Magic Fixup: Streamlining Photo Editing by Watching Dynamic Videos

HADI ALZAYER, University of Maryland & Adobe, USA
ZHIHAO XIA, Adobe, USA
XUANAR ZHANG, Adobe, USA
ELI SHECHTMAN, Adobe, USA
JIA-BIN HUANG, Adobe, USA
MICHAEL GHARBI, Adobe, USA

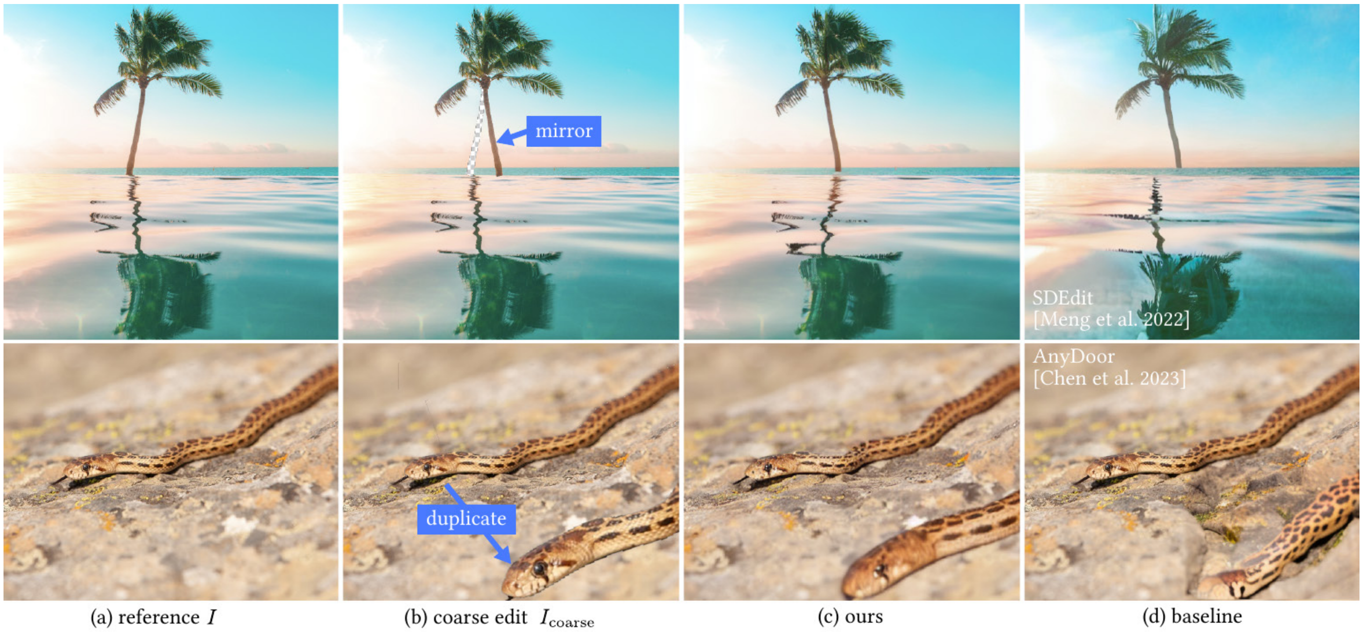


Fig. 1. **Applications of Magic Fixup.** We propose a diffusion model for image editing. Starting from an input image (a), a user specifies their desired changes by rearranging automatically segmented scene objects using simple 2D transforms to produce a coarse edit (b). Our model transforms this coarse edit into a realistic image (c), correctly accounting for secondary effects critical for realism, such as reflections on the water (top) or changes in depth-of-field (bottom), producing much more plausible edits than state-of-the-art methods (d). Photos sourced from ©Unsplash.

We propose a generative model that, given a coarsely edited image, synthesizes a photorealistic output that follows the prescribed layout. Our method transfers fine details from the original image and preserve the identity of its parts. Yet, it adapts it to the lighting and context defined by the new layout. Our key insight is that videos are a powerful source of supervision for this task: objects and camera motions provide many observations of how the world changes with viewpoint, lighting, and physical interactions. We construct an image dataset in which each sample is a pair of source and target frames extracted from the same video at randomly chosen time intervals. We warp the source frame toward the target using two motion models that mimic the expected test-time user edits. We supervise our model to translate the warped image into the ground truth, starting from a pretrained diffusion model. Our model design explicitly enables fine detail transfer from the source frame to the generated image, while closely following the user-specified layout. We show that by using simple segmentations and coarse 2D manipulations, we can synthesize a photorealistic edit faithful to the user's input while addressing second-order effects like harmonizing the lighting and physical interactions between edited objects. Project page and code can be found at https://magic-fixup.github.io

CCS Concepts: • **Computing methodologies → Image editing**.

Additional Key Words and Phrases: Photorealistic editing, Spatial editing, Learning from videos

Authors' Contact Information: Hadi Alzayer, hadi@umd.edu, University of Maryland & Adobe, College Park, MD, USA; Zhihao Xia, zhihao.zach.xia@gmail.com, Adobe, San Jose, CA, USA; Xuanar Zhang, cezhangxer@gmail.com, Adobe, San Jose, CA, USA; Eli Shechtman, elishe@adobe.com, Adobe, Seattle, CA, USA; Jia-Bin Huang, jbhuang@umd.edu, Adobe, San Francisco, CA, USA; Michael Gharbi, mgharbi@gmail.com, Adobe, San Francisco, CA, USA.

ACM XXXX-XXXX/2025/8-ART
https://doi.org/XXXXXXX.XXXXXXX

ACM Reference Format:
Hadi Alzayer, Zhihao Xia, Xuanar Zhang, Eli Shechtman, Jia-Bin Huang, and Michael Gharbi. 2025. Magic Fixup: Streamlining Photo Editing by Watching Dynamic Videos. 1, 1 (August 2025), 25 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

Image editing is a labor-intensive process. Although humans can quickly and easily rearrange parts of an image to compose a new one, simple edits can easily look unrealistic, e.g., when the scene lighting and physical interactions between objects become inconsistent. Fixing these issues manually to make the edit plausible requires professional skills and careful modifications, sometimes down to the pixel level. The success of recent generative models [Dhariwal and Nichol 2021; Esser et al. 2021; Ho et al. 2020; Rombach et al. 2022] paves the way for a new generation of automated tools that increase the realism of image edits while requiring much sparser user inputs [Andonian et al. 2021; Couairon et al. 2023; Kim et al. 2022; Sarukkai et al. 2024]. Generative methods providing explicit spatial keypoints control have been proposed but are either limited to certain domains [Pan et al. 2023] or modest changes [Shi et al. 2024]. State-of-the-art approaches, however, regenerate pixels based on a user-specified text prompt and a mask of the region to influence [Brooks et al. 2023; Cao et al. 2023; Wang et al. 2023; Xie et al. 2023]. This interface is not always natural. In particular, it does not allow spatial transformations of the existing scene content, as we show in Figure 2, and object identities are often not fully preserved by the re-synthesis step [Chen et al. 2024; Song et al. 2023].

In this paper, we propose a new approach to image editing that offers the controls of conventional editing methods and the realism of the modern generative model (Figure 1). Our method uses human inputs where it shines: users can segment the image and rearrange its parts manually in a "cut-and-transform" approach, e.g., using simple 2D transforms, duplication, or deletion to construct their desired layout, just like a collage [Sarukkai et al. 2024]. We call our collage-like editing interface the *Collage Transform*. We then train a diffusion model to take care of the hard work of making the edit photorealistic. Our model "projects" the coarsely edited image onto the natural image manifold, fixing up all the low-level image cues that violate its image prior, such as tweaking poses, blending object boundaries, harmonizing colors, adding cast shadows, reflections and other second-order interactions between the object and the environment.

Crucially, we explicitly fine-tune a latent diffusion model [Rombach et al. 2022] so its output deviates as little as possible from the user's specifications and the appearance of the original objects in the scene. This is essential for photographers, as they spend significant effort capturing their images and would like to retain the content identity as much as possible. When editing an image, there is a subtle balance between being faithful to the original image and harmonizing the edited image to preserve realism. This is the regime that our work focuses on. Our insight is that videos provide a rich signal of how an edited photo's appearance should change to preserve photorealism. From videos, we can learn how objects' appearances change in the real world as they deform and move under changing light. Camera motion and disocclusions give us priors about what hides behind other objects and how the same object looks under changing perspectives.

To exploit these cues, we build a paired image dataset from a large-scale video corpus. Each pair corresponds to two frames sampled from the same video: source and target frames. We then automatically segment [Kirillov et al. 2023], and transform objects in the source frame to match the pose of the corresponding objects in the target frame, using two motion models based on optical flow, designed to simulate the coarse edits a user would make using our Collage Transform interface. Since the images are now roughly aligned, we can train our model to convert the coarsely edited image into the ground truth target frame in an image-to-image [Isola et al. 2017; Saharia et al. 2022] fashion. This alignment procedure encourages the model to follow the user-specified layout at test time closely. Additionally, our model is carefully designed to transfer fine details from the reference source frame to preserve the identity and appearance of objects in the scene.

Our approach can produce plausible and realistic results from real user edits, and effectively projects coarse user edits into photorealistic images, confirming our insights on the advantages of using video data and a carefully designed motion model. Compared to the state-of-the-art, we show our outputs are preferred 89% of the time in a user study.

In short, our contributions are as follows:

- the Collage Transform, a natural interface for image editing that allows users to select and alter any part of an input image using simple transforms and that automatically turns the resulting edit into a realistic image,
- a new paired data generation approach to supervise the conversion from coarse edits to real images, which extracts pairs of video frames and aligns the input with the ground truth frame using simple motion models,
- a conditioning procedure that uses: 1. the warped image to guide layout in the diffusion generator, and 2. features from a second diffusion model to transfer fine image details and preserve object identity.
- a comprehensive analysis on the model's generalization to diverse editing tasks, like spatial editing, colorization, 3D transformation, NS perspective warping.

2 Related Work

Classical image editing. Classical image editing techniques offer various types of user controls to achieve diverse objectives. For instance, image retargeting aims to alter an image's size while preserving its key features and content [Avidan and Shamir 2007; Rubinstein et al. 2008; Simakov et al. 2008; Wang et al. 2008]. In contrast, image reshuffling rearranges an image's content based on user-provided rough layouts and imprecise mattes [Barnes et al. 2009; Cho et al. 2008; Simakov et al. 2008]. Image harmonization integrates objects from different images, adjusting their low-level statistics for a seamless blend [Jia et al. 2006; Sunkavalli et al. 2010]. A common thread in these classical image editing applications is the crucial role of user interaction, which provides the necessary control for users to realize their vision. Our method aligns with this approach, allowing users to reconfigure a photograph based on their preliminary edits.

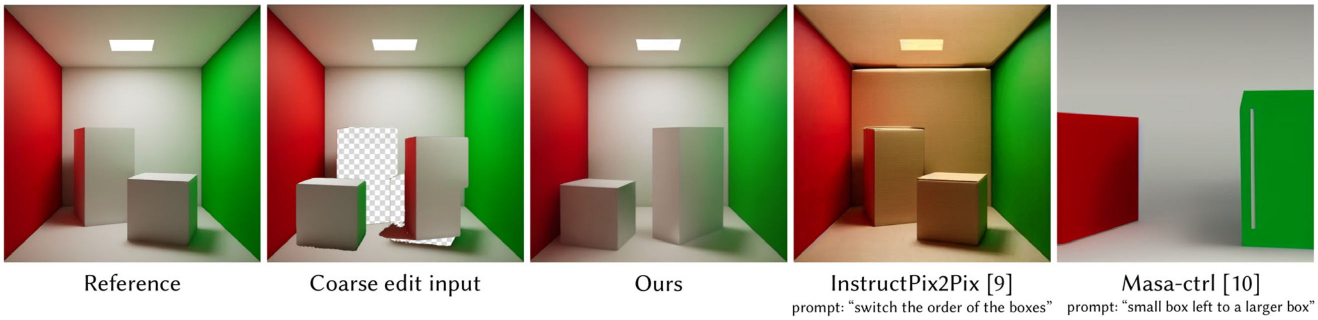

Reference
Coarse edit input
Ours
InstructPix2Pix [9]
prompt: “switch the order of the boxes”
Masa-ctrl [10]
prompt: “small box left to a larger box”

Fig. 2. **Comparison with text based control.** Our method directly takes a coarse user edit and makes it photorealistic. Our editing is both easy and precise, and our model can harmonize the global illumination appropriately. Text-based editing methods [Brooks et al. 2023; Cao et al. 2023] on the other hand, are not able to perform such edits, resulting in global appearance changes [Brooks et al. 2023] or unrealistic image [Cao et al. 2023].

Controllable image generation. The rapid advancement in photorealistic image generation has inspired researchers to adapt generative models for image editing tasks. Early efforts focused on high-level edits, like altering age or style, by manipulating latent space of Generative Adversarial Networks (GANs) [Abdal et al. 2019, 2020; Chai et al. 2021]. In a vein similar to our work, Generative Visual Manipulation [Zhu et al. 2016] involves projecting user-edited images onto the natural image manifold as approximated by a pre-trained GAN. The recent introduction of CLIP embeddings [Radford et al. 2021] has further propelled image editing capabilities, particularly through text prompts [Avrahami et al. 2022; Brooks et al. 2023; Crowson et al. 2022; Gal et al. 2022; Hertz et al. 2023; Kim et al. 2022; Mokady et al. 2023]. DragGAN [Pan et al. 2023] introduces fine control in image editing by using key-handles to dictate object movement, and follow-up works extend the drag-control idea to diffusion models [Luo et al. 2024; Mou et al. 2024; Shi et al. 2024]. Image Sculpting [Yenphraphai et al. 2024] takes a different approach by directly reposing the reconstructed 3D model of an object and re-rendering it, providing high level of control, but time consuming editing process unlike our Collage Transform interface that is designed to increase editing efficiency. CollageDiffusion [Sarukkai et al. 2024] guides text-to-image generation by using a collage as additional input. However, while CollageDiffusion focuses on controlling the generation of an image from scratch, we focus on using collage-like transformation to edit a reference image, and focus on preserving its identity.

Reference-based editing with generative models. To extend controllable image generation into editing real (non-generated images), one can invert the image back to noise [Song et al. 2021], and then guide the iterative denoising process to control the image generation[Bansal et al. 2023; Cao et al. 2023; Meng et al. 2022]. However, naively guiding the model without any grounding can lead to a loss in image identity. Prior work [Chen et al. 2024; Epstein et al. 2023; Yang et al. 2023] preserves the image identity through a pretrained feature extractor like CLIP [Radford et al. 2021] or DINO [Oquab et al. 2024], using a Control-Net like feature-injection [Chen et al. 2024; Zhang et al. 2023a], a dual-network approach [Cao et al. 2023; Hu 2024], or a combination of those approaches [Chen et al. 2024; Xu et al. 2024]. We adopt the dual-network approach, as it allows us to fully fine-tune the model and taylor it to our photorealistic editing task using our video-based dataset. AnyDoor [Chen et al. 2024] similarly uses video frames during training, but their focus is to recompose individual objects into the scene. On the other hand, we use video data to recompose the *entire scene* and use motion models designed for a convenient photo editing interface. Closest to our work is MotionGuidance [Geng and Owens 2024] that uses optical flow to guide editing the reference frame with diffusion guidance [Bansal et al. 2023] for a highly user-controllable edit. However, dense optical flow is difficult to manually provide for a user, unlike simple cut-and-transform edits in our Collage Transform. Furthermore, they rely on a prohibitively time-consuming guidance that take as long as 70 minutes for a single sample. On the other hand, our approach takes less than 5 seconds to fix up the user edit, allowing for interactive editing process.

3 Method

We aim to enable an image editing workflow in which users can select objects in a photograph, duplicate, delete or rearrange them using simple 2D transforms to produce a realistic new image (§ 3.1). We leverage image priors from pretrained diffusion models to project the coarsely edited image onto the natural image manifold, so the user can focus on specifying high-level changes without worrying about making their edits plausible (§ 3.2). Existing diffusion models can produce impressive results but often do so at the expense of control and adherence to the user input [Meng et al. 2022]. In particular, they tend to “forget” the identity and appearance of the edited object [Yang et al. 2023], and often only loosely conform to the user-specified pose [Chen et al. 2024]. Our method addresses these issues using two mechanisms. First, our synthesis pipeline is a conditional diffusion model (§ 3.4) that follows the coarse layout defined by the user, and transfers fine details from the reference input image (§ 3.3) to best preserve the original image content. Second, we construct a supervised dataset exploiting object motion from videos to finetune the pretrained model to explicitly encourage content preservation and faithfulness to the input edit (§ 3.5).

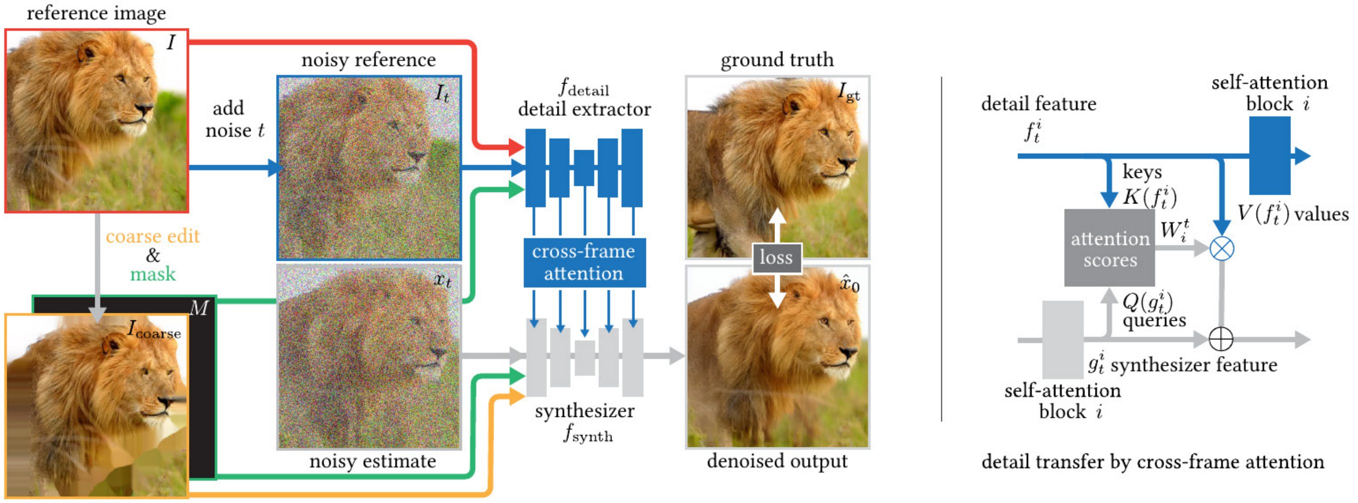


Fig. 3. **Overview.** Our pipeline (left panel) uses two parallel models, a detail extractor (top) and a synthesizer (bottom), to generate a realistic image from a coarse user edit and a mask recording missing regions caused by the edit. The detail extractor processes the reference image, a noisy version of the reference and the mask, to produce a set of features that guide the synthesis and allow us to preserve the object appearance and fine details from the reference image. The synthesizer generates the output conditioned on the mask and coarse edit. The features from the detail extractor are injected via cross-attention at multiple stages in the synthesizer, in order to transfer details from the input. Both models are finetuned on our paired dataset. The right panel shows a detailed view of our cross-attention detail transfer operator. Photos sourced from ©Adobe Stock.

3.1 Specifying coarse structure with simple transforms

Starting from an image $I \in \mathbb{R}^{3hw}$, $h = w = 512$, we run an automatic segmentation algorithm [Kirillov et al. 2023] to split the image into non-overlapping semantic object segments. The user can edit this image by applying 2D transformations to the individual segments (e.g., translation, scaling, rotation, mirroring). Segments can also be duplicated or deleted. Figure 1 illustrates this workflow. We keep track of holes caused by disocclusions when moving the segment in a binary mask $M \in \{0, 1\}^{hw}$, and inpaint them using a simple algorithm [Bertalmío et al. 2001]. We denote the resulting, coarsely edited image by $I_{\text{coarse}} \in \mathbb{R}^{3hw}$.

We operate in an intermediate latent space for efficiency, but our approach also applies to pixel-space diffusion. With a slight abuse of notation, in the rest of the paper $I, I_{\text{coarse}} \in \mathbb{R}^{3hw}$, with $h = w = 64$ refer to the input and coarse edit after encoding with the latent encoder from Stable Diffusion [Rombach et al. 2022], and M the mask downsampled to the corresponding size using nearest neighbor interpolation. The latent triplet $(I, I_{\text{coarse}}, M)$ forms the input to our algorithm.

3.2 From coarse edits to realistic images using diffusion

We want to generate a realistic image that (1) follows the large-scale structure defined by the coarse user edit, and (2) preserves the fine details and low-level object appearance from the unedited image, filling in the missing regions. Our pipeline, illustrated in Figure 3, uses 2 parallel models.

The first, which we call *synthesizer* f_{synth}, generates our final output image. The second model, which we name *detail extractor* f_{detail}, transfers fine-grained details from the unedited reference image I to our synthesized output during the diffusion process. It modulates the synthesizer by cross-attention at each diffusion step, an approach similar to Masa-Ctrl [Cao et al. 2023] and AnimateAnyone [Hu 2024]. Both models are initialized from a pretrained Stable Diffusion v1.4 model [Rombach et al. 2022], and finetuned on our paired dataset (§ 3.5). Since we have a detailed reference image I to guide the synthesis, we do not need the coarse semantic guidance provided by CLIP, so we remove the CLIP cross-attention from the model.

Let $T \in \mathbb{N}^*$ be the number of sampling steps, and $\alpha_0, \ldots, \alpha_T \in \mathbb{R}^+$ be the alphas of the diffusion noise schedule [Ho et al. 2020]. Starting from an image $x_0 \in \mathbb{R}^{3hw}$, the forward diffusion process progressively adds Gaussian noise, yielding a sequence of increasingly noisy iterates:

$$x_t \sim \mathcal{N}\left(\sqrt{\alpha_t}\, x_{t-1}; (1-\alpha_t)\mathbf{I}\right). \tag{1}$$

The base diffusion model f is trained to reverse this diffusion process and synthesize an image iteratively, starting from pure noise $x_T \sim \mathcal{N}(0, I)$. The synthesizer and detail extractor in our approach make a few modifications to this base model, which we describe next.

3.3 Extracting details from the reference image

During inference, at each time step t, we start by extracting a set of features F_t from the reference image using f_{detail} (Figure 3, top). These features will guide the synthesis model and help preserve realistic image details and object identity. Since we use a pretrained diffusion model as a feature extractor, we start by adding noise to the reference unedited image:

$$I_t = \sqrt{\bar{\alpha}_t}\, I + (1 - \bar{\alpha}_t)\epsilon, \tag{2}$$

with $\epsilon \sim \mathcal{N}(0, \mathbf{I})$, $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$. We extract the feature tensors immediately before each of the $n = 11$ self-attention blocks in the model:

$$F_t := [f_t^1, \ldots, f_t^n] = f_{\text{detail}}([I_t, I, M]; t), \tag{3}$$

where $[\cdot]$ denotes concatenation along the channel dimension. Our feature extractor also takes as input the clean reference image since it is always available for detail transfer and mask, so the model knows which regions need inpainting. Since the pretrained model only takes I as an input, we modify the first layer at initialization by padding its weight with zeros to accept the additional channel inputs. Using a noisy version of the reference ensures the extracted features are comparable to those in the cross-attention operators of the synthesis model.

3.4 Image synthesis by detail transfer to the coarse edit

The synthesizer f_{synth} generates the final image, conditioned on the detail features F_t. Unlike standard diffusion sampling, we do not start from pure Gaussian noise. Instead, inspired by SDEDit [Meng et al. 2022], we start from an extremely noisy version of the coarsely edited image:

$$x_{T-1} = \sqrt{\bar{\alpha}_{T-1}} I_{\text{coarse}} + (1 - \bar{\alpha}_{T-1})\epsilon, \tag{4}$$

so we effectively bypass the first denoising step with adding noise to the edit rather than pure noise. This initialization circumvents a commonly observed issue where diffusion models struggle to generate images whose mean and variance deviate from the normal distribution. This is particularly important in our setup as the user input can have arbitrary color distribution, and we need the model to match the user input. This has been shown to stem from a domain gap between training and sampling [Guttenberg 2023; Lin et al. 2024]: the model never sees pure noise during training, but a sample from the normal distribution is the starting point for inference. Our latent initialization addresses this issue by directly bridging the gap between training and inference. In Fig. 5 we show how initializing with pure noise leads to a low contrast image, while our initialization allows the model to preserve the input color range well. For subsequent steps during inference, we update the current image estimate x_t at each time step t, using the following update rule:

$$x_{t-1} = f_{\text{synth}}([x_t, I_{\text{coarse}}, M]; t, F_t). \tag{5}$$

We provide the mask and coarse edit as conditions by simple concatenation, but because we need to extract fine details from the reference, we found passing the reference information by cross-attention with the features F_t provided richer information. Again, we extend the weight tensor of the first convolution layer with zeros to accommodate the additional input channels.

Detail transfer via cross-attention. We use the intermediate features $F_t = [f_t^1, \ldots, f_t^n]$, extracted *before* the detail extractor's self-attention layers to transfer fine image details from the reference image to our synthesis network by cross-attention with features $[g_t^1, \ldots, g_t^n]$ extracted *after* the corresponding self-attention layers in the synthesis model. See the right panel of Fig. 3 for an illustration, where Q, K, V are linear projection layers to compute the query, key, and value vectors, respectively, and W_i^t is the matrix of attention scores for layer i, at time step t. The feature tensors g_t^i, f_t^i are 2D matrices whose dimensions are the number of tokens and feature channels, which depend on the layer index i.

3.5 Training with paired supervision from video data

We train our model on a new dataset obtained by extracting image pairs from videos to reconstruct a ground truth frame given an input frame and a coarse edit automatically generated from it. Our insight is that motion provides useful information for the model to learn how objects change and deform. Videos let us observe the same object interact with diverse backgrounds, lights, and surfaces. For example, skin wrinkles as a person flexes their arm, their clothes crease in complex ways as they walk, and the grass under their feet reacts to each step. Even camera motion yields disocclusion cues and multiple observations of the same scene from various angles.

Concretely, each training sample is a tuple $(I, I_{\text{gt}}, I_{\text{coarse}}, M)$, where I and I_{gt} are the input and ground-truth frames, respectively, extracted from the video with a time interval sampled uniformly at random from $\{1, \ldots, 10\}$ seconds between them. However, if the computed flow between the two frames was too large (at least 10 percent of the image has a flow magnitude of 350 pixels), we resample another pair. This is to ensure that the warping produces reasonable outputs. We construct the coarse edit I_{coarse} and corresponding mask M using an automated procedure that warps I to approximately match I_{gt}, in a way that mimics our Collage Transform interface. For this, we use one of 2 possible editing models: a flow-based model and a piecewise affine motion model (Fig 4).

Flow-based editing model. We compute the optical flow using RAFT-Large [Teed and Deng 2020] for each consecutive pair of frames between I and I_{gt} and compose the flow vectors by backward warping the flow to obtain the flow between the two frames. We then forward warp I using softmax-splatting [Niklaus and Liu 2020], to obtain I_{coarse}, which roughly aligns with the ground truth frame. The forward warping process creates holes in the image. We record these holes in the mask M. Our model needs to learn to inpaint these regions and those we have no correspondence (e.g., an object appearing in the frame). Using flow-based warping helps the model learn to preserve the identity of the input, rather than always hallucinating new poses and content.

Piecewise affine editing model. Optical flow warping can sometimes match the ground truth too closely. As we discuss in Section 4 and Figure 10, training the flow-based editing model only can limit the diversity of our outputs, leading to images that do not deviate much from the coarse edit. Flow-warping is also reasonably distinct from our expected test-time user inputs (§ 3.1). Our second editing model addresses these issues by transforming the reference frame as a *collage*. We compute a depth map for the image using MiDaS [Ranftl et al. 2021, 2020] and automatically segment the image using SegmentAnything [Kirillov et al. 2023].

We then transform each segment using the affine transformation that best matches the optical flow for this segment, compositing them back to front according to each segment's average depth. For the image regions that are not segmented, we use the optical flow warping scheme described above. Due to the coarser alignment, the model learns how different parts of the scene interact in a realistic setting, like associating objects and their shadows and reflections.

We use a dataset consisting of 12 million 5-10 second clips of stock videos, and we filter out keywords that indicate static scenes

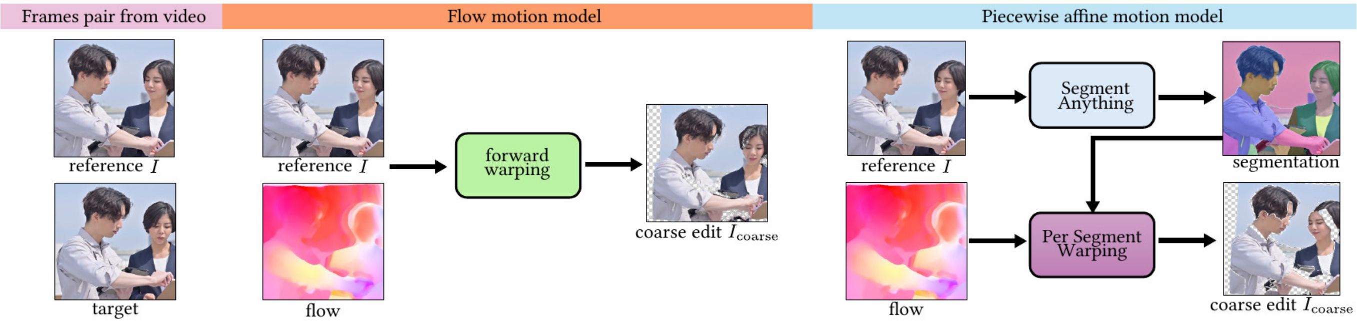


Fig. 4. **Dataset synthesis.** Our dataset synthesis pipeline starts by sampling two frames from a video. We set one frame to be the reference, which we warp by one of our motion models, and set the other frame to be the target that we warp the reference towards. To generate aligned training pairs, we use 2 motion model to warp the reference frame towards the ground truth (target frame). The first model uses optical flow (left). It provides the most accurate alignment but does not correspond to what the user would provide during inference. This motion model encourages adherence of our model's output to the layout specified using the coarse edit. To generate training pairs closer to the collage-like user inputs, we use a second motion model (right). For this, we segment everything in the image [Kirillov et al. 2023] and apply similarity transforms to each segment, estimated from the flow within the segment. Figure 10 analyses the impact of these motion models on the final result. Photos sourced from ©AdobeStock.

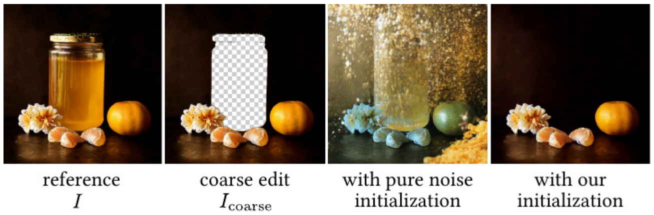

reference I
coarse edit I_{coarse}
with pure noise initialization
with our initialization

Fig. 5. **Effects of Latent Initialization.** Starting from pure noise, as is standard practice, the model struggles to generate images with deep blacks and synthesizes nonsensical content to keep the image's mean and standard deviation close to the starting Gaussian noise. This is a known issue with current diffusion models [Guttenberg 2023; Lin et al. 2024]. Instead, during inference, we initialize the latent to the warped image with a very large amount of Gaussian noise before running the diffusion. This simple change makes a drastic difference and lets the model preserve the image content. Photo sourced from ©Unsplash.

or synthetic/animated videos, as we are only interested in photorealistic videos and also highly dynamic scenes where the motion is too large (like car racing). For each valid clip, we sample one pair and compute the warping using both motion models. After filtering for desired motion, we use 2.5 million clips, creating a dataset consists of 2.5 million samples for each motion model, making a total of 5 million training pairs.

3.6 Implementation details

We finetune both models jointly for 120,000 steps with a batch size of 32, using Adam [Kingma and Ba 2014], with a learning rate of 1×10^{-5} on 8 NVIDIA A100 GPUs, which takes approximately 48 hours. Note that this is considerably more efficient than recent compositing work [Yang et al. 2023] that uses 64 NVIDIA V100 GPUs for 7 days. We hypothesize that the stronger input signal helps the model converge faster. We use a linear diffusion noise schedule, with $\alpha_1 = 0.9999$ and $\alpha_T = 0.98$, with $T = 1000$. During inference, we sample using DDIM for 50 denoising steps.

4 Experimental Results

We evaluate our method qualitatively on a set of user edits to demonstrate real-world use cases, as well as on a held-out validation dataset created in the same way as our training set (§ 3.5) for quantitative evaluation. *In the appendix, we show additional applications of the model on editing tasks beyond spatial recomposition, like colorization, perspective editing, and 3D transformation.*

Our model is trained on a synthetically-generated dataset. We validate that it generalizes to real user edits using a prototype interface illustrating our segment-based editing workflow. The user can segment any part of the image and transform, duplicate, or delete it. We provide a video demonstrating this editing interface in the supplementary materials, and contrast the user's edit with the model's output. To the best of our knowledge, no previous work focuses exactly on our use case (photorealistic spatial edits), so we adapt closely related techniques to our problem setting for comparison. Specifically, we compare to the following baselines:

1. SDEdit [Meng et al. 2022]: a general text-based editing method that trades off the adherence to the input image and the faithfulness to the text. This is the most general method we compare against, as we can directly provide it with the coarse user edit and a generated caption. Since SDEdit can take the coarse edit directly as an input, we emphasize it the most in our comparisons.
2. DragDiffusion [Shi et al. 2024]: a drag-based editing model that takes source-target key-handles to move parts of the object for re-posing.
3. InstructPix2Pix [Brooks et al. 2023]: a text-based editing method that follows an instruction-like style captions.
4. MasaCtrl [Cao et al. 2023]: a text-based editing method that achieves a consistent identity preservation through a self-attention mechanism. However, it relies on DDIM inversion as an essential step, which can compromise the method's robustness.

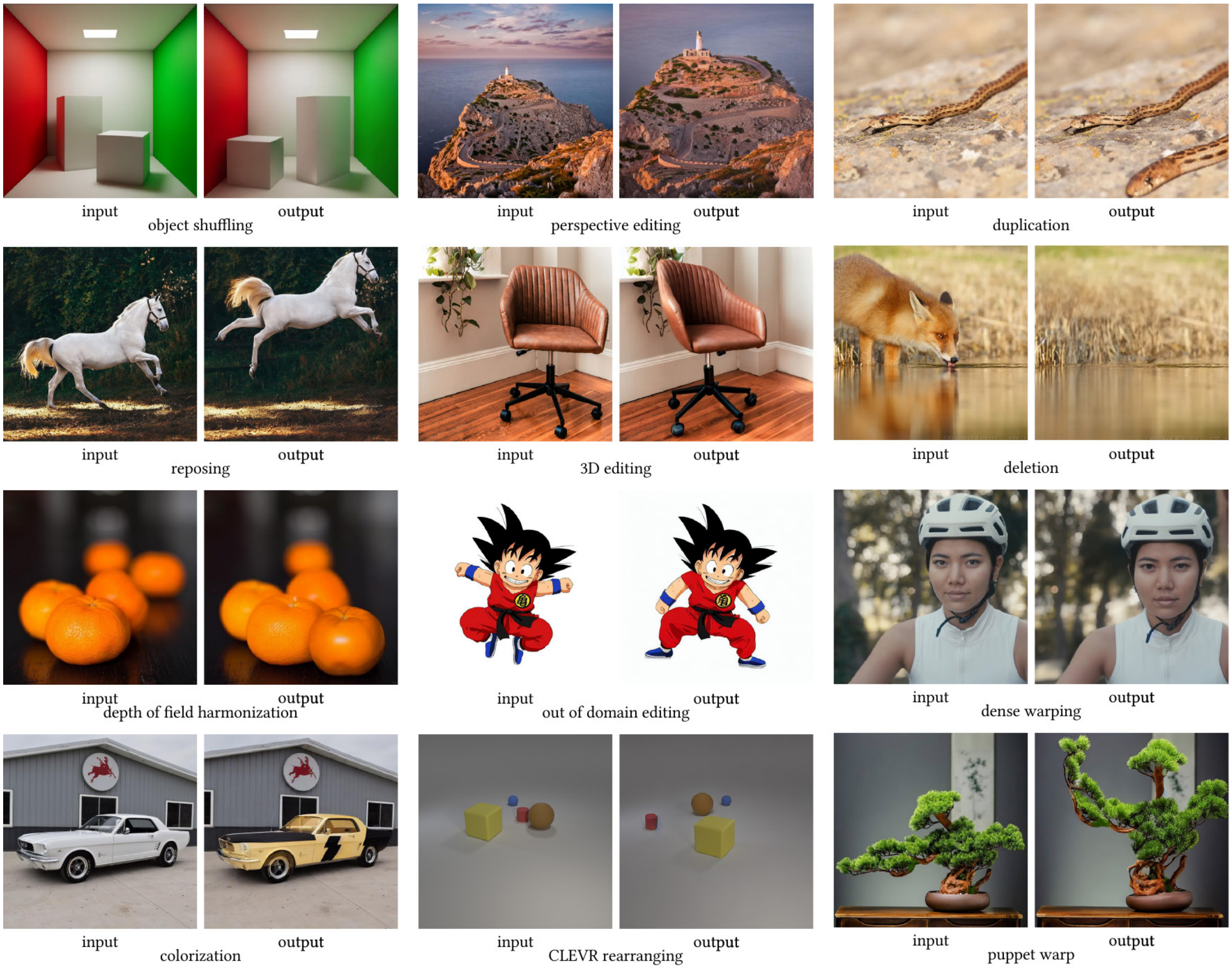


Fig. 6. **MagicFixup applications.** Our method can generalize to a large number of different applications. We just need to create a coarse edit, and MagicFixup realistically clean up the edit. Here we show a selection of tasks we used MagicFixup to achieve, like image recomposition, perspective editing, reposing, 3D transformation, and colorization. We show the coarse edits for these examples throughout the appendix. Photos sourced from ©Unsplash and CC licensed images.

Adapting the baselines. We convert our inputs to the interface expected by these baselines for comparison. SDEdit requires choosing a strength parameter dictating the amount of noise added to the input and trades off between faithfulness and unconstrained synthesis. We set the strength to 0.4 in all experiments, i.e. we start at 40% of the way through the diffusion process, adding the corresponding level of noise to I_{coarse}. Unlike ours, their model expects a text input, which we automatically compute using BLIP [Li et al. 2022]. We use the same generated caption with the other text-based methods (MasaCtrl and InstructPix2Pix) for the quantitative evaluation on our large corpus. For the qualitative evaluation on user edits, we choose a caption that describes the user edit to the best of our abilities. However, we note that text description of spatial edits is inherently ambiguous (which is the motivation of our proposed method).

To compare with DragDiffusion [Shi et al. 2024], we record the segment motion in our user interface, compute the motion vectors for each pixel, and use this information to automatically create the keypoint-handles input needed by DragDiffusion.

4.1 Quantitative evaluation

While the task of image editing is inherently subjective, a natural task is to evaluate our method on edits generated from videos using the motion models we discuss earlier. We use a held-out split of our dataset, and evaluate our method against the baselines on the performance at reconstructing the target frame. In Table 1 we show

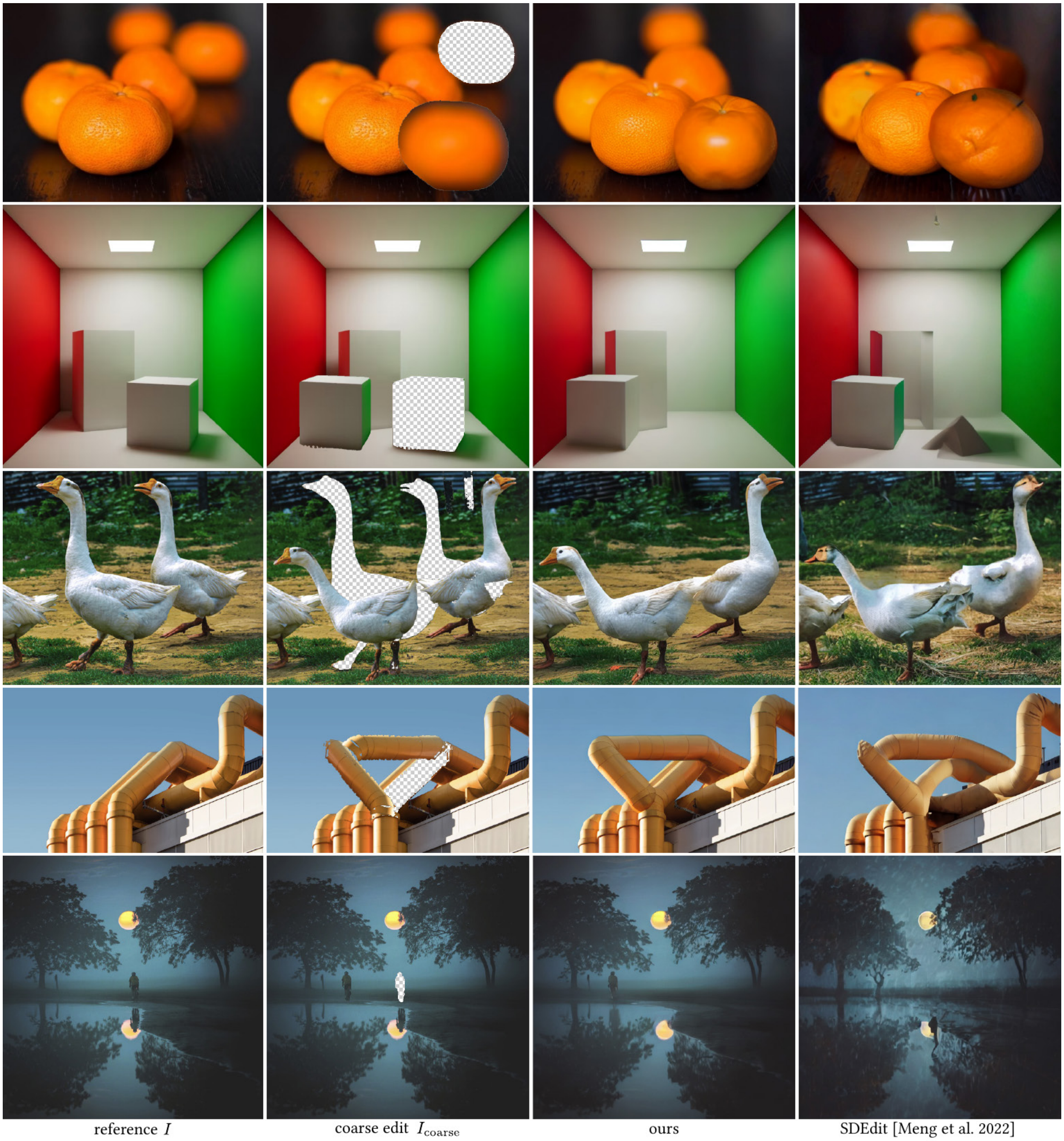


Fig. 7. **Spatial editing.** We show example of scene recompositing. Our model is capable of synthesizing compelling effects that harmonize realistically with the rest of the image such as: changing the depth of field (row 2), adjusting the global illumination (green reflection on the cube, row 3), and removing or adding reflections (row 6). Photos sourced from ©Unsplash.

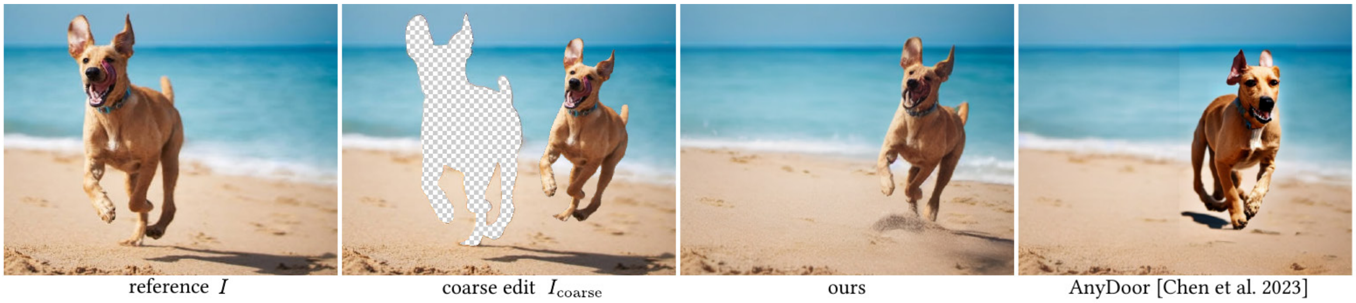


Fig. 8. **Comparison to Anydoor [Chen et al. 2024].** Anydoor was trained to insert objects from one image to another. We can repurpose their approach for our image editing task by using the same image as source and target. Their approach does not preserve the dog's identity in this example. AnyDoor also does not harmonize the lighting properly (the sun direction and shadows are wrong), the image is too bright, and some blending seams are visible. On the other hand, our output shows natural shadows and plausible contacts with the ground, adding realistic moving sand consistent with the pose. Photo sourced from ©Unsplash.

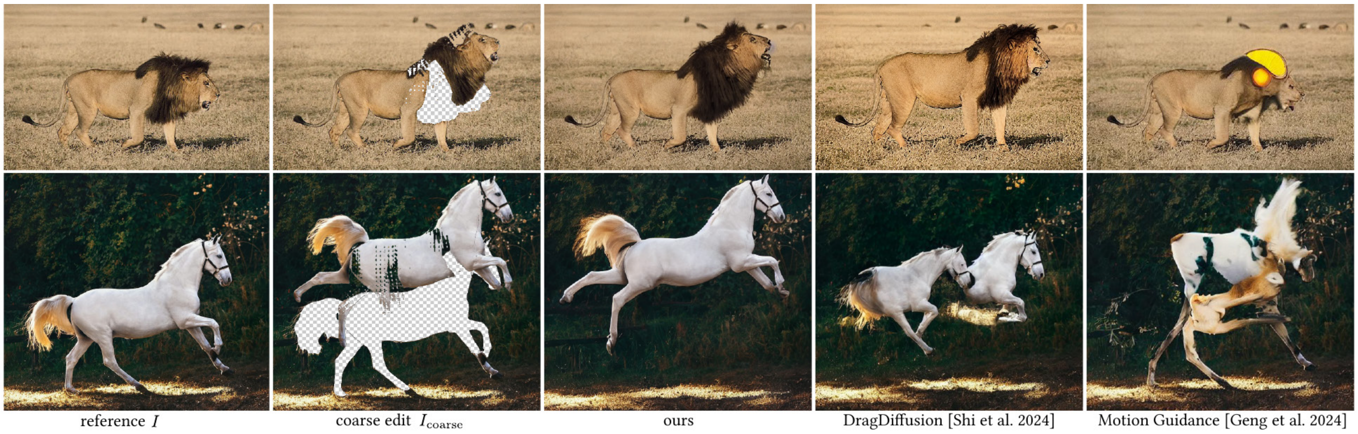

reference I
coarse edit I_{coarse}
ours
DragDiffusion [Shi et al. 2024]
Motion Guidance [Geng et al. 2024]

Fig. 9. **Comparison with DragDiffusion.** We use the Drag Diffusion [Shi et al. 2024] to generate the results in the right column. We seed dragging control points this method expects for each of the modified image segments, and displace them using the same affine transform used to produce our coarse edit (second column). DragDiffusion generates fairly conservative image edits, and collapses with more drastic reposing edits. However, our method successfully handles wide range of reposing levels. Photos sourced from ©Unsplash.

that our method significantly outperforms the baselines on reconstruction metrics. We also computed the flow error, by comparing the RAFT optical flow between the reference and target, and the flow between the reference and the output. Intuitively, the smaller the flow error, the more that the method is faithful to the user edit. We show that our method is significantly outperforming all the baselines across all metrics. The second best method is DragDiffusion, which is likely due to the fact that it accepts a form of spatial edit as an input (through drag handles), and the LoRA optimization on each image to preserve identity. Across all text-based methods, SDEdit performs the best in general likely due to the fact that it can directly accept the user edit as an input rather than purely relying on the caption to perform the edit.

Table 1. **Quantitative evaluation.** Perceptual loss, SSIM, and flow error on a held-out validation subset of our video dataset. Our method significantly outperforms all the baselines across all metrics.

Method	Motion Model	LPIPS ↓	SSIM ↑	Flow (px) ↓
MasaCtrl [Cao et al. 2023]	-	0.44	0.49	23.1
IP2P [Brooks et al. 2023]	-	0.47	0.42	24.9
DragDiff* [Shi et al. 2024] [1]	Piecewise affine	0.26	0.58	16.9
	Flow-based	0.26	0.59	15.8
SDEdit [Meng et al. 2022]	Piecewise affine	0.40	0.57	24.7
	Flow-based	0.37	0.61	22.4
Ours	Piecewise affine	**0.21**	**0.70**	**5.33**
	Flow-based	**0.16**	**0.78**	**3.06**

4.2 Evaluation on user edits

The training dataset we use for MagicFixup allows for a host of different applications. In Fig. 6 we highlight 12 different example applications we generated using MagicFixup, which includes edits

[1] We find that DragDiffusion fails to produce an output on edits with large motion, so we restricted this evaluation to the subset where it is able to produce an output. Note that the ordering of the methods remains the same if we restrict all methods to the same subset.

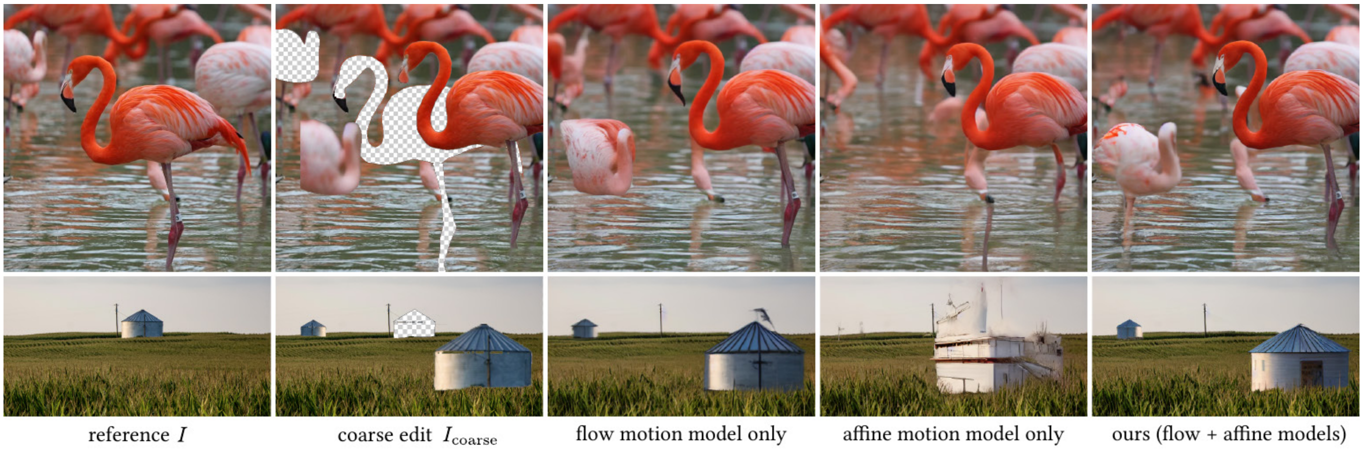


Fig. 10. **Motion models ablation.** We compare how the 2 motion models we use to create our coarse edits (column 2) during training affect the model's behavior. If we warp the reference frame (column 1) using the flow only (column 3), the model learns how to harmonize the edges of the edited regions, but remains very conservative and does not add much details to increase realism. On the other extreme, if we only use the piecewise affine motion model (column 4), the model learns to hallucinate excessively, losing its ability to preserve object identity. Our full solution trains with both motion models (column 5) to increase the model versatility, allowing the model to generate realistic details while still maintaining good adherence to the user input. Photos sourced from ©Unsplash.

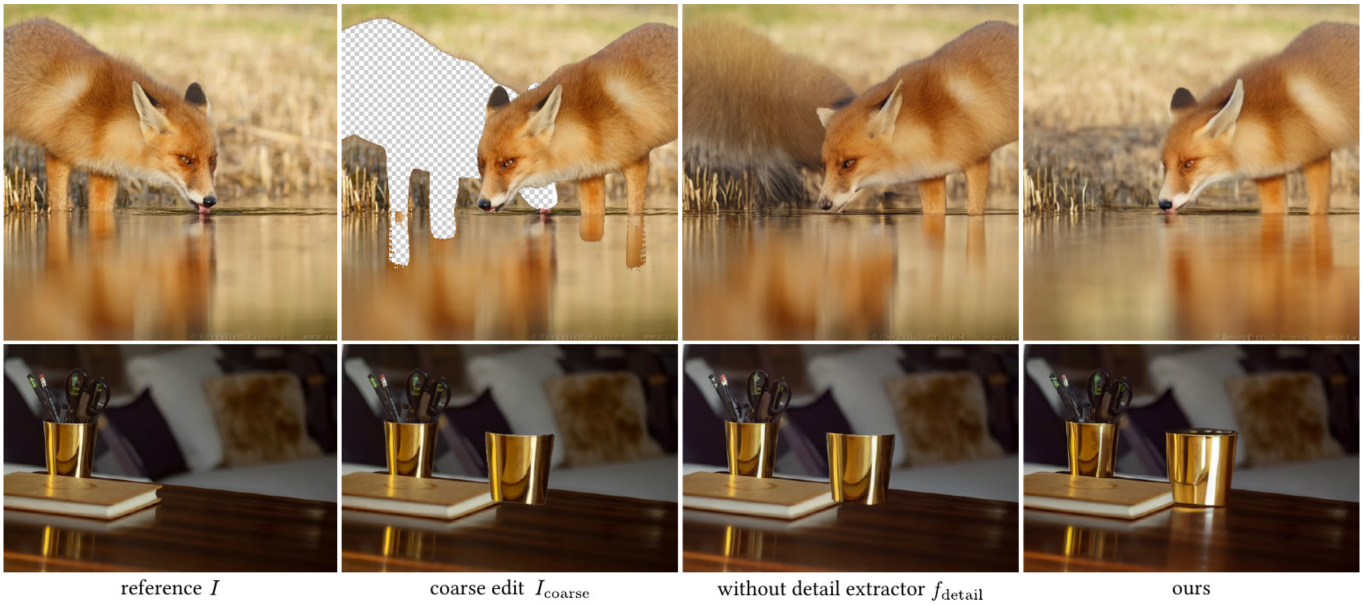

reference I
coarse edit I_{coarse}
without detail extractor f_{detail}
ours

Fig. 11. **Architecture ablation.** Without the detail extractor branch and using CLIP to extract the reference features (3rd column), the model struggles with spatial reasoning as it cannot access the grounding of the original reference image (1st column). This ablation's outputs are overly conservative, not steering too far away from the coarse edit (2nd column). Our full model produces much more realistic edits (4th column), with harmonious shadows and object-background contact. It refines object boundaries and synthesizes plausible reflections. Photos sourced from ©Roeselien Raymond and ©Unsplash.

outside of the training data, like perspective edits, 3D transformations, and even colorization. In this section, we discuss edits generated using our Collage Transform interface and compare against pose-editing baselines. We further highlight additional applications beyond image recomposition in the appendix.

Collage transform editing. Using our user interface, we created a collection of edits that spatially recompose photos. In Fig. 7 we show how our model adds realistic details to objects moved to a region of sharper focus, snaps disconnected objects together, and resynthesizes shadows and reflections as needed. Another natural baseline for spatial recomposition is inpainting an object and reinserting it in the image. We use AnyDoor [Chen et al. 2024] as the insertion method, and compare the recomposition result. In Fig. 8, we used our model to delete the dog (and automatically remove the shadow), and then re-inserted the dog using AnyDoor. The dog's identity underwent significant changes, and AnyDoor does not harmonize the composite with the ground. It also does not completely remove the halo caused by the inpainting mask in the destination region. In contrast, our model synthesizes a coherent output without discontinuity artifacts. We also used AnyDoor in duplicating the snake in Fig. 1, and we show that AnyDoor has a loss of identity on the snake, while our method correctly introduces some defocus blur to adjust for the reference's shallow depth of field. We compare against text-only editing methods in Fig. 2, and show that InstructPix2Pix [Brooks et al. 2023] only alters the apperance without following the spatial edit instruction prompt, and MasaCtrl [Cao et al. 2023] completely loses the input identity due to the failure of DDIM inversion. In the appendix, we also show additional comparisons against text-only methods.

Image reposing. Since we allow the user to edit the image by selecting segments of arbitrary size, the user can re-pose objects by selecting sub-parts and applying an affine transformation on them, effectively animating the object. We compare our method to DragDiffusion [Shi et al. 2024] that uses drag handles, and Motion Guidance [Geng and Owens 2024] that uses flow to guide the diffusion sampling to follow the user edit. To ensure a fair comparison, we keep track of dense pixel correspondence in our Collage Transform user interface for the user edit. Using the dense correspondence maps, we can directly generate the drag handles and flow inputs these baselines require. In Fig. 9, DragDiffusion moves the lion's body higher up, which loosely aligns with the user edit, but is inconsistent with the user's intent of only moving the head. This example highlights how a non-interactive point-dragging interface can be at odds with the user's desired output, because it does not provide a good preview of what the model would generate before running it. Our Collage Transform interface is more immediate, and our coarse edit aligns with the final output. On the other hand, despite having dense flow, Motion Guidance completely fails to follow the user edit as the test time optimization process is unreliable. In the second example, DragDiffusion collapses, likely because the user input is complex and goes beyond a minimal displacement of the subject that it can handle, and Motion Guidance lifts the horse up in the air but fails to keep it in one piece.

Note that both DragDiffusion and Motion Guidance require a costly test-time optimization for each input. On NVIDIA A100, DragDiffusion takes approximately 2 minutes, and Motion Guidance takes nearly an hour for a single input. In contrast, our method only requires the feed-forward sampling, taking approximately 5 seconds.

Perceptual user study. To evaluate the realism of our editing, we conducted a user study comparing the quality of our edits against three baselines: SDEdit [Meng et al. 2022], InstrictPix2Pix [Brooks et al. 2023], and Masa-Ctrl [Cao et al. 2023]. We used 11 diverse photo edits, with 21 students participating and voting for all pairs of images. For each pair, we provided the users with the reference image as well as the *intended* user edit, and asked for each sample the following "For the following edit, which of those images do you find a more realistic result?" in a 2-alternative forced-choice (2AFC) format. In Fig. 12 we show the aggregated votes for our method against each baseline. We see that our method is overwhelmingly preferred. For each baseline comparison we have 231 total votes, and only 14 votes were for SDEdit [Meng et al. 2022], 10 votes were for InstructPix2Pix [Brooks et al. 2023], and 19 votes for MasaCtrl [Cao et al. 2023]. We show comparison results against those baselines in Appendix D, and we encourage the reader to compare the results directly.

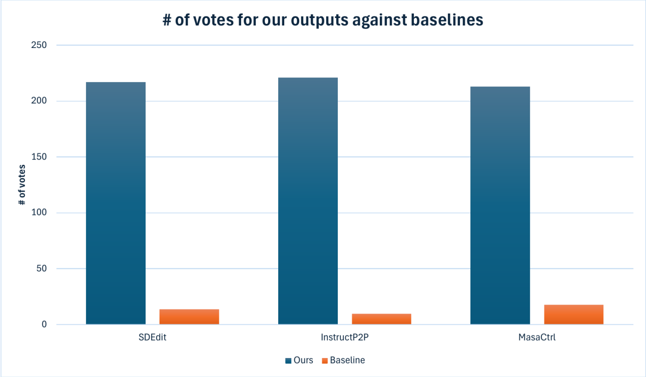


Fig. 12. **User study results.** We asked 21 users to compare the editing results from our method and the outputs using baselines. For each baseline, the user directly compares between our method's output and the baseline's. Overall, the users overwhelmingly preferred our method. Out of 231 votes, only 14 votes were for SDEdit [Meng et al. 2022], 10 votes were for InstructPix2Pix [Brooks et al. 2023], and 19 votes for MasaCtrl [Cao et al. 2023]

4.3 Ablation studies

In this section, we evaluate the role that different motion models play, as well as the importance of cross-reference attention.

Motion models ablation. Intuitively, training the model only on flow-warped images would prevent the model from learning to synthesize drastic changes, since flow-warping tends to be well-aligned around the edges. On the other hand, using the piecewise-affine motion model requires the model to adjust the pose of each segment (and learn to connect them together nicely), which forces the model to only use the input as a coarse conditioning. In Fig. 10, we show that the behavior of the model trained on different motion models is consistent with our intuition, where the model trained on flow-only preserves the content and refines the edges, while the model trained only on the piecewise-affine model struggles with preserving identity. On the other hand, the model trained on different motion models falls in the sweet-spot where it addresses user edits faithfully while adding content as needed.

Architecture ablation. Prior work relies on using Image-CLIP embeddings or DINO features to encode the information of the content being inpainted or inserted into the image [Chen et al. 2024; Yang et al. 2023]. The CLIP features are a reminiscent of the way Stable-Diffusion is trained with cross-attention with text CLIP embeddings. However, as we believe that CLIP features only carry semantic features that are too weak to pass useful information about the

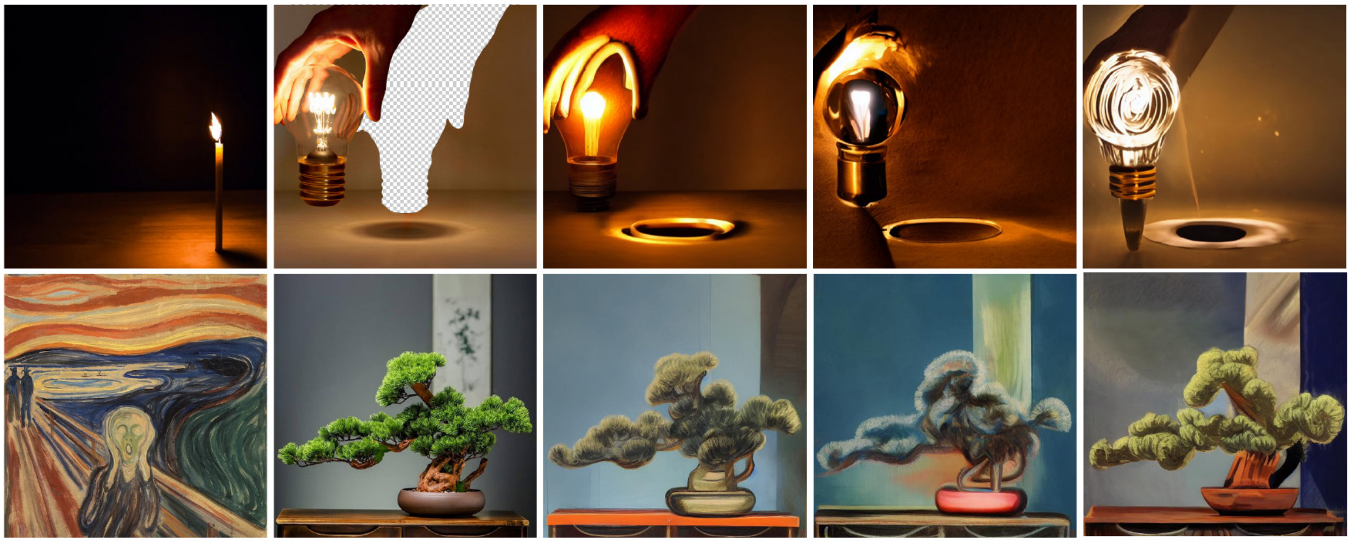


Fig. 13. **Ablating reference input.** When passing an arbitrary image as a reference to the detail extractor network independent of the input of the synthesis network, we find an effect similar to style transfer. The model preserves the spatial structure of the "coarse edit" input while maintaining the style and global appearance of the "reference." This behavior likely contributes to the model robustness in generalization to new domains and processing new types of edits. The scream painting is now in public domain, and remaining photos were sourced from ©Unsplash.

Table 2. **Quantitative ablations.** Perceptual loss on a held-out validation set from our video dataset.

Model & Training Data	**Test Data**	**LPIPS ↓**
Piecewise affine	Piecewise affine Flow-based	**0.231 ± 0.007** 0.220 ± 0.007
Flow-based	Piecewise affine Flow-based	**0.229 ± 0.007** **0.190 ± 0.007**
Both motion models (no cross-ref attn)	Piecewise affine Flow-based	0.327 ± 0.007 0.269 ± 0.008
Both motion models (Full method)	Piecewise affine Flow-based	**0.231 ± 0.007** **0.196 ± 0.007**

reference structure. We use a cross-reference mechanism, similar to Masa-Ctrl [Cao et al. 2023], and unlike prior work, we completely remove the CLIP cross-attention layer. To validate our design decision, we compare using CLIP image embedding of the reference for cross-attention as opposed to the cross-reference-attention we propose. We observe that when relying only on CLIP embeddings, the model struggles in harmonizing the edited regions as shown in Fig. 11. We find that the ablated model is conservative, and cannot address secondary effects like reflections.

Quantitative comparison. We evaluate our ablations on a held-out validation dataset from our video dataset. In the table on the right, Table 2, we show that the model trained with flow-data and affine-motion are the top performers on perceptual loss on both types of test and that dropping the cross-reference attention and replacing it with CLIP embedding causes a severe drop in performance.

Detail extractor inputs To better understand how the model is utilizing the reference image in preserving the image details, we ask, what would happen if the provided "reference image" is completely independent from the provided "coarse edit?" Note that this case never occurs in training, but by modifying the inputs, we can gain insights on the inner workings of the model. In Fig. 13 we show how the model behaves when provided an unrelated reference and edited images, and highlight diverse samples from the model. We notice that the model preserves the "style" and global appearance of the reference image, while preserving the spatial structure of the coarse edit. Intuitively, the reference image provides a sample of a clean image is supposed to look like, while the coarse edit provides the guidance of the spatial structure that the user intends to keep. This also touches on recent style transfer work [Hertz et al. 2024] that achieves style transfer through shared attention layers, which is similar to the effect we see here where the model transfers the style of the reference to the coarse edit. In Appendix E we provide additional quantitative analysis ablating the inputs of the detail extractor and synthesis UNets, like the disocclusion masks and the denoising timestep embedding.

4.4 Generalization beyond real photos

While we only use video data of real videos, and filter out the majority of non-photorealistic videos in our training data, we explore the model's ability to generalize to completely new domains. In Fig. 14 we show the model's ability to generalize to new domains, like cartoons and vector art. We find that the model smoothly re-connects any disconnected parts of the image, and correctly re-synthesizes the art's outline that was lost in the editing process. On the other hand, we find that SDEdit fails on all of those domains.

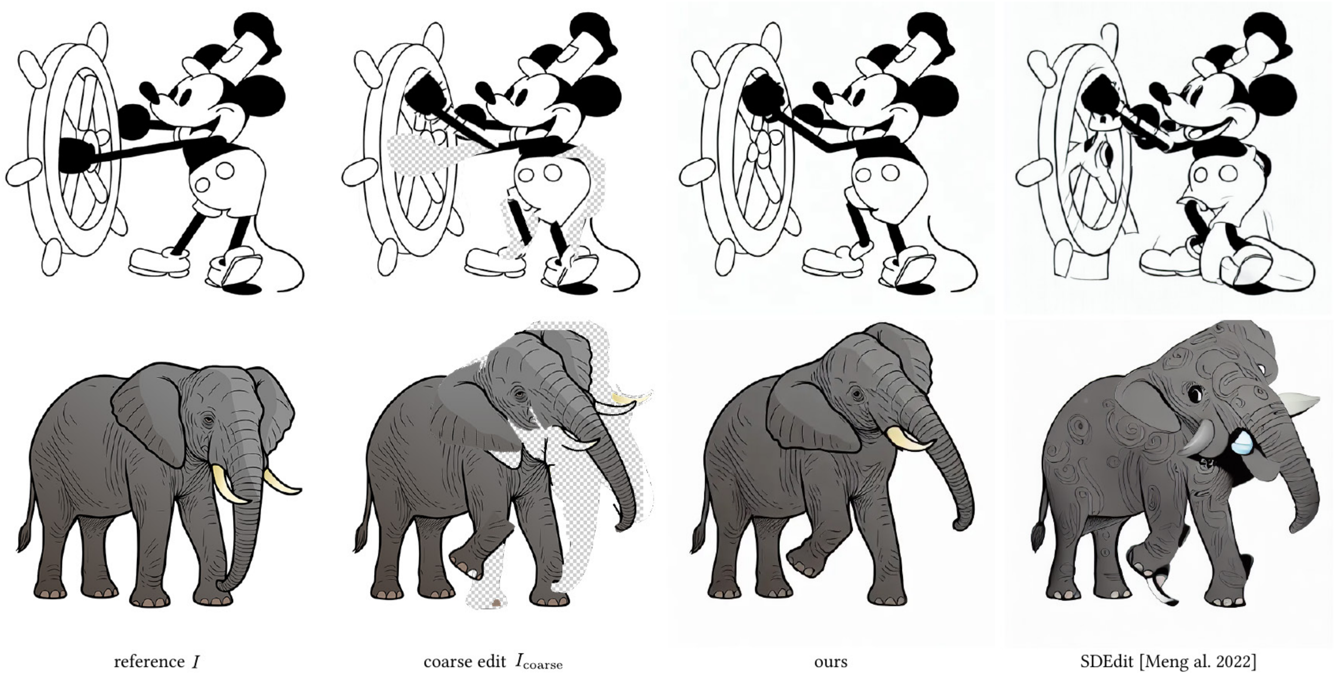


Fig. 14. **Beyond real photos.** Despite training the model on exclusively real videos, we find that the model can generalize to new domains beyond the training data, like cartoons and vector art. Photos are using public domain materials and CC licensed images.

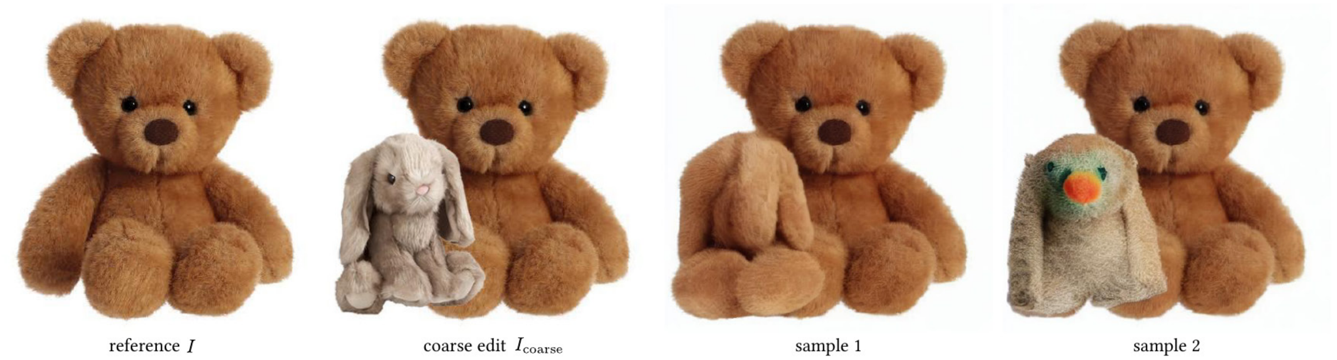


Fig. 15. **Limitations**. Since the model can was only trained on spatial edits by rearranging the parts of a single image, the model struggles in inserting objects from outside the original image. Here we see that the model attempt to stylize the bunny to have an appearance similar to the teddy bear, but struggles to preserve the bunny's identity. Photos sourced from ©Unsplash.

5 Limitations and conclusions

We present a method of assisting artists in photo editing through generative models while retaining a large level of control that traditional editing pipelines provide. We observe that with the appropriate motion model, we can use videos to train a model that can serve as a direct plugin in the editing process. We hope that our work inspires future editing research that can simply remove the cumbersome last-mile work by the press of a button.

Our generative model is trained for spatial compositions using video data. It can spatially re-compose parts of the image but would struggle to insert objects from a completely different image as shown in Figure. 15. Furthermore, we inherit the limitations of Latent Diffusion Models, which we use as our base models, especially for generating hands, faces, and small objects.

Acknowledgement. We would like to thank Sachin Shah, Nick Kolkin, Paul Guerro, Bryan Russell, and Tianwei Yin for fruitful discussions and feedback in revising the paper.

References

Rameen Abdal, Yipeng Qin, and Peter Wonka. 2019. Image2stylegan: How to embed images into the stylegan latent space?. In *Proceedings of the IEEE/CVF international conference on computer vision.* 4432–4441.

Rameen Abdal, Yipeng Qin, and Peter Wonka. 2020. Image2stylegan++: How to edit the embedded images?. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 8296–8305.

Alex Andonian, Sabrina Osmany, Audrey Cui, YeonHwan Park, Ali Jahanian, Antonio Torralba, and David Bau. 2021. Paint by word. *arXiv preprint arXiv:2103.10951* (2021).

Shai Avidan and Ariel Shamir. 2007. Seam carving for content-aware image resizing. In *ACM SIGGRAPH 2007 Papers* (San Diego, California) *(SIGGRAPH '07)*. Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/1275808.1276390

Omri Avrahami, Dani Lischinski, and Ohad Fried. 2022. Blended diffusion for text-driven editing of natural images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 18208–18218.

Max Bain, Arsha Nagrani, Gül Varol, and Andrew Zisserman. 2021. Frozen in Time: A Joint Video and Image Encoder for End-to-End Retrieval. In *IEEE International Conference on Computer Vision*.

Arpit Bansal, Hong-Min Chu, Avi Schwarzschild, Soumyadip Sengupta, Micah Goldblum, Jonas Geiping, and Tom Goldstein. 2023. Universal Guidance for Diffusion Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. 843–852.

Connelly Barnes, Eli Shechtman, Adam Finkelstein, and Dan B Goldman. 2009. PatchMatch: A randomized correspondence algorithm for structural image editing. *ACM Trans. Graph.* 28, 3 (2009), 24.

Marcelo Bertalmío, A. Bertozzi, and Guillermo Sapiro. 2001. Navier-stokes, fluid dynamics, and image and video inpainting. *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001* 1 (2001), I–I. https://api.semanticscholar.org/CorpusID:695955

Tim Brooks, Aleksander Holynski, and Alexei A Efros. 2023. Instructpix2pix: Learning to follow image editing instructions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 18392–18402.

Mingdeng Cao, Xintao Wang, Zhongang Qi, Ying Shan, Xiaohu Qie, and Yinqiang Zheng. 2023. MasaCtrl: Tuning-Free Mutual Self-Attention Control for Consistent Image Synthesis and Editing. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 22560–22570.

Lucy Chai, Jonas Wulff, and Phillip Isola. 2021. Using latent space regression to analyze and leverage compositionality in {GAN}s. In *International Conference on Learning Representations*. https://openreview.net/forum?id=sjuuTm4vj0

Xi Chen, Lianghua Huang, Yu Liu, Yujun Shen, Deli Zhao, and Hengshuang Zhao. 2024. AnyDoor: Zero-shot Object-level Image Customization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 6593–6602.

Taeg Sang Cho, Moshe Butman, Shai Avidan, and William T Freeman. 2008. The patch transform and its applications to image editing. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 1–8.

Guillaume Couairon, Jakob Verbeek, Holger Schwenk, and Matthieu Cord. 2023. DiffEdit: Diffusion-based semantic image editing with mask guidance. In *The Eleventh International Conference on Learning Representations*. https://openreview.net/forum?id=3lge0p5o-M-

Katherine Crowson, Stella Biderman, Daniel Kornis, Dashiell Stander, Eric Hallahan, Louis Castricato, and Edward Raff. 2022. Vqgan-clip: Open domain image generation and editing with natural language guidance. In *European Conference on Computer Vision*. Springer, 88–105.

Prafulla Dhariwal and Alexander Nichol. 2021. Diffusion Models Beat GANs on Image Synthesis. In *Advances in Neural Information Processing Systems*, Vol. 34. 8780–8794.

Dave Epstein, Allan Jabri, Ben Poole, Alexei Efros, and Aleksander Holynski. 2023. Diffusion Self-Guidance for Controllable Image Generation. In *Advances in Neural Information Processing Systems*, Vol. 36. Curran Associates, Inc., 16222–16239.

Patrick Esser, Robin Rombach, and Bjorn Ommer. 2021. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 12873–12883.

Rinon Gal, Or Patashnik, Haggai Maron, Amit H Bermano, Gal Chechik, and Daniel Cohen-Or. 2022. StyleGAN-NADA: CLIP-guided domain adaptation of image generators. *ACM Transactions on Graphics (TOG)* 41, 4 (2022), 1–13.

Daniel Geng and Andrew Owens. 2024. Motion Guidance: Diffusion-Based Image Editing with Differentiable Motion Estimators. In *International Conference on Representation Learning*, Vol. 2024. 17451–17472. https://proceedings.iclr.cc/paper_files/paper/2024/file/4b1d9a1fbf7b2a93bea08e18792fe436-Paper-Conference.pdf

Nicholas Guttenberg. 2023. Diffusion with Offset Noise. Retrieved January 22, 2024 from https://www.crosslabs.org/blog/diffusion-with-offset-noise

Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. 2023. Prompt-to-Prompt Image Editing with Cross-Attention Control. In *Proceedings of the International Conference on Learning Representations (ICLR)*. https://openreview.net/forum?id=_CDixzkzeyb

Amir Hertz, Andrey Voynov, Shlomi Fruchter, and Daniel Cohen-Or. 2024. Style Aligned Image Generation via Shared Attention. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 4775–4785.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising Diffusion Probabilistic Models. In *Advances in Neural Information Processing Systems*, Vol. 33. 6840–6851.

Li Hu. 2024. Animate anyone: Consistent and controllable image-to-video synthesis for character animation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 8153–8163.

Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. 2017. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1125–1134.

Jiaya Jia, Jian Sun, Chi-Keung Tang, and Heung-Yeung Shum. 2006. Drag-and-drop pasting. *ACM Trans. Graph.* 25, 3 (jul 2006), 631–637. https://doi.org/10.1145/1141911.1141934

Gwanghyun Kim, Taesung Kwon, and Jong Chul Ye. 2022. Diffusionclip: Text-guided diffusion models for robust image manipulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2426–2435.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *CoRR* abs/1412.6980 (2014). https://api.semanticscholar.org/CorpusID:6628106

Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. 2023. Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.

Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. 2022. BLIP: Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation. In *ICML*.

Shanchuan Lin, Bingchen Liu, Jiashi Li, and Xiao Yang. 2024. Common diffusion noise schedules and sample steps are flawed. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 5404–5411.

Sean J. Liu, Maneesh Agrawala, Stephen DiVerdi, and Aaron Hertzmann. 2022. ZoomShop: Depth-Aware Editing of Photographic Composition. *Computer Graphics Forum* 41, 2 (2022), 57–70. https://doi.org/10.1111/cgf.14458

Grace Luo, Trevor Darrell, Oliver Wang, Dan B Goldman, and Aleksander Holynski. 2024. Readout Guidance: Learning Control from Diffusion Features. In *CVPR*.

Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. 2022. SDEdit: Guided Image Synthesis and Editing with Stochastic Differential Equations. In *International Conference on Learning Representations*.

Ron Mokady, Amir Hertz, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. 2023. Null-text inversion for editing real images using guided diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 6038–6047.

Mathew Monfort, Alex Andonian, Bolei Zhou, Kandan Ramakrishnan, Sarah Adel Bargal, Tom Yan, Lisa Brown, Quanfu Fan, Dan Gutfruend, Carl Vondrick, et al. 2020. Moments in Time Dataset: One Million Videos for Event Understanding. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 42, 2 (2020), 502–517. https://doi.org/10.1109/TPAMI.2019.2901464

Chong Mou, Xintao Wang, Jiechong Song, Ying Shan, and Jian Zhang. 2024. DragonDiffusion: Enabling Drag-style Manipulation on Diffusion Models. In *The Twelfth International Conference on Learning Representations*. https://openreview.net/forum?id=OEL4FJMg1b

Simon Niklaus and Feng Liu. 2020. Softmax Splatting for Video Frame Interpolation. In *IEEE Conference on Computer Vision and Pattern Recognition*.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernández, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Gabriel Synnaeve, Ishan Misra, Mike Rabbat, Armand Joulin, and Piotr Bojanowski. 2024. DINOv2: Learning Robust Visual Features without Supervision. *Transactions on Machine Learning Research* 11, 1 (2024), 1–28. https://doi.org/10.48550/arXiv.2304.07193

Xingang Pan, Ayush Tewari, Thomas Leimkühler, Lingjie Liu, Abhimitra Meka, and Christian Theobalt. 2023. Drag Your GAN: Interactive Point-Based Manipulation on the Generative Image Manifold. In *ACM SIGGRAPH 2023 Conference Proceedings* (Los Angeles, CA, USA) *(SIGGRAPH '23)*. Association for Computing Machinery, New York, NY, USA, Article 78, 11 pages. https://doi.org/10.1145/3588432.3591500

Karran Pandey, Paul Guerrero, Matheus Gadelha, Yannick Hold-Geoffroy, Karan Singh, and Niloy J. Mitra. 2024. Diffusion Handles Enabling 3D Edits for Diffusion Models by Lifting Activations to 3D. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 7695–7704.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*. PMLR, 8748–8763.

René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. 2021. Vision Transformers for Dense Prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 12179–12188.

René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun. 2020. Towards Robust Monocular Depth Estimation: Mixing Datasets for Zero-Shot Cross-Dataset Transfer. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 44, 3 (2020), 1623–1637.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-Resolution Image Synthesis With Latent Diffusion Models. In *Proceedings* *of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).* 10684–10695.

Michael Rubinstein, Ariel Shamir, and Shai Avidan. 2008. Improved seam carving for video retargeting. *ACM transactions on graphics (TOG)* 27, 3 (2008), 1–9.

Chitwan Saharia, William Chan, Huiwen Chang, Chris Lee, Jonathan Ho, Tim Salimans, David Fleet, and Mohammad Norouzi. 2022. Palette: Image-to-image diffusion models. In *ACM SIGGRAPH 2022 Conference Proceedings.* 1–10.

Vishnu Sarukkai, Linden Li, Arden Ma, Christopher Ré, and Kayvon Fatahalian. 2024. Collage diffusion. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision.* 4208–4217.

Yujun Shi, Chuhui Xue, Jun Hao Liew, Jiachun Pan, Hanshu Yan, Wenqing Zhang, Vincent Y. F. Tan, and Song Bai. 2024. DragDiffusion: Harnessing Diffusion Models for Interactive Point-based Image Editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).* 8839–8849.

Denis Simakov, Yaron Caspi, Eli Shechtman, and Michal Irani. 2008. Summarizing visual data using bidirectional similarity. In *2008 IEEE conference on computer vision and pattern recognition.* IEEE, 1–8.

Jiaming Song, Chenlin Meng, and Stefano Ermon. 2021. Denoising Diffusion Implicit Models. In *International Conference on Learning Representations.* https://openreview.net/forum?id=St1giarCHLP

Yizhi Song, Zhifei Zhang, Zhe Lin, Scott Cohen, Brian Price, Jianming Zhang, Soo Ye Kim, and Daniel Aliaga. 2023. ObjectStitch: Object Compositing With Diffusion Model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.* 18310–18319.

Kalyan Sunkavalli, Micah K. Johnson, Wojciech Matusik, and Hanspeter Pfister. 2010. Multi-scale image harmonization. *ACM Trans. Graph.* 29, 4, Article 125 (jul 2010), 10 pages. https://doi.org/10.1145/1778765.1778862

Zachary Teed and Jia Deng. 2020. Raft: Recurrent all-pairs field transforms for optical flow. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16.* Springer, 402–419.

Richard Tucker and Noah Snavely. 2020. Single-view View Synthesis with Multiplane Images. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*

Su Wang, Chitwan Saharia, Ceslee Montgomery, Jordi Pont-Tuset, Shai Noy, Stefano Pellegrini, Yasumasa Onoe, Sarah Laszlo, David J Fleet, Radu Soricut, et al. 2023. Imagen editor and editbench: Advancing and evaluating text-guided image inpainting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.* 18359–18369.

Yu-Shuen Wang, Chiew-Lan Tai, Olga Sorkine, and Tong-Yee Lee. 2008. Optimized scale-and-stretch for image resizing. In *ACM SIGGRAPH Asia 2008 papers.* Association for Computing Machinery, 1–8.

Shaoan Xie, Zhifei Zhang, Zhe Lin, Tobias Hinz, and Kun Zhang. 2023. Smartbrush: Text and shape guided object inpainting with diffusion model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.* 22428–22437.

Zhongcong Xu, Jianfeng Zhang, Jun Hao Liew, Hanshu Yan, Jia-Wei Liu, Chenxu Zhang, Jiashi Feng, and Mike Zheng Shou. 2024. MagicAnimate: Temporally Consistent Human Image Animation using Diffusion Model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).* 1481–1490.

Binxin Yang, Shuyang Gu, Bo Zhang, Ting Zhang, Xuejin Chen, Xiaoyan Sun, Dong Chen, and Fang Wen. 2023. Paint by example: Exemplar-based image editing with diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.* 18381–18391.

Jiraphon Yenphraphai, Xichen Pan, Sainan Liu, Daniele Panozzo, and Saining Xie. 2024. Image Sculpting: Precise Object Editing with 3D Geometry Control. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.*

Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. 2023a. Adding Conditional Control to Text-to-Image Diffusion Models.

Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. 2023b. Adding Conditional Control to Text-to-Image Diffusion Models.

Jun-Yan Zhu, Philipp Krähenbühl, Eli Shechtman, and Alexei A. Efros. 2016. Generative Visual Manipulation on the Natural Image Manifold. In *European Conference on Computer Vision.* https://api.semanticscholar.org/CorpusID:14924561

A Appendix: Additional applications

We trained MagicFixup on inputs edited using affine transforms and flow warping. In this section, we explore how the model works under different types of edits, to better understand how the model function and cleans up different types of coarse edits. We find that the model is surprisingly robust and consistently produces photorealistic outputs, even in extreme edits like colorization that were completely different from our training data.

Colorization Out model was only trained to enable spatial edits. So, we do not expect it to work well on significantly different image editing tasks like changing an object's color: our model's inputs only provide spatial transform information. We tried to see what happens nonetheless. In this task, we coarsely edited an object's color, before passing this coarse edit to our model. We show the results in Fig. 18 To our surprise, we found that our model can generate reasonable re-colorings!

Perspective editing The task of perspective editing involves warping the image to simulate capturing different parts of image with different focal lengths. Previously, Zoomshop [Liu et al. 2022] achieves perspective editing by estimating the depth map of the image, unprojecting, then reprojecting different depth ranges. The warping operation creates holes, which are then inpainted using an off the shelf method. However, a critical limitation of Zoomshop is that it can take as long as 4 hours of manual editing to achieve a clean edit, as the authors state. This is because the method requires a perfect depth map that is carefully edited by the user, as well as manually cleaning up the inpainting. However, MagicFixup excels at cleaning up coarse edits, so we implemented our own perspective editing pipeline to test our model. This pipeline lets the user unproject the scene into a set of Multi-Plane-Images [Tucker and Snavely 2020], and reprojects each plane using the user's desired field of view. In Fig. 19 we show the results of MagicFixup for perspective editing. We attempted to visually match the edits shown in ZoomShop, and include their results as a reference only (we do not have access to the original intermediate edits from ZoomShop). We find that MagicFixup introduces a super-resolution effect when enlarging distant background objects like the hill in the background, and also seamlessly cleans up the holes and seams created from the reprojection.

More complex deformations. To allow for more complex spatial edits than simple affine transformations, we experimented with Adobe Photoshop's Puppetwarp tool to create more complex deformations. In Fig. 21 we show results of a bonsai tree reshaped into a dancing-like figure. We also applied Puppetwarp to edit a portrait to add a very coarse smile, and our model significantly improves the quality of the edit, even adding subtle face wrinkles associated with the smile, on the cheeks and around the eyes.

3D transformations. To apply 3D transformations, DiffusionHandles [Pandey et al. 2024] proposes to use depth estimation to unproject the image, and applying 3D transformation on the object of interest followed by reprojection. In DiffusionHandles, the authors use the transformed depth map as the input to a depth conditioned ControlNet [Zhang et al. 2023b]. We can directly use the (coarse) reprojected RGB as input to our model to enable similar 3D reprojections. In Fig. 20 we show 3D editing results using MagicFixup. We show we can handle 3D transformations similarly to DiffusionHandles, and we outperform DiffusionHandles in preserving the identity of the image content. In the first example, we see that DiffusionHandles alters the wall on the right, and in the second example it changes the number of cars parked in the background. In the last row, the reflections on the mug in the background are altered, and the shading of the plate the mug was placed on became unnatural. On the other hand, MagicFixup completely preserves object identity.

B Appendix: Reproducibility

To ensure the reproducibility of our results, we plan to release our code along with a version of the model trained on a public video dataset. We use the public Moments in Time dataset (MiT) [Monfort et al. 2020] for our open-source model, due to the similarity of the types of videos in our dataset, as our dataset contains stock-like clips similar to the ones in Moments in Time [Monfort et al. 2020] and WebVid10M [Bain et al. 2021]. We use 700k pairs of frames from MiT in contrast to 2.4M in the main model to train the model. We avoid using the larger WebVid10M dataset as it was recently taken down and the legality of using it is unclear. In Figure 22 we show that the open source model can achieve similar effects of addressing secondary artifacts like shadows and reflections.

C Appendix: Samples of our training data

In Fig. 24 we show samples processed from our internal dataset that we used to train the model (show ref frame, target frame, flow warped, affine warped frames). Our videos come from stock-like internal dataset that is free of watermarks, unlike the commonly used public video datasets like WebVid10M [Bain et al. 2021].

D Appendix: Additional comparisons with text based methods

While text based editing methods lack the spatial control required to recompose photos precisely, we include additional qualitative comparisons for a comprehensive evaluation in Figure 23. We compare against InstructPix2Pix [Brooks et al. 2023] and MasaCtrl [Cao et al. 2023], and include our main baseline, SDEdit [Meng et al. 2022] for reference. To preserve input identity, MasaCtrl relies on a DDIM inversion step to reconstruct the input. However, inversion is not always reliable and can result in images that are similar in the high level appearance but with a different content from the input, as we show in the fox example. In contrast, we pass the input in a feed forward manner that allows the network to reliably preserve the input identity. On the other hand, InstructPix2Pix either leaves the input image intact with minimal changes, or severely alter the image identity, making it unreliable for spatial editing. We find that SDEdit is the most reliable baseline as it can take the user edit directly as an input, improving the spatial controllability. As a result, we use SDEdit as our primary baseline in the main paper.

E Appendix: Ablation on models inputs

For a comprehensive ablation study of the inputs to the detail extractor and synthesis UNets. We analyze the role of the mask to the different UNets, and we experiment with dropping the timestep embedding from the detail extractor UNet. Dropping the time embedding in the detail extractor is equivalent to doing a one-time feature extraction, which would make it similar in style to using CLIP or DINO features rather than doing a feature extraction that depends on the current denoising step. We finetune our main model for each of these configurations on images generated using WebVid-10M as we no longer have access to the original internal data. For fairness, we also finetune the main model on the same dataset. We show the results in Tab. 3. Overall, timestep embedding in the reference UNet is essential. Providing the mask to the reference UNet is not needed, but the performance difference is within the standard error.

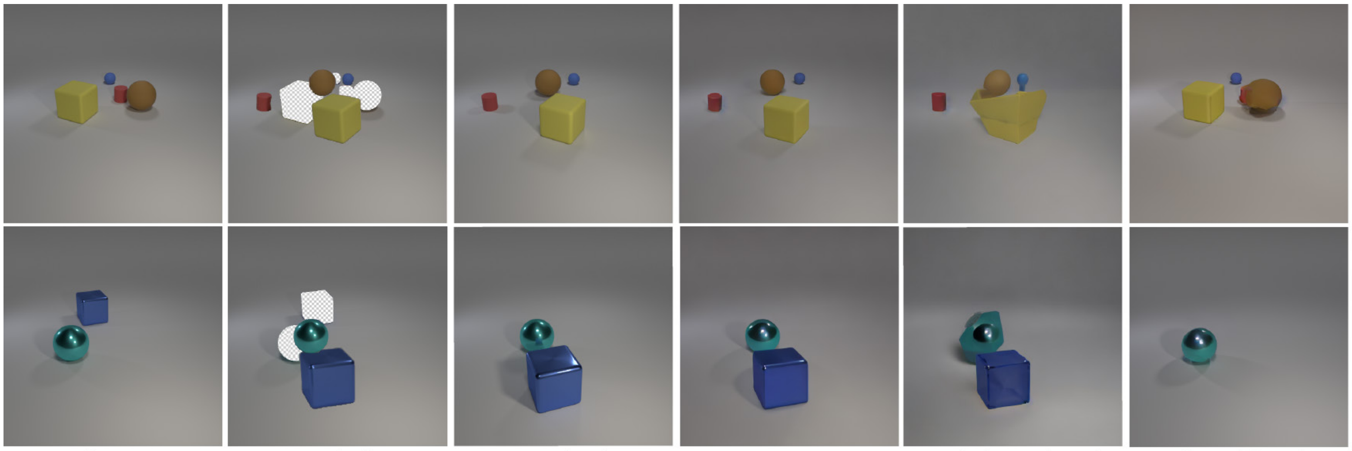
We find that providing the timestep embedding is essential for performance, which indicates that the detail extractor network extracts different features from the reference throughout the denoising process. This supports the intuitive understanding that the image generation process requires different levels of details for each step, as diffusion model generally starts by generating the coarse structure of the output and then synthesizing the higher frequency details later on. For the disocclusion mask, we find that the providing the mask is essential to the synthesis network, but provides no additional information to the detail extractor network. We also ablate including the noisy reference in the detail extractor UNet, and find that its effect on performance is negligible as expected.

F Appendix: Quantiative evaluation using CLEVR

We used evaluation dataset generated through our dataset construction pipeline, as it provides a natural test set for spatial editing. It is challenging to construct large scale evaluation datasets without developing a novel motion model that we avoid training on. However, to substantiate our results further, we rely on the CLEVR dataset. CLEVR places objects with varying materials on a surface board, and includes multiple lights that showcase interesting shadows and shading. To levarage CLEVR to evaluate our model, we generate 50 random collection of objects, and rearrange each collection 3 times. Then, for an image in a given collection, we warp it to match the two other arrangements. This way we can automatically construct coarse edits, and have access to ground truth data for quantitative evaluation at the same time. In total, the test dataset consists of 300 edits. In Fig. 16, we show two samples and the outputs of MagicFixup against SDEdit and DragDiffusion. We find that our method synthesizes new shadows and harmonizes the objects layering. On the other hand, while SDEdit can preserve the target arrangements, and DragDiffusion struggles to spatially relocate the objects. Since DragDiffusion cannot rearrange the objects, we restrict our quantiative evaluation in Table 4 to our method and SDEdit, and we find that our method outperforms the baseline in all metrics. While the CLEVR dataset is an imperfect test set, and out of distribution for our method, we find that the results further corroborate the robustness of our method and support our qualitative results.

Table 3. **Ablation on models inputs**. Here we ablate the inputs to the detail-extractor and synthesis UNets. We find that the providing the mask is essential to the synthesis network, but provides no additional information to the detail extractor network. We also find that providing the timestep embedding is essential for performance, which indicates that the detail extractor network extracts different features from the reference throughout the denoising process.

Motion model	Flow motion model			Piecewise affine motion model		
Method	**LPIPS ↓**	**SSIM ↑**	**Flow (px) ↓**	**LPIPS ↓**	**SSIM ↑**	**Flow (px) ↓**
w/o mask in either UNet	0.277 ± 0.01	0.615 ± 0.01	**35.554 ± 2.14**	0.294 ± 0.01	0.596 ± 0.01	**34.066 ± 2.19**
w/o mask in detail ext.	**0.187 ± 0.01**	**0.715 ± 0.01**	**35.682 ± 1.88**	**0.220 ± 0.01**	**0.667 ± 0.01**	**36.051 ± 2.09**
w/o timestep in detail ext.	0.504 ± 0.01	0.498 ± 0.01	71.769 ± 3.58	0.579 ± 0.01	0.457 ± 0.01	73.004 ± 3.49
w/o noisy input in detail ext.	**0.207 ± 0.01**	**0.699 ± 0.01**	**37.654 ± 2.03**	**0.242 ± 0.01**	**0.655 ± 0.01**	**37.126 ± 1.99**
ours	**0.194 ± 0.01**	**0.708 ± 0.01**	**35.716 ± 1.93**	**0.232 ± 0.01**	**0.657 ± 0.01**	**35.785 ± 1.93**



reference I | coarse edit I_{coarse} | ground truth | ours | SDEdit [Meng al. 2022] | DragDiffusion [Shi et al. 2023]

Fig. 16. **CLEVR rearranging.** We modify the CLEVR dataset generation pipeline to generate random sets of objects in different spatial arrangements, and synthesize a coarse edit that corresponds to re-aligning one arrangement to the other. Compared to the baselines, our method can better preserve the objects and harmonizes it with the environments. We find that Drag based methods like DragDiffusion struggle in generating motion beyond reposing, and SDEdit drastically morphs the objects.

Table 4. **CLEVR rearranging evaluation.** We generate a version of the CLEVR dataset where synthesize 50 random collections of objects, and rearrange each collection in three different ways. We evaluate the method's performance in realistically rearranging the objects against the ground truth.

Method	**LPIPS ↓**	**SSIM ↑**	**Flow (px) ↓**
SDEdit	0.156 ± 0.007	0.886 ± 0.002	26.02 ± 2.57
Ours	**0.078 ± 0.007**	**0.913 ± 0.003**	**8.03 ± 1.20**

G Iterative editing

One interesting question is how we can iteratively edit in image instead of making all the changes in one shot. In Figure 17 we iteratively edited the photo of the fox next to the water in a manner that is almost similar to stop-motion animation. In each step, we apply the edit on the model's output from the previous step. So we set the model's output as the "reference" for the new edit. We find that the model gracefully handles the first three iterative edits, and notice that the model's output. In the second edit, we see that the model auto-completes the body of the fox, and allows additional edits that are not possible with the original reference. We believe that our work paves a path for a future research direction that allows a human in the loop to interactively edit their photos alongside generative models.

H Appendix: Collage transform interface

To facilitate creating user edits quickly, we created our own interface that supports the user selecting any object or parts they would like to edit, and make the edit by apply an affine transformation, duplication, or deletion. We show our user interface with an example demonstrating its usage in Figure. 25 The interface allowed our users to create edits smoothly without any prior editing experience. Several of the edits used in this paper were created by novice users with no editing background. Beyond the simplicity of the interface, we maintain a dense correspondence map between the pixels in the original image and the edit. The correspondence maps are critical to fairly compare against the baselines that take drag handles or dense flow as an input, as we can directly use the correspondence to compute the needed input.

I Appendix: Expanded user study with SDEdit

Since SDEdit [Meng et al. 2022] is our primary baseline, we conduct an additional study only comparing our method with SDEDit, and

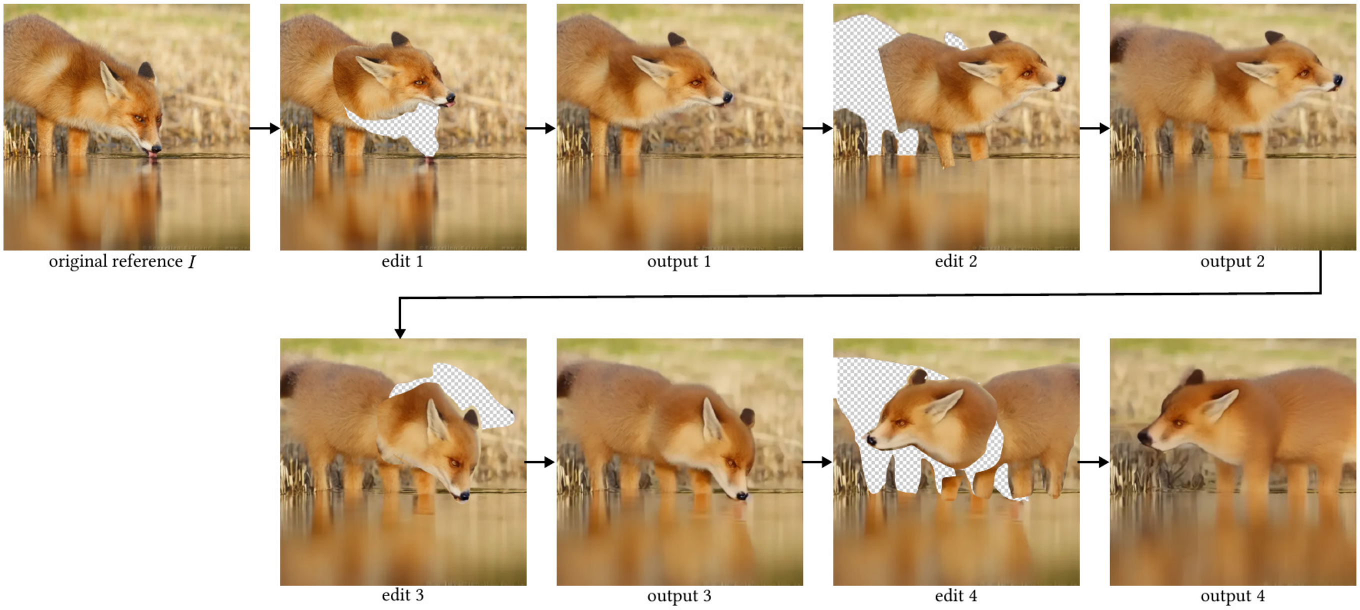


Fig. 17. **Iterative editing.** We iteratively edit the photo by setting the model's output as the new "reference" in each step, and spatially editing the model's output. Note that the model can coherently maintain the image's identity before it starts degrading with the fourth edit. We find that the iterative approach opens new possible edits. For example, in output 2 above, the model adds the rest of the fox's body, which provides context for additional edits that are not possible directly from the original image. Photo sourced from ©Roeselien Raymond.

significantly more user edits. We used 30 diverse photo edits, with 27 students participating and voting for all pairs of images. We conducted the study similar to the user study described in Section 4, in a 2-alternative forced-choice (2AFC) format. For 80% of the edits, at least 75% of the users preferred our method. For the remaining images, except for one image, users preferred our method 65 – 80% of the time. For one image in out of domain edit (editing a non-realistic artistic painting), users preferred both edits almost equally likely (52 % of users preferred SDEdit).

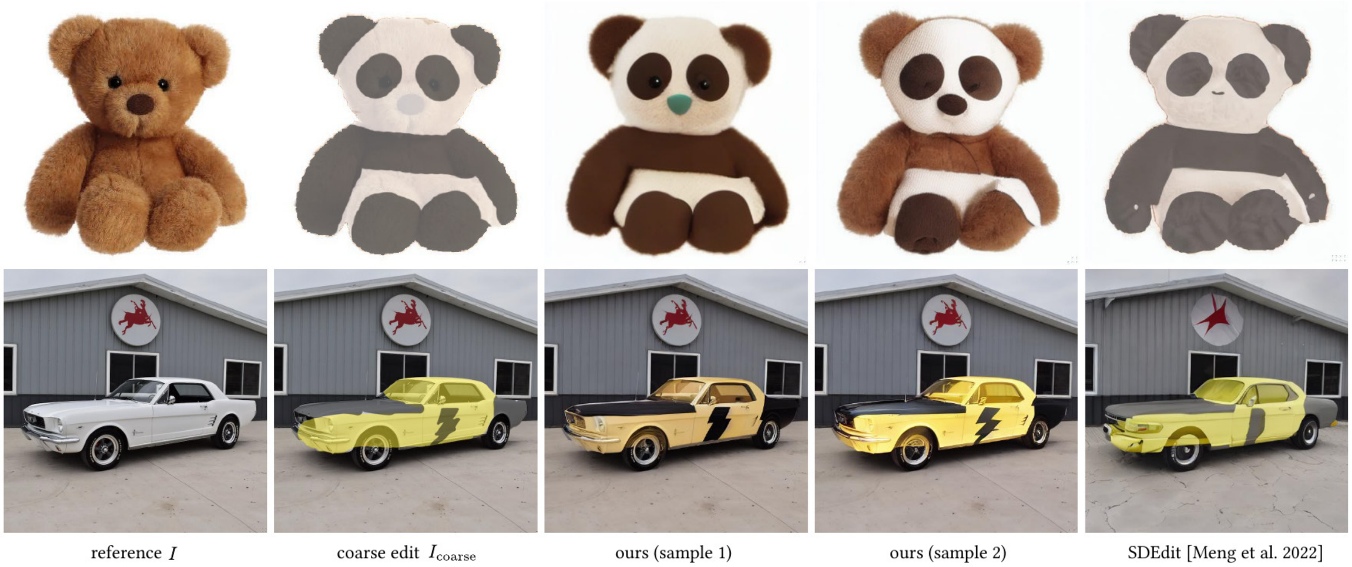


Fig. 18. **Colorization.** Even though our motion models only included spatial transformations, we experiment with running the model on out of domain coarse edits. Surprisingly, we find the model to synthesize realistic colorized outputs. The model also cleans up uneven coarse edges. For example, the lightning drawn on the Mustang contains uneven curvy edges, and the model cleans it up nicely. We also show multiple samples to highlight the diversity of the outputs the model can generate to address these edits. Photos sourced from ©Unsplash.

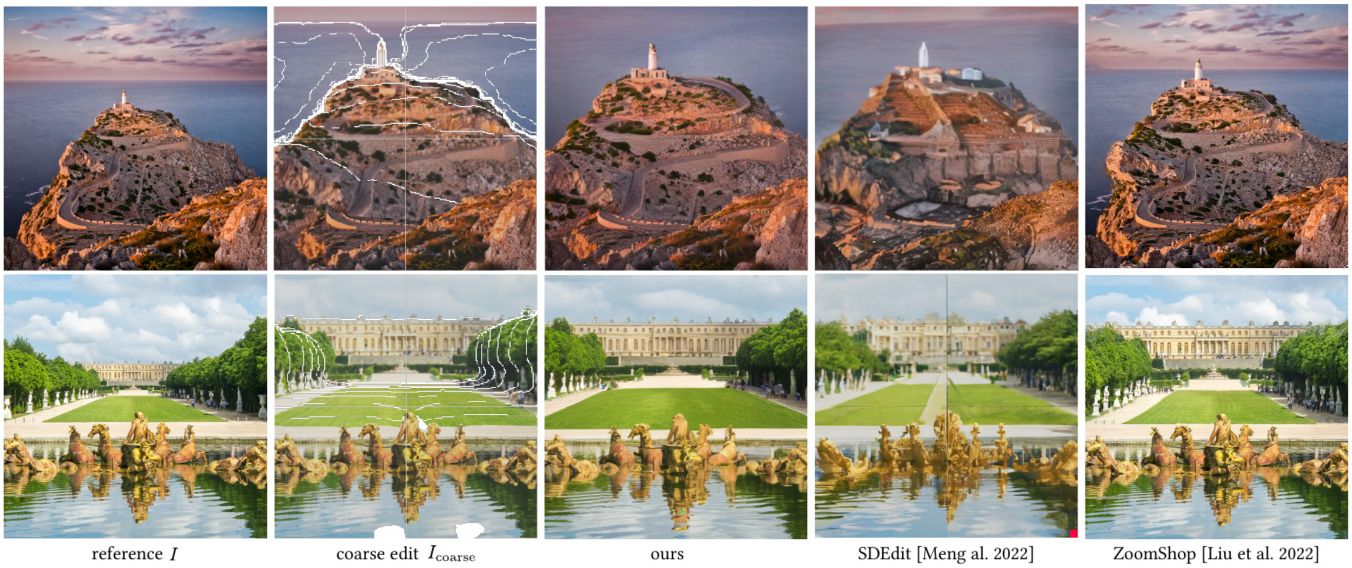


Fig. 19. **Perspective editing.** By unprojecting the scene, then reprojecting different regions using variable camera parameters, we can manipulate perspective and make distant objects appear larger. While it is time consuming to create a high quality perspective edit (ZoomShop [Liu et al. 2022] takes as long as 4 hours of manual labor), by using MagicFixup we can take a coarse reprojection and make it realistic. Here we attempt to reproduce the results from ZoomShop with our method, and include their results as a point of reference. Photos sourced from ZoomShop [Liu et al. 2022].

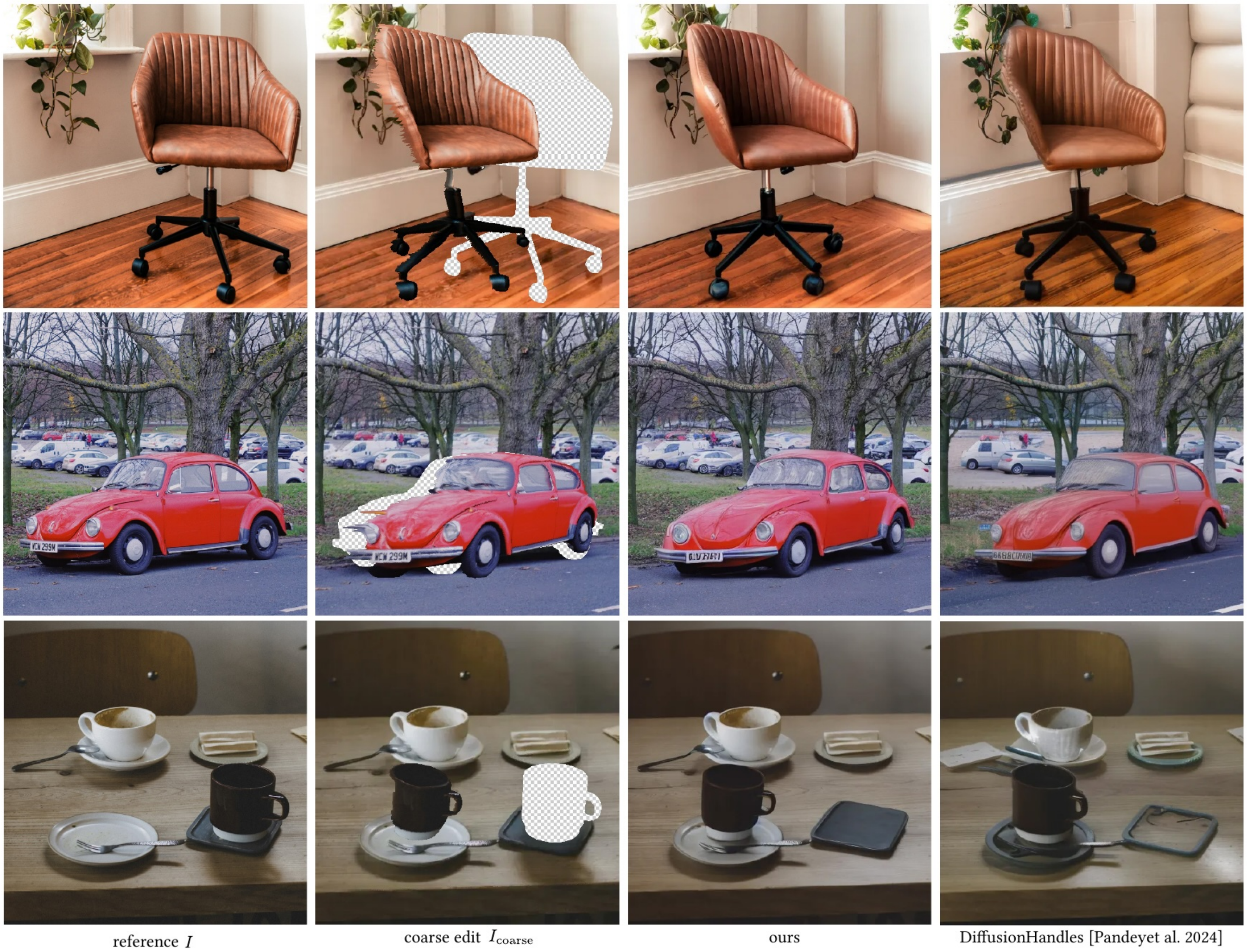

reference I
coarse edit I_{coarse}
ours
DiffusionHandles [Pandeyet al. 2024]

Fig. 20. **3D transformations.** By unprojecting the image, applying 3D transformations on the unprojected point cloud, and reprojecting, we can achieve coarse 3D edits. We show that MagicFixup can addresses the artifacts generated from the reprojection, while preserving the image identity. On the other hand, we find DiffusionHandles [Pandey et al. 2024] to alter the background identity on the right wall of the first example, the number of cars in the second example, and the shading of the plate and altering the identity of the spoon next to the white mug. Photos sourced from ©Unsplash.

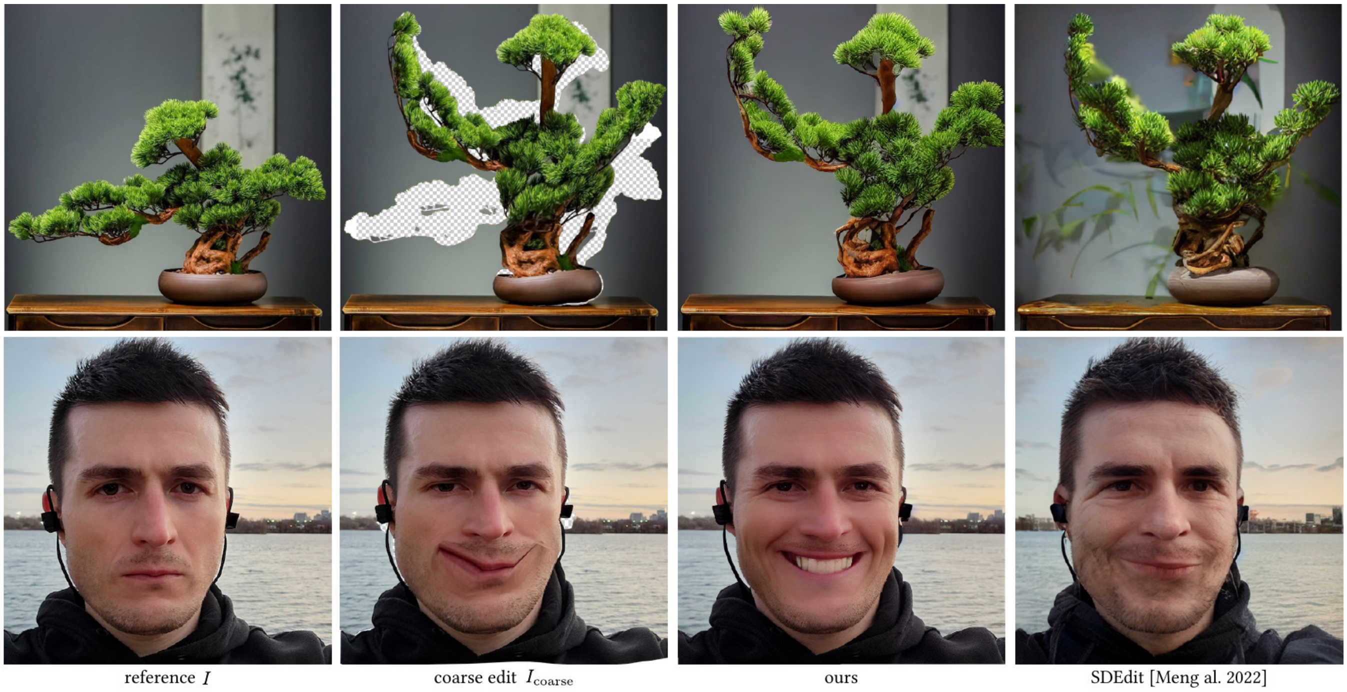


Fig. 21. **Photoshop's puppet warp.** We experiment with finer grain deformation using Photoshop's puppet warp feature. We deform the bonsai tree to resemble a dancing figure, and introduce a rough smile to a person's face. Our model then improves the realism of the edit. Note that in the second row, the model introduced a natural smile along with wrinkles around the mouth and eyes to display a more natural smile. Photos sourced from ©Unsplash and CC materials.

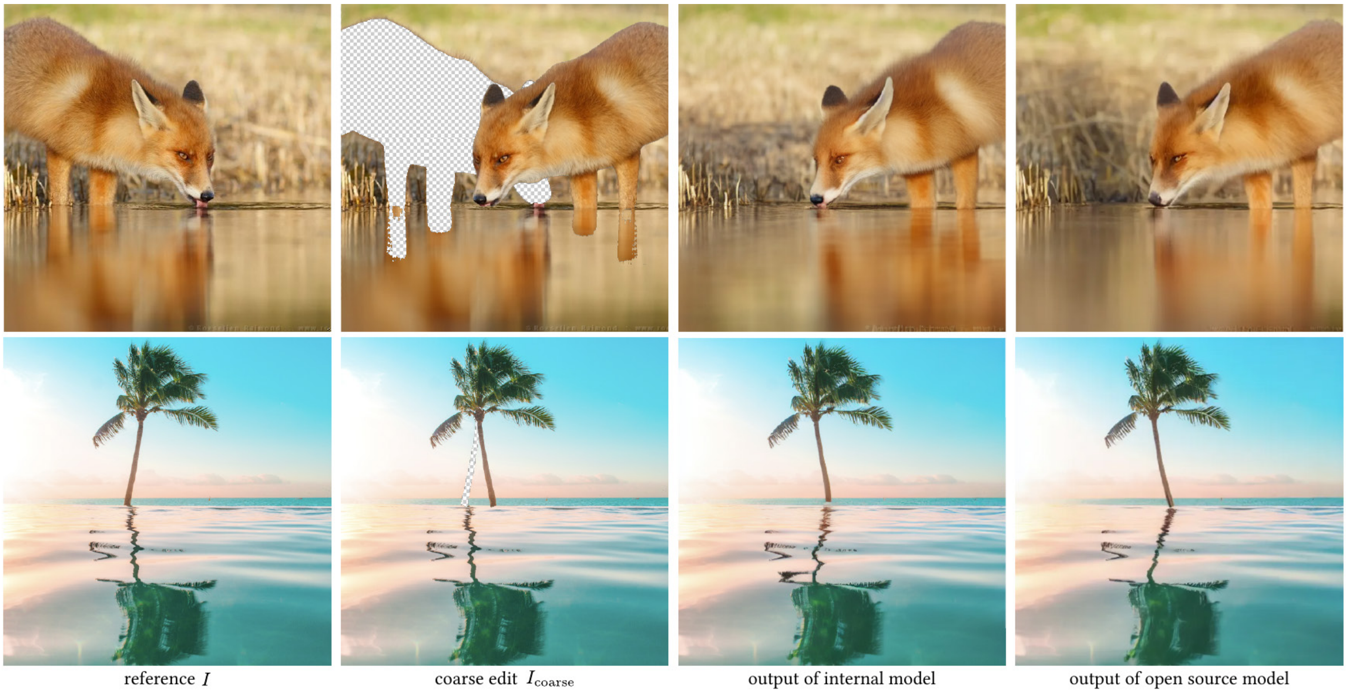


Fig. 22. **Comparison with model trained on public data.** To maximize reproducibility, we train a version of the model on public video datasets that we plan to publicly release and open source. Here we show that the model trained on public data can similarly address secondary artifacts like reflections, and clean up artifacts due to coarse selection and editing as shown in the first row example with a coarse segmentation of the fox. Photos sourced from ©Roeselien Raymond ©Unsplash.

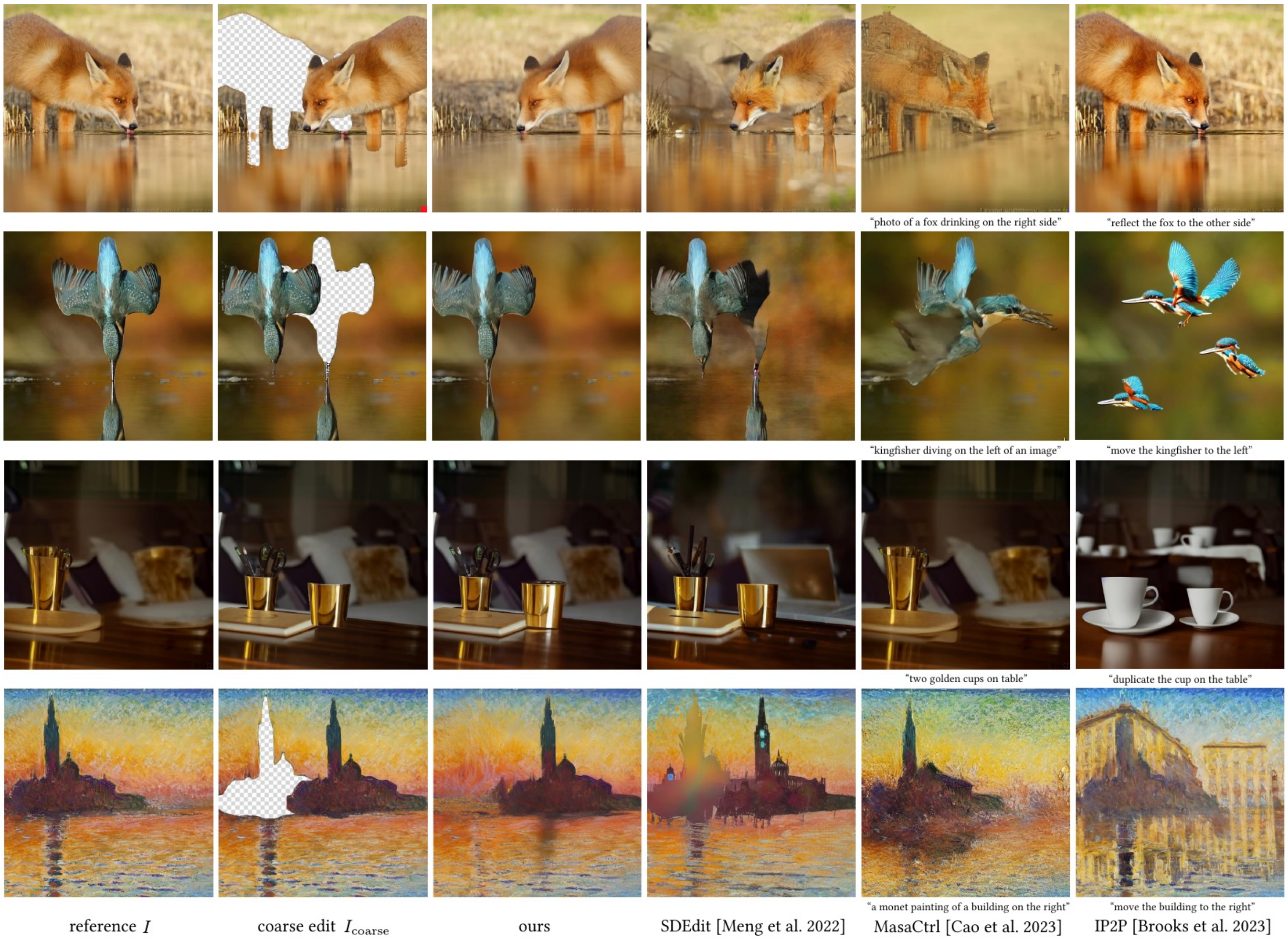

“photo of a fox drinking on the right side”
“reflect the fox to the other side”
“kingfisher diving on the left of an image”
“move the kingfisher to the left”
“two golden cups on table”
“duplicate the cup on the table”
“a monet painting of a building on the right”
“move the building to the right”
reference I
coarse edit I_{coarse}
ours
SDEdit [Meng et al. 2022]
MasaCtrl [Cao et al. 2023]
IP2P [Brooks et al. 2023]

Fig. 23. **Additional text comparisons.** We compare our method against text conditioned baselines MasaCtrl [Cao et al. 2023] and InstructPix2Pix [Brooks et al. 2023]. To preserve the input identity, MasaCtrl requires a DDIM inversion step to the input image, which is prone to failing in reconstructing the input (as we show in the first two rows, the output identity is completely different from the input due to DDIM inversion failure), and in the cases where it succeeds in DDIM inversion, it is not possible to convey the user intent through a text prompt. Similarly, InstructPix2Pix either completely changes the identity of the image, or fails into editing the image to follow the text instruction. We show the text captions we used for both baselines under the image. We show the SDEdit [Meng et al. 2022] output as a reference, and we see that it is much more effective in following the user edit than the text baselines, which is why we rely on it as our main baseline. Photos sourced from ©Roeselien Raymond, ©Unsplash, and public domain data.

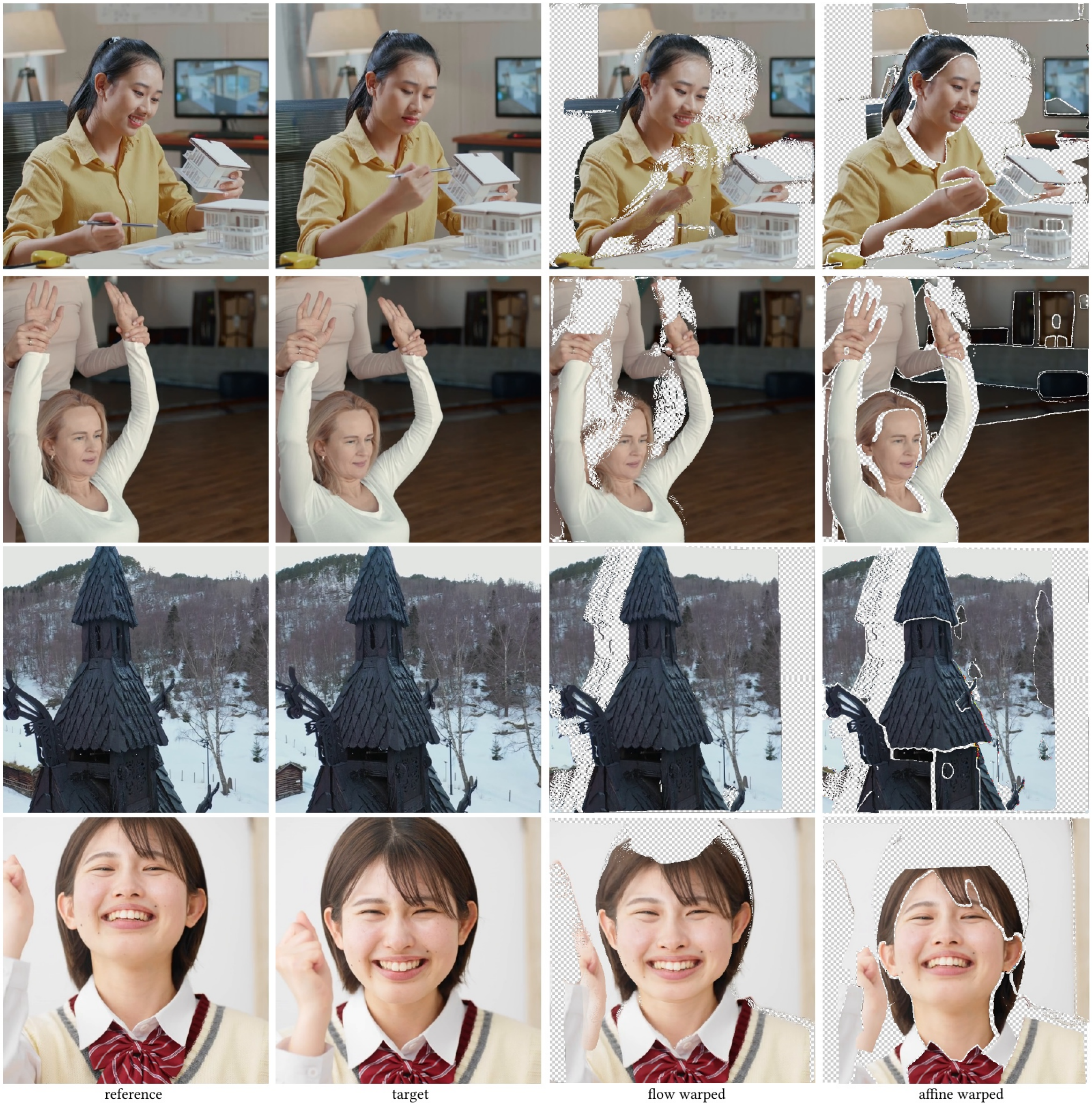
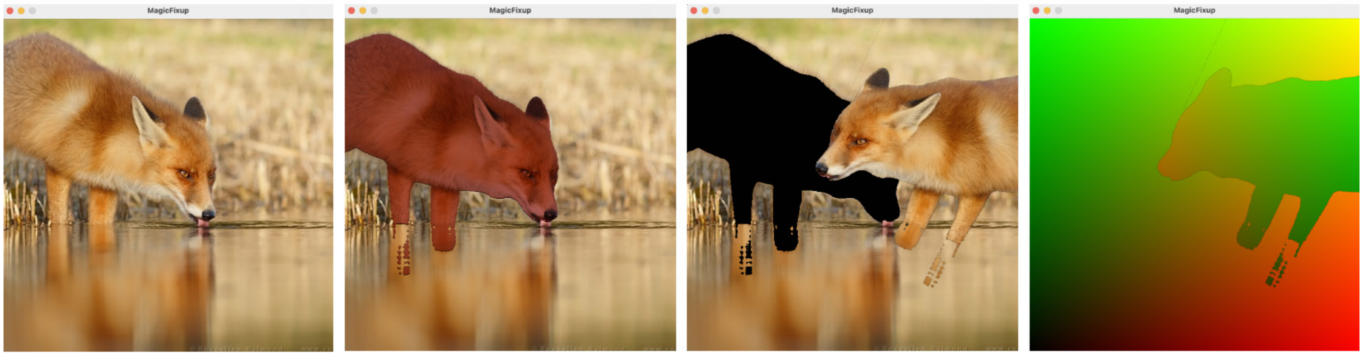


Fig. 24. **Dataset samples.** We highlight some examples from our dataset along with the outputs of the flow and affine motion models. Note that the flow model densely aligns the reference image to the target, while the affine transformations provide much coarser alignment. For example, notice in the last row that in the flow warped image, the woman's smile and facial expression is aligned with the target, while in the affine warped we only see an alignment through scaling and shifting the person's segmentation mask. Media sourced from ©Adobe Stock.



(a) input image (b) segment any part (c) edit with affine transformations (d) pixel correspondence maps

Fig. 25. **Collage transform interface.** To create user edits while maintaining correspondences between the original image and the edit, we created the Collage transform interface. The user can select any object or part they would like to edit, apply the desired affine transformation, duplication, or deletion. The correspondence map that we maintain allow us to accurately and fairly compare against the baselines by computing flow or drag keyhandles using the correspondence maps. Photo sourced from ©Roeselien Raymond.