, A. B. Conforto, J. M. Delgado-García, S. N. F. Da Guarda, M. Gerwig, C. Habas, N. Hagura, R. B. Ivry, P. Mariën, et al. Consensus paper: roles of the cerebellum in motor control—the diversity of ideas on cerebellar involvement in movement. *The Cerebellum*, 11:457–487, 2012.

[10] Y. Du, M. Yang, P. Florence, F. Xia, A. Wahid, B. Ichter, P. Sermanet, T. Yu, P. Abbeel, J. B. Tenenbaum, et al. Video language planning. *arXiv preprint arXiv:2310.10625*, 2023.

[11] S. Zhou, Y. Du, J. Chen, Y. Li, D.-Y. Yeung, and C. Gan. Robodreamer: Learning compositional world models for robot imagination. *arXiv preprint arXiv:2404.12377*, 2024.

[12] D. Driess, F. Xia, M. S. Sajjadi, C. Lynch, A. Chowdhery, B. Ichter, A. Wahid, J. Tompson, Q. Vuong, T. Yu, et al. Palm-e: An embodied multimodal language model. *arXiv preprint arXiv:2303.03378*, 2023.

[13] J. Wu, S. Yin, N. Feng, X. He, D. Li, J. Hao, and M. Long. ivideogpt: Interactive videogpts are scalable world models. *arXiv preprint arXiv:2405.15223*, 2024.

[14] N. Karaev, I. Rocco, B. Graham, N. Neverova, A. Vedaldi, and C. Rupprecht. Cotracker: It is better to track together. *arXiv preprint arXiv:2307.07635*, 2023.

[15] C. Wen, X. Lin, J. So, K. Chen, Q. Dou, Y. Gao, and P. Abbeel. Any-point trajectory modeling for policy learning. *arXiv preprint arXiv:2401.00025*, 2023.

[16] M. Xu, Z. Xu, Y. Xu, C. Chi, G. Wetzstein, M. Veloso, and S. Song. Flow as the cross-domain manipulation interface. *arXiv preprint arXiv:2407.15208*, 2024.

[17] Z. Xu, C. Gao, Z. Liu, G. Yang, C. Tie, H. Zheng, H. Zhou, W. Peng, D. Wang, T. Chen, et al. Manifoundation model for general-purpose robotic manipulation of contact synthesis with arbitrary objects and robots. *arXiv preprint arXiv:2405.06964*, 2024.

[18] D. P. Kingma. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[19] W. Peebles and S. Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4195–4205, 2023.

[20] Y. J. Ma, V. Kumar, A. Zhang, O. Bastani, and D. Jayaraman. Liv: Language-image representations and rewards for robotic control. In *International Conference on Machine Learning*, pages 23301–23320. PMLR, 2023.

[21] B. Selman and C. P. Gomes. Hill-climbing search. *Encyclopedia of cognitive science*, 81 (333-335):10, 2006.

[22] B. Liu, Y. Zhu, C. Gao, Y. Feng, Q. Liu, Y. Zhu, and P. Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. *Advances in Neural Information Processing Systems*, 36, 2024.

[23] J. Luo, C. Xu, F. Liu, L. Tan, Z. Lin, J. Wu, P. Abbeel, and S. Levine. Fmb: A functional manipulation benchmark for generalizable robotic learning. *The International Journal of Robotics Research*, page 02783649241276017, 2023.

[24] H. R. Walke, K. Black, T. Z. Zhao, Q. Vuong, C. Zheng, P. Hansen-Estruch, A. W. He, V. Myers, M. J. Kim, M. Du, et al. Bridgedata v2: A dataset for robot learning at scale. In *Conference on Robot Learning*, pages 1723–1736. PMLR, 2023.

[25] F. Zhu, H. Wu, S. Guo, Y. Liu, C. Cheang, and T. Kong. Irasim: Learning interactive real-robot action simulators. *arXiv preprint arXiv:2406.14540*, 2024.

[26] T. Lesort, N. Díaz-Rodríguez, J.-F. Goudou, and D. Filliat. State representation learning for control: An overview. *Neural Networks*, 108:379–392, 2018.

[27] N. Ferns, P. Panangaden, and D. Precup. Metrics for finite markov decision processes. In *UAI*, volume 4, pages 162–169, 2004.

[28] S. Nasiriany, V. Pong, S. Lin, and S. Levine. Planning with goal-conditioned policies. *Advances in neural information processing systems*, 32, 2019.

[29] C. Finn and S. Levine. Deep visual foresight for planning robot motion. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2786–2793. IEEE, 2017.

[30] L. Kaiser, M. Babaeizadeh, P. Milos, B. Osinski, R. H. Campbell, K. Czechowski, D. Erhan, C. Finn, P. Kozakowski, S. Levine, et al. Model-based reinforcement learning for atari. *arXiv preprint arXiv:1903.00374*, 2019.

[31] D. Hafner, T. Lillicrap, M. Norouzi, and J. Ba. Mastering atari with discrete world models. *arXiv preprint arXiv:2010.02193*, 2020.

[32] D. Hafner, J. Pasukonis, J. Ba, and T. Lillicrap. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023.

[33] N. Hansen, H. Su, and X. Wang. Td-mpc2: Scalable, robust world models for continuous control. *arXiv preprint arXiv:2310.16828*, 2023.

[34] B. Baker, I. Akkaya, P. Zhokov, J. Huizinga, J. Tang, A. Ecoffet, B. Houghton, R. Sampedro, and J. Clune. Video pretraining (vpt): Learning to act by watching unlabeled online videos. *Advances in Neural Information Processing Systems*, 35:24639–24654, 2022.

[35] V. Micheli, E. Alonso, and F. Fleuret. Transformers are sample-efficient world models. *arXiv preprint arXiv:2209.00588*, 2022.

[36] K. Grauman, A. Westbury, E. Byrne, Z. Chavis, A. Furnari, R. Girdhar, J. Hamburger, H. Jiang, M. Liu, X. Liu, et al. Ego4d: Around the world in 3,000 hours of egocentric video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18995–19012, 2022.

[37] J. Bruce, M. D. Dennis, A. Edwards, J. Parker-Holder, Y. Shi, E. Hughes, M. Lai, A. Mavalankar, R. Steigerwald, C. Apps, et al. Genie: Generative interactive environments. In *Forty-first International Conference on Machine Learning*, 2024.

[38] M. Shridhar, Y. L. Lo, and S. James. Generative image as action models. *arXiv preprint arXiv:2407.07875*, 2024.

[39] D. Valevski, Y. Leviathan, M. Arar, and S. Fruchter. Diffusion models are real-time game engines. *arXiv preprint arXiv:2408.14837*, 2024.

[40] H. Bharadhwaj, R. Mottaghi, A. Gupta, and S. Tulsiani. Track2act: Predicting point tracks from internet videos enables diverse zero-shot robot manipulation. *arXiv preprint arXiv:2405.01527*, 2024.

[41] Z. Jiang, H. Jiang, and Y. Zhu. Doduo: Learning dense visual correspondence from unsupervised semantic-aware flow. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 12420–12427. IEEE, 2024.

[42] D. Seita, Y. Wang, S. J. Shetty, E. Y. Li, Z. Erickson, and D. Held. Toolflownet: Robotic manipulation with tools via predicting tool flow from point clouds. In *Conference on Robot Learning*, pages 1038–1049. PMLR, 2023.

[43] C. Yuan, C. Wen, T. Zhang, and Y. Gao. General flow as foundation affordance for scalable robot learning. *arXiv preprint arXiv:2401.11439*, 2024.

[44] Y. Du, S. Yang, B. Dai, H. Dai, O. Nachum, J. Tenenbaum, D. Schuurmans, and P. Abbeel. Learning universal policies via text-guided video generation. *Advances in Neural Information Processing Systems*, 36, 2024.

[45] D. Brandfonbrener, O. Nachum, and J. Bruna. Inverse dynamics pretraining learns good representations for multitask imitation. *Advances in Neural Information Processing Systems*, 36, 2024.

[46] C. Gao, H. Gao, S. Guo, T. Zhang, and F. Chen. Cril: Continual robot imitation learning via generative and prediction model. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6747–5754. IEEE, 2021.

[47] Y. J. Ma, S. Sodhani, D. Jayaraman, O. Bastani, V. Kumar, and A. Zhang. Vip: Towards universal visual reward and representation via value-implicit pre-training. *arXiv preprint arXiv:2210.00030*, 2022.

[48] S. Nair, A. Rajeswaran, V. Kumar, C. Finn, and A. Gupta. R3m: A universal visual representation for robot manipulation. *arXiv preprint arXiv:2203.12601*, 2022.

[49] K. Zakka, A. Zeng, P. Florence, J. Tompson, J. Bohg, and D. Dwibedi. Xirl: Cross-embodiment inverse reinforcement learning. In *Conference on Robot Learning*, pages 537–546. PMLR, 2022.

[50] P. Sermanet, C. Lynch, Y. Chebotar, J. Hsu, E. Jang, S. Schaal, S. Levine, and G. Brain. Time-contrastive networks: Self-supervised learning from video. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 1134–1141. IEEE, 2018.

[51] W. Huang, C. Wang, Y. Li, and F.-f. Li. Rekep: Spatio-temporal reasoning of relational keypoint constraints for robotic manipulation. 2024.

[52] N. Di Palo and E. Johns. Keypoint action tokens enable in-context imitation learning in robotics. *arXiv preprint arXiv:2403.19578*, 2024.

[53] S. Bahl, R. Mendonca, L. Chen, U. Jain, and D. Pathak. Affordances from human videos as a versatile representation for robotics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13778–13790, 2023.

[54] T. Shu, X. Gao, M. S. Ryoo, and S.-C. Zhu. Learning social affordance grammar from videos: Transferring human interactions to human-robot interactions. In *2017 IEEE international conference on robotics and automation (ICRA)*, pages 1669–1676. IEEE, 2017.

[55] K. Zhang, B. Li, K. Hauser, and Y. Li. Adaptigraph: Material-adaptive graph-based neural dynamics for robotic manipulation. *arXiv preprint arXiv:2407.07889*, 2024.

[56] H. Jiang, B. Huang, R. Wu, Z. Li, S. Garg, H. Nayyeri, S. Wang, and Y. Li. Roboexp: Action-conditioned scene graph via interactive exploration for robotic manipulation. *arXiv preprint arXiv:2402.15487*, 2024.

[57] S. Kumar, J. Zamora, N. Hansen, R. Jangir, and X. Wang. Graph inverse reinforcement learning from diverse videos. In *Conference on Robot Learning*, pages 55–66. PMLR, 2023.

[58] C. Wang, L. Fan, J. Sun, R. Zhang, L. Fei-Fei, D. Xu, Y. Zhu, and A. Anandkumar. Mimicplay: Long-horizon imitation learning by watching human play. *arXiv preprint arXiv:2302.12422*, 2023.

[59] L. Shao, T. Migimatsu, Q. Zhang, K. Yang, and J. Bohg. Concept2robot: Learning manipulation concepts from instructions and human demonstrations. *The International Journal of Robotics Research*, 40(12-14):1419–1434, 2021.

[60] J. Liang, R. Liu, E. Ozguroglu, S. Sudhakar, A. Dave, P. Tokmakov, S. Song, and C. Vondrick. Dreamitate: Real-world visuomotor policy learning via video generation. *arXiv preprint arXiv:2406.16862*, 2024.

[61] S. Liu, Z. Ren, S. Gupta, and S. Wang. Physgen: Rigid-body physics-grounded image-to-video generation. In *European Conference on Computer Vision*, 2024.

[62] N. Ravi, V. Gabeur, Y.-T. Hu, R. Hu, C. Ryali, T. Ma, H. Khedr, R. Rädle, C. Rolland, L. Gustafson, et al. Sam 2: Segment anything in images and videos. *arXiv preprint arXiv:2408.00714*, 2024.

[63] A. Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.

[64] A. Dubey, A. Jauhri, A. Pandey, A. Kadian, A. Al-Dahle, A. Letman, A. Mathur, A. Schelten, A. Yang, A. Fan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

[65] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16000–16009, 2022.

[66] X. Ma, Y. Wang, G. Jia, X. Chen, Z. Liu, Y.-F. Li, C. Chen, and Y. Qiao. Latte: Latent diffusion transformer for video generation. *arXiv preprint arXiv:2401.03048*, 2024.

[67] L. Zhang, A. Rao, and M. Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3836–3847, 2023.

[68] F. Bao, S. Nie, K. Xue, Y. Cao, C. Li, H. Su, and J. Zhu. All are worth words: A vit backbone for diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 22669–22679, 2023.

[69] R. S. Sutton. Reinforcement learning: An introduction. *A Bradford Book*, 2018.

[70] Y. He, T. Yang, Y. Zhang, Y. Shan, and Q. Chen. Latent video diffusion models for high-fidelity long video generation. *arXiv preprint arXiv:2211.13221*, 2022.

[71] T. Unterthiner, S. Van Steenkiste, K. Kurach, R. Marinier, M. Michalski, and S. Gelly. Towards accurate generative models of video: A new metric & challenges. *arXiv preprint arXiv:1812.01717*, 2018.

[72] A. Hore and D. Ziou. Image quality metrics: Psnr vs. ssim. In *2010 20th international conference on pattern recognition*, pages 2366–2369. IEEE, 2010.

[73] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

[74] J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[75] A. Q. Nichol and P. Dhariwal. Improved denoising diffusion probabilistic models. In *International conference on machine learning*, pages 8162–8171. PMLR, 2021.

[76] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

[77] D. Podell, Z. English, K. Lacey, A. Blattmann, T. Dockhorn, J. Müller, J. Penna, and R. Rombach. Sdxl: Improving latent diffusion models for high-resolution image synthesis. *arXiv preprint arXiv:2307.01952*, 2023.

[78] J. Aldaco, T. Armstrong, R. Baruch, J. Bingham, S. Chan, K. Draper, D. Dwibedi, C. Finn, P. Florence, S. Goodrich, et al. Aloha 2: An enhanced low-cost hardware for bimanual teleoperation. *arXiv preprint arXiv:2405.02292*, 2024.

[79] J. Wang, Y. Yuan, H. Che, H. Qi, Y. Ma, J. Malik, and X. Wang. Lessons from learning to spin" pens". *arXiv preprint arXiv:2407.18902*, 2024.

[80] T. Chen, E. Cousineau, N. Kuppuswamy, and P. Agrawal. Vegetable peeling: A case study in constrained dexterous manipulation. *arXiv preprint arXiv:2407.07884*, 2024.

[81] C. Gao, Z. Li, H. Gao, and F. Chen. Iterative interactive modeling for knotting plastic bags. In *Conference on Robot Learning*, pages 571–582. PMLR, 2023.

[82] C. Lynch, A. Wahid, J. Tompson, T. Ding, J. Betker, R. Baruch, T. Armstrong, and P. Florence. Interactive language: Talking to robots in real time. *IEEE Robotics and Automation Letters*, 2023.

A Method Details

A.1 Latent Diffusion Models

Diffusion Models. Diffusion models [73, 74] typically contain a forward nosing process and a reverse denoising process. During the forward process, we gradually apply noise to real data x_0: $q(x_t|x_0) = \mathcal{N}(x_t; \sqrt{\overline{a}_t}x_0, (1-\overline{a}_t)\mathbf{I})$ over T timesteps, where constants $\overline{a}_t$ are hyperparameters. By applying the reparameterization trick, we can sample $x_t = \sqrt{\overline{a}_t}x_0 + \sqrt{1-\overline{a}_t}\epsilon_t$, where $\epsilon_t \sim \mathcal{N}(0, \mathbf{I})$. During the reverse process, it starts from Gaussian noise $x_T \sim \mathcal{N}(0, \mathbf{I})$ and gradually removes noises to recover x_0: $p_\theta(x_{t-1}|x_t) = \mathcal{N}(\mu_\theta(x_t), \Sigma_\theta(x_t))$. With reparameterizing μ_θ as a noise prediction network ϵ_θ, the model can be trained with $\mathcal{L}_{simple}(\theta) = ||\epsilon_\theta(x_t) - \epsilon_t||_2^2$. We also learn the covariance Σ_θ following Nichol and Dhariwal [75], Peebles and Xie [19] with the full KL loss.

Latent Diffusion and Tokenization. Latent diffusion models [76, 66] perform diffusion process in a low-dimensional latent space z^{ld} rather than the original pixel space. We leverage the pre-trained VAE in SDXL [77] to compress each frame o_t to latent representations: $z_t^{ld} = Enc(o_t)$, and after the denoising process, the latent representation can be decoded back to the pixel space with the VAE decoder: $o_t = Dec(z_t^{ld})$. For each z^{ld}, it is divided into image patches and tokenized by convolutional networks to P tokens with D dimensions (hidden size). Sequencing the image tokens of all T frames, we get the video token in the shape of $T \times P \times D$.

Spatial-Temporal Attention Mechanism. We leverage transformers [63] to implement the dynamics module, and use the memory-efficient spatial-temporal attention mechanism [66, 37, 25], where each attention block consists of a spatial attention block and a temporal attention block. The spatial attention operates on the $1 \times P$ tokens within each frame, and the temporal attention operates on the $T \times 1$ tokens across T timesteps at the same location.

A.2 Language-Vision Representation

The original fine-tuning loss $\mathcal{L}_{LIV} = \mathcal{L}_I(\psi_I) + \mathcal{L}_L(\psi_I, \psi_L)$ is calculated on sampled sub-trajectory batch data $\left\{o_s^i, \ldots, o_k^i, o_{k+1}^i, \ldots, o_T^i, g^i\right\}_{i=1}^B$ from each task $\mathcal{T}_i$, where $s \in [0, T_i - 1]\,, s \leq k < T_i$. They have the following forms:

$$\mathcal{L}_I(\psi_I) = \frac{1-\gamma}{B}\sum_{i=1}^{B}[-\mathcal{S}(\psi_I(o_s^i), \psi_I(o_T^i))] + \log\frac{1}{B}\sum_{i=1}^{B}\exp[\mathcal{S}(\psi_I(o_k^i), \psi_I(o_T^i)) + 1 - \gamma\mathcal{S}(\psi_I(o_{k+1}^i), \psi_I(o_T^i))],$$
$$\mathcal{L}_L(\psi_I, \psi_L) = \frac{1-\gamma}{B}\sum_{i=1}^{B}\left[-\log\frac{e^{(1-\gamma)\mathcal{S}\left(\psi_I\left(o_T^i\right), \psi_L\left(\left(g^i\right)\right)\right.}}{\frac{1}{B}\sum_{j=1}^{B}\left[e^{(1-\gamma)\mathcal{S}\left(\psi_I\left(o_T^j\right), \psi_L(g^i)\right)}\right]}\right], \tag{2}$$

A.3 Low-Level Policy

In this work, we use a low-level policy with a similar structure to ATM [15]. The flow-guided policy, video-guided policy, and the flow-video-guided policy are illustrated in Figure 5. We employ a spatial-temporal attention mechanism. Specifically, the input contains the agent view observation history $o_{t-4:t}^a \in \mathbb{R}^{4\times3\times128\times128}$ and the eye in hand observation history $o_{t-4:t}^e \in \mathbb{R}^{4\times3\times128\times128}$ at timestep t, the proprioception state history $s_{t-4:t} \in \mathbb{R}^{4\times47}$, the language tokens extracted from Meta Llama 3.1 8B [64] $g \in \mathbb{R}^{T_g\times4096}$, the predicted flow for both the agent view $\mathbf{p}_{t:t+16}^a \in \mathbb{R}^{16\times529\times2}$ and eye in hand view $\mathbf{p}_{t:t+16}^e \in \mathbb{R}^{16\times529\times2}$, and the predicted future videos for both the agent view $\hat{o^a}_{t:t+16} \in \mathbb{R}^{16\times3\times128\times128}$ and eye in hand view $\hat{o^e}_{t:t+16} \in \mathbb{R}^{16\times3\times128\times128}$. The low-level policies output the future action sequences $a \in \mathbb{R}^{8\times7}$ where 7 is the action size. The policies have three stages:

Spatial Encoding. First, o^a and o^e are patchified by CNNs, and adding spatial and temporal positional encodings. The language token is processed by an MLP to reduce the dimension to 384.

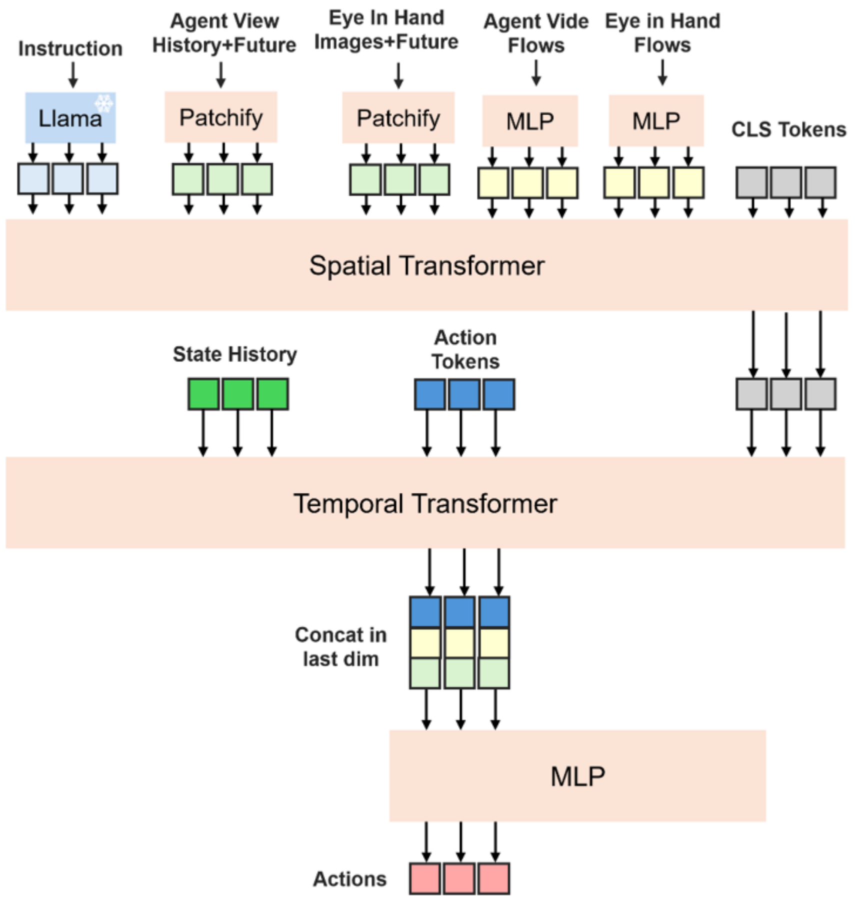


Figure 5: Low-level policies.

For video-guided policy and flow-video-guided policy, the $\hat{o^a}$ and $\hat{o^e}$ are also patchified in the same manner. For the flow-guided policy and flow-video-guided policy, the flows $\mathbf{p}^a$ and $\mathbf{p}^e$ are tokenized where and MLP is used to process each query point and get a set of flow tokens $\in \mathbb{R}^{529\times384}$. Second, these tokens are sent to a 4 layer spatial transformer together with a CLS token $\in \mathbb{R}^{16\times384}$.

Temporal Encoding. The output at the CLS token positions are concatenated with s_t and a set of action tokens $\in \mathbb{R}^{16\times384}$ and sent to a temporal transformer.

Late Fusion. The output at the action tokens are concatenated with the flow tokens. For video-guided policy and flow-video-guided policy, they are also concatenated with the observation history tokens. Then an MLP is used to process these inputs and predict the future actions a.

B Experiment Details

B.1 Training Details

We report the hyperparameters of the models we trained in Table 3 and Table 4. We train all data with observation history equals to 16 and future flow horizon equals to 16.

C More Results

We can see that our method consistently outperforms baselines in all datasets. LVDM performs badly on LIBERO-LONG and FMB, and better on Bridge-V2. This is because the videos in Bridge-

	CVAE-S	CVAE-B	CVAE-L
Encoder Layer	4	6	8
Decoder Layer	6	8	12
Hideen Size	384	768	1024
Learning Rate	1e-4	5e-5	1e-5
Image Patch Size	8	8	8
Head Number	4	8	12

Table 3: Hyperparameters of CVAE.

	D-S	D-B	D-L
Layers	8	12	16
Hideen Size	384	768	1024
Learning Rate	1e-4	1e-4	1e-4
Head Number	6	12	16

Table 4: Hyperparameters of the dynamics module.

V2 are shorter than the previous two benchmarks. IRASim performs better than LVDM, which shows the importance of trajectory guidance. However, it generates long-horizon videos in an auto-regressive manner, which has worse results than our method, showing that model-based planning can also help generate high-quality videos by concatenating short-horizon videos generated with rich flow guidance. The results on the FMB benchmark are the worst for all methods. This is because the training videos have many discontinuous transitions, where the robot gripper instantly moves to where the next stage begins. Since our model leverages history observations as input conditions, it can sometimes overcome this discontinuous gap. We qualitatively show the model-based planning results on the four tasks in Figure 4.

Since FLIP is a universal framework for all manipulation tasks as long as they have language-annotated video datasets, here we qualitatively show FLIP can be used for complex long-horizon video generation including the ALOHA tasks [78], pen spinning [79], robot pilling [80], tying plastic bags [81], and human peeling eggs, as shown in Figure 8. More video demos are on our website.

C.1 Plan-Guided Low-Level Policy

Setup. In this evaluation we explore how the generated flow and video plans can be used as conditions for training a manipulation policy to accomplish the task. We aim to answer the question: which one, flow or video (or both at the same time), is more suitable to be used as the condition to guide the learning of the underlying strategy? We use LIBERO-LONG [22] for evaluation, where for each task in LIBERO-LONG, we use 10 demonstrations with action labels and 50 demonstrations without action labels, as done in the baseline method ATM [15]. In the evaluation, we use a receding horizon online planning manner: FLIP will first search the whole task plan, and the policy will take a plan of 16 steps as condition and predicts and go 8 steps, then FLIP will plans again and the whole process goes iteratively.

Results. The results are in Figure 6. We can see that the flow-guided policy achieves a little higher success rate and lower variance than ATM, showing that dense flow information is better than sparse flow information. The video-guided policy achieves the best average success rates across all methods, but it has a large variance. This shows that high-quality future videos can serve as better guidance for policy learning, however, consistently generating high-quality videos across different tasks is more difficult than flows. Surprisingly, the flow-video-guided policy performs worse than when they were the only condition respectively. This may come from that the errors from two conditions are superimposed, which can lead to worse performances.

C.2 Experiments on Fundamental Modules of FLIP

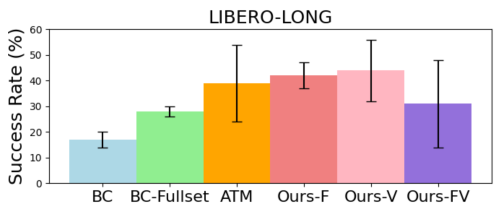


Figure 6: Success rates of different low-level policies on LIBERO-LONG.

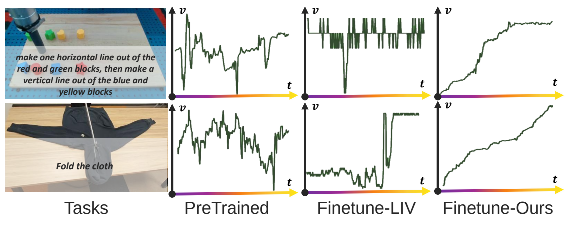


Figure 7: Value curves from the pretrained LIV, fine-tuned by LIV, and fine-tuned by FLIP.

	LIBERO-10			Language-Table			Bridge-V2		
	Latent L2 ↓	FVD ↓	PSNR ↑	Latent L2 ↓	FVD ↓	PSNR ↑	Latent L2 ↓	FVD ↓	PSNR ↑
LVDM [70]	0.366	109.41	18.852	0.364	124.75	19.943	0.328	111.34	18.104
IRASim [25]	0.307	92.76	19.205	0.335	132.56	18.156	0.318	107.89	19.967
FLIP-SC	0.271	89.77	20.089	0.304	137.89	18.904	0.316	127.65	18.375
FLIP(Ours)	**0.197**	**27.62**	**28.602**	**0.159**	**21.23**	**33.632**	**0.171**	**38.41**	**34.576**

Table 5: Quantitative results on short-horizon video generation.

	LIBERO-10		Bridge-V2	
	ADE ↓	LTDR ↑	ADE ↓	LTDR ↑
ATM [15]	19.6	53.8%	18.4	66.1%
Ours-ABS	20.5	57.3%	17.9	59.3%
Ours-NoAUX	14.5	73.2%	12.7	75.6%
Ours	**12.7**	**76.5%**	**11.9**	**80.2%**

Table 6: Quantitative results of the action model.

Action Module Experiments. We use two metrics to assess the flow generation model π_f quantitatively [41]: 1) Average Distance Error (ADE) between the generated and the ground truth flows in pixel units on all query points; 2) Less Than Delta Ratio (LTDR): the average percentage of points within the distance threshold of 1, 2, 4, and 8 pixels between the reconstructed and the ground truth flows at each time step. Since most of the points are stationary points, in order to better demonstrate the results, we only calculate points with $\delta_s \geq 1$.

We use LIBERO-LONG [22] and Bridge-V2 [24] for evaluation. We compare our method with 3 baselines: 1) ATM [15], the state-of-the-art flow prediction module for manipulation tasks; 2) Ours-ABS: directly generating absolute flow coordinates at each timestep rather than generating the scale and direction; 3) Ours-NoAUX: the same architecture of ours with no auxiliary training losses (the flow and image reconstruction losses).

From Table 6, we can see that Ours-ABS generally achieves the same results as ATM, and predicting the scale and directions are better than ATM and Ours-ABS, showing that directly regressing the absolute coordinates is worse than predicting the delta of flows at each timestep. We can also see that the auxiliary losses can help improve the final results.

Dynamics Module Experiments. We evaluate our dynamics module separately with the ground truth flows as conditions on *short-horizon* video generation. We use PSNR [72], latent L2 loss, and FVD [71] as metrics. We use LIBERO-LONG [22], Bridge-V2 [24], and Language-Table [82] as the evaluation datasets. We use three baselines (as introduced in Section 5.2): 1) LVDM [70]; 2) IRAsim [25]; 3) Ours-SC: using AdaLN-Zero for all kinds of conditions.

Results are in Table 5. The result trends across methods are generally consistent with the long-horizon video generation results in Table 2. FLIP-SC generally achieves the same performance with IRASim, showing that even if the model is given dense flow information, it requires a fine-grained mechanism to leverage the condition for video generation.

Value Module Experiments. We here qualitatively show the fine-tuned value curves of our method compared to the original LIV [20] method on two different tasks consisting of Language-Table [82] and cloth folding in Figure 7. We also show the value curves before fine-tuning. We can see our method consistently gets smoother value curves than the original LIV method, where the value curves have violent oscillations.

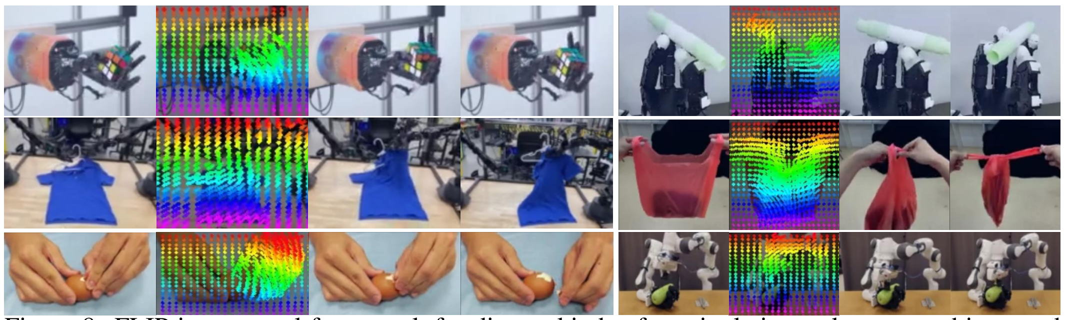


Figure 8: FLIP is a general framework for diverse kinds of manipulation tasks across objects and robots, even for human hands. All of the flows and images are generated.

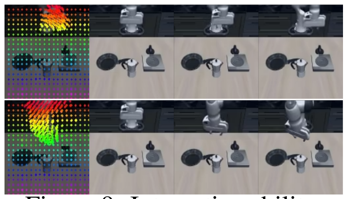


Figure 9: Interactive ability.

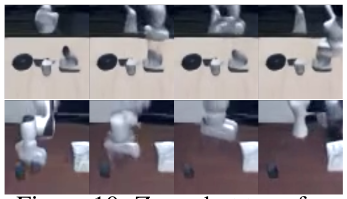


Figure 10: Zero-shot transfer.

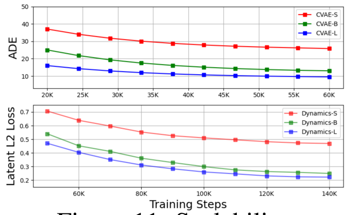


Figure 11: Scalability.

C.3 Applications and Scaling

Finally, we train FLIP on LIBERO-90, a large-scale simulation manipulation dataset to show three properties of FLIP. We use 50 videos for each task in the resolution of $3 \times 64 \times 64$.

Interactive World Model. We first show that the trained dynamics module is interactive: it can generate corresponding videos given image flows specified by humans. We use SAM2 [62] to select the region of the robot arm and manually give flows in different directions. Results are shown in Figure 9. We can see the robot arm can move left or right according to the given flow.

Zero-Shot Generation. Secondly, we show that the trained FLIP has zero-shot transfer ability. We test the trained model on LIBERO-LONG. Results are shown in Figure 10. Interestingly, we can see that the pretrained model, without fine-tuning, can generate natural movement for the robot arm with unseen observations and instructions. This shows FLIP has a certain knowledge transfer ability.

Model Scaling. We show that the action and dynamics module are scalable with increasing model sizes. Figure 11 shows the smoothed ADE and Latent L2 loss on the validation set. It shows that increasing the model size can consistently help achieve better performance for both modules.