Teamwork: Collaborative Diffusion with Low-rank Coordination and Adaptation

SAM SARTOR, College of William & Mary, USA
PIETER PEERS, College of William & Mary, USA

Large pretrained diffusion models can provide strong priors beneficial for many graphics applications. However, generative applications such as neural rendering and inverse methods such as SVBRDF estimation and intrinsic image decomposition require additional input or output channels. Current solutions for channel expansion are often application specific and these solutions can be difficult to adapt to different diffusion models or new tasks. This paper introduces Teamwork: a flexible and efficient unified solution for jointly increasing the number of input and output channels as well as adapting a pretrained diffusion model to new tasks. Teamwork achieves channel expansion without altering the pretrained diffusion model architecture by coordinating and adapting multiple instances of the base diffusion model (i.e., teammates). We employ a novel variation of Low Rank-Adaptation (LoRA) to jointly address both adaptation and coordination between the different teammates. Furthermore Teamwork supports dynamic (de)activation of teammates. We demonstrate the flexibility and efficiency of Teamwork on a variety of generative and inverse graphics tasks such as inpainting, single image SVBRDF estimation, intrinsic decomposition, neural shading, and intrinsic image synthesis.

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

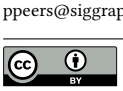
Additional Key Words and Phrases: Diffusion, Input-output Expansion, Collaborative Diffusion, LoRA, Inpainting, SVBRDF, Intrinsic Decomposition

ACM Reference Format:
Sam Sartor and Pieter Peers. 2025. Teamwork: Collaborative Diffusion with Low-rank Coordination and Adaptation. In *SIGGRAPH Asia 2025 Conference Papers (SA Conference Papers '25), December 15–18, 2025, Hong Kong, Hong Kong*. ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3757377.3763870

1 Introduction

Diffusion models [Karras et al. 2022; Rombach et al. 2022; Song et al. 2021] are a versatile class of generative models that have been applied to a wide range of image synthesis tasks such as image restoration [Dhariwal and Nichol 2021; Ho et al. 2020, 2022a], super-resolution [Kadkhodaie and Simoncelli 2021; Saharia et al. 2023], image-to-image translation [Saharia et al. 2022; Sasaki et al. 2021], and text-to-image synthesis [Nichol et al. 2022; Ramesh et al. 2022]. While flexible and powerful, diffusion models are also incredibly data hungry and computationally expensive to train. Consequently, recent work leverages large pretrained text-to-image diffusion models [Stability AI 2022, 2023a; von Platen et al. 2022] as a prior for different tasks.

Authors' Contact Information: Sam Sartor, College of William & Mary, Williamsburg, USA, slsartor@wm.edu; Pieter Peers, College of William & Mary, Williamsburg, USA, ppeers@siggraph.org.



SA Conference Papers '25, December 15–18, 2025, Hong Kong, Hong Kong

ACM ISBN 979-8-4007-2137-3/2025/12
https://doi.org/10.1145/3757377.3763870

Pretrained text-to-image diffusion models typically take, besides a text prompt, a three-channel input (i.e., noisy image) and output an three-channel RGB image, possibly via a latent space of different dimensions. Many graphics applications, however, require additional input channels (e.g., neural shading [Nalbach et al. 2017; Zeng et al. 2024]) and/or output more than three data channels (e.g., spatially-varying reflectance distribution function (SVBRDF) estimation [Sartor and Peers 2023; Vecchio et al. 2024], intrinsic image decomposition [Kocsis et al. 2024; Zeng et al. 2024], and matting [Zhang and Agrawala 2024]). Common solutions for input expansion include ControlNet [Zhang et al. 2023] and adding zero-convolutions to the head (e.g., [Sartor and Peers 2023; Zeng et al. 2024]). Expanding the number of output channels is less standardized and relies on bespoke solutions ranging from shared/joint attention [Hertz et al. 2023; Zhang and Agrawala 2024] or adding specialized tokens (i.e., prompt keywords) to switch between output tasks/channel (e.g., [Luo et al. 2024; Zeng et al. 2024]). Expanding both input and output channels requires a combination of disparate methods, which each can differ how much embedded priors in the pretrained model are preserved, how well the method mitigates overfitting, and training costs.

In this paper, we present a flexible unified solution, named *Teamwork*, for expanding the number of input and output channels when adapting a pretrained diffusion model. We formulate channel expansion as adapting multiple instances of the same foundational base model that each take care of three input or output channels. In order to model correlations between different channels, the different adapted instances of the base model require a mechanism to coordinate and exchange information. Teamwork solves both problems, adaptation and coordination, simultaneously via an elegant extension of LoRA [Hu et al. 2022] that models offsets to the linear layers over *all* base model instances jointly with an low-rank approximation. Teamwork offers a number of advantages over prior work. First, both input and output expansion is handled in the same framework. Second, Teamwork is easy to implement on top of a regular LoRA implementation. Third, Teamwork is efficient; it does not incur any computation or memory overhead compared to adapting each base-diffusion model separately with regular LoRA. Finally, Teamwork supports dynamic (de)activation of input and output channels without retraining or finetuning. Not only does this offer additional flexibility during inference, it also allows us, similar to RGB→X [Zeng et al. 2024], to train Teamwork with heterogeneous datasets that feature different subsets of data channels thereby facilitating enrichment and diversification of training exemplars.

We demonstrate the efficacy and flexibility of our Teamwork framework on a variety of inverse computer graphics tasks, such as single image SVBRDF estimation and intrinsic image decomposition, as well as generative computer graphics tasks, such as inpainting, neural rendering, and intrinsic image synthesis. Using

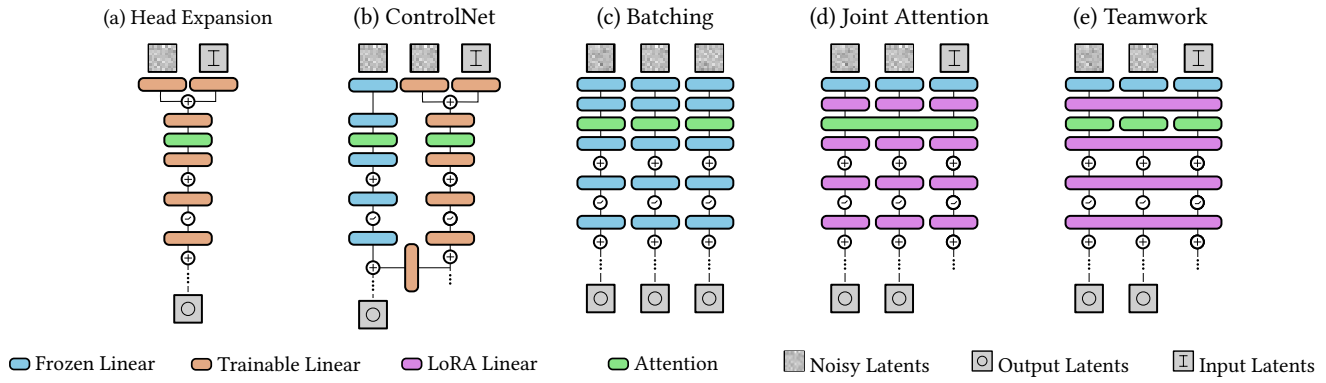


Fig. 1. Schematic overview of different common input and output channel expansion techniques for pretrained diffusion models. Input expansion: (a) Zero-convolution expands the input head with zero-initialized weights and subsequently finetunes the model. The latter operation increases the risk of overfitting, and thus destroy potentially valuable embedded priors. (b) ControlNet reduces the risk of overfitting by finetuning a copy of the (frozen) pretrained diffusion model while injecting weight offsets to each layer in the original diffusion model. Output expansion: (c) Batching, a common training optimization step, allows to run multiple instances of a model in parallel at inference. However, each model instance in a batch is unaware of the others, and thus no coordination occurs between the different instances. (d) Joint Attention coordinates between different instances of adaptations of a diffusion model by replacing self-attention layers with joint-attention layers over the multiple instances. Conceptually, Joint Attention is the dual of Teamwork (e) which keeps attention computations within each instance, but instead shares the features from the linear layers.

these applications, we validate Teamwork's performance against competing expansion and coordination techniques, demonstrate the importance of coordination, and explore the impact of dynamic (de)activation of input and output channels. Code and trained models can be found at https://github.com/samsartor/teamwork.

2 Related Work

Diffusion Models. Diffusion models form a class of powerful generative machine learning solutions that have the uncanny ability to synthesize high-quality images covering a wide variety of styles ranging from photorealistic to artistic. However, image diffusion models are also incredibly data hungry and computationally expensive to train. Fortunately, several large pretrained diffusion models are publicly available [Stability AI 2022, 2023a; von Platen et al. 2022], which have subsequently been adapted to perform novel tasks. There exists a wide variety of adaptation methods, but some of the most commonly used ones are model finetuning and Low-Rank Adaptation (LoRA) [Hu et al. 2022]. Finetuning (i.e., continuing to train the model on a new task) requires carefully designed training sets to avoid overfitting [Biderman et al. 2024; Shuttleworth et al. 2024]. LoRAs, on the other hand, mitigate overfitting by keeping the original weights $\mathbf{W}$ of linear layers, and modeling an offset $\Delta\mathbf{W}$ by a low-rank approximation which furthermore regularizes the adaptation. Formally, given the frozen linear weights $\mathbf{W} \in \mathbb{R}^{m\times n}$ of a layer, the adapted weights are then expressed as: $\mathbf{W} + \Delta\mathbf{W}$, with $\Delta\mathbf{W} = \mathbf{AB}$, $\mathbf{A} \in \mathbb{R}^{m\times r}$, and $\mathbf{B} \in \mathbb{R}^{r\times n}$, forming a low rank approximation ($r << m, n$) of the parameter offsets. While other regularization techniques (e.g., weight decay and drop-out) and adaptation methods (e.g., Elastic Weight Consolidation [Kirkpatrick et al. 2017], Synaptic Intelligence [Zenke et al. 2017], Adaptor Layers [Houlsby et al. 2019; Lin et al. 2020]) exist, LoRAs have the added advantage of requiring little storage overhead or increased latency. However, model adaptation does not alter the number of input and output channels. Teamwork keeps LoRA's compactness and efficiency, while at the same time it adds the ability to expand the number of coordinated input and output channels.

Input Expansion. A lightweight strategy to increase the number of input channels (e.g., to add additional conditions to the diffusion model) is to expand the input head with a zero-convolution and subsequently finetune the model (Figure 1.a). Zero-convolution head expansion has been used, for example, to add conditions to an unconditional SVBRDF prediction model [Sartor and Peers 2023], and to include additional input maps in a neural shading network [Zeng et al. 2024]. However, finetuning increases the risk of overfitting [Biderman et al. 2024; Shuttleworth et al. 2024]. ControlNet [Zhang et al. 2023] offers an elegant alternative that reduces the risk of overfitting by leaving the base model unchanged, and instead ControlNet adds a parallel model (with identical architecture) that provides weight offsets at each layer of the diffusion model (Figure 1.b). A variant of ControlNet is ControlLoRA [Hecong 2023] where the weights of the ControlNet are adapted with LoRAs. GLIGEN [Li et al. 2023] also keeps the base model frozen, but adds a gated self-attention layer between the self-attention and cross-attention in the transformer blocks. While suitable for input channel expansion, neither ControlNet, nor ControlLoRA, nor GLIGEN can perform output channel expansion.

Output Expansion. Expanding the number of output channels of a pretrained diffusion model without full retraining remains an open problem and task-specific solutions have been introduced. Recent work in leveraging pretrained diffusion models for intrinsic image decomposition [Luo et al. 2024; Zeng et al. 2024] (i.e., decomposing an image in different reflectance components such as albedo, roughness, normals and irradiance) introduced specialized prompt keywords to direct the diffusion model to generate the requested output. While each output is generated using the same model weights,

there is no explicit coordination between the inference processes of different outputs, and coherence is implicitly encouraged by using the same seed for each input plus a strong conditioning on the input image. An alternative strategy is to leverage multiple (differently) adapted instances of a pretrained base model that each produce part of the desired output channels. However, to ensure that all instances generate a coherent output, some form of coordination between the different models is needed – without coordination each model operates independently, cf. batching (Figure 1.c). Hertz et al. [2023] share attention between the different instances to generate multiple images with a consistent style. Recently, Zhang et al. [2024] employed joint attention to simultaneously generate a fore and background layer (Figure 1.d). While joint attention is a very flexible and powerful coordination strategy, it incurs a significant (quadratic) computational overhead cost in terms of model instances. In contrast, the coordination between different teammates in Teamwork does not incur any computational overhead compared to the cost of adapting each diffusion model instance with regular LoRA.

Multi-diffusion Inference. Multidiffusion [Bar-Tal et al. 2023] interleaves multiple diffusion models to synthesize high resolution images and panoramas. Multidiffusion does not explicitly coordinate between the different diffusion instances, but instead relies on implicit coordination by spatially overlapping each models' output and by interleaving and combining the denoising steps. Hence, it does not support output expansion, only domain extension. Furthermore, multidiffusion only supports coordination with (spatially) nearby diffusion instances. In contrast, Teamwork supports channel expansion, and it provides explicit support for coordination between all diffusion instances.

A common strategy in video-diffusion [Blattmann et al. 2023; Girdhar et al. 2024; Gupta et al. 2025; Ho et al. 2022b; Singer et al. 2023] is to run an image diffusion model per-frame and coordinate synthesis between frames by augmenting the diffusion model with a combination of 3D convolutions (or factored 2D spatial and 1D temporal convolutions) and temporal attention. However, due to the computational overhead of temporal attention, these models often synthesize a few key frames and then interpolate the in-between frames. Lumiere [Bar-Tal et al. 2024] synthesizes a whole video at once by using a cascading architecture that down/upsamples both spatially and temporally. A key difference between video diffusion and Teamwork is that video diffusion performs the same task for each frame, whereas Teamwork performs semantically different tasks per diffusion instance. Furthermore, due to the memory efficiency of the low-rank approximation, Teamwork scales better in the number of output channels.

3 Method

Our goal is to widen the number of output or input channels of a given pretained diffusion model such as Stable Diffusion [Esser et al. 2025; Stability AI 2022, 2023b], Deep Floyd [Stability AI 2023a], or Flux [Black Forest Labs 2024], as well as adapt the model for a given task. Modifying the architecture of the base diffusion model typically destroys much of the embedded knowledge. Therefore, we retain the original architecture, and instead run $T = \lceil c/3 \rceil$ adapted instances of a base diffusion model, called *teammates*, that coordinate their outputs for c output/input channels. We first explain the basis idea for widening and adapting the output channels, and extend this idea to expanding the input channels in Section 3.2.

3.1 Output Expansion

Similar as in LoRA [Hu et al. 2022], we focus on adapting the linear layers of the diffusion model. Denote $\mathbf{W} \in \mathbb{R}^{m \times n}$ the frozen weights of a linear layer in the base diffusion model, and $\mathbf{y} = [\mathbf{y}_1 \dots \mathbf{y}_T], \mathbf{y}_i \in \mathbb{R}^m$ and $\mathbf{x} = [\mathbf{x}_1 \dots \mathbf{x}_T], \mathbf{x}_i \in \mathbb{R}^n$ are the concatenated output and input feature vectors respectively of T instances of the linear layer. We can then compactly formulate the combined linear layer as:

$$\mathbf{y} = \begin{bmatrix} \mathbf{W} & & \\ & \ddots & \\ & & \mathbf{W} \end{bmatrix} \mathbf{x}, \tag{1}$$

which is equivalent to batching T instances of a diffusion model (Figure 1.c). Each instance is oblivious to the others, which is expressed by the block diagonal structure of the combined linear weight matrix in Equation (1) and thus each $\mathbf{x}_i$ only affects $\mathbf{y}_j$ when $i = j$. Consequently, the output of each instance is independently generated from the outputs of the other instances.

To adapt the joint model in Equation (1) for output channel expansion, we need (1) to adapt the behavior of each instance (i.e., different teammates should output a different subset of the output channels), and (2) ensure that the outputs of different teammates are coherent (e.g., similar to how there is coherence between R, G and B channels in an image, so is there often coherence between different channels in multi-channel data such as SVBRDFs or intrinsic images). We propose to achieve both goals jointly by addition of an offset weight matrix:

$$\mathbf{y} = \texttt{block}(\mathbf{W})\mathbf{x} + \Delta\mathbf{W}\mathbf{x}, \tag{2}$$

where $\Delta\mathbf{W} \in \mathbb{R}^{Tm \times Tn}$ are the adaptation offset weights. The matrix $\Delta\mathbf{W}$ is very large, and an efficient encoding is desired. Inspired by LoRA [Hu et al. 2022], we employ a low-rank representation:

$$\Delta\mathbf{W} = \begin{bmatrix} \mathbf{A}_1 \\ \vdots \\ \mathbf{A}_T \end{bmatrix} \begin{bmatrix} \mathbf{B}_1 & \dots & \mathbf{B}_T \end{bmatrix}. \tag{3}$$

with low rank factors $\mathbf{A}_i \in \mathbb{R}^{m \times r}$ and $\mathbf{B}_i \in \mathbb{R}^{r \times n}$ respectively.

To gain better insight in Equation (2), we contrast it to applying regular LoRA to each of the T teammates. In the latter case, the combined weight matrix becomes: $\Delta\mathbf{W} = \texttt{block}_1(\mathbf{A}_i\mathbf{B}_i)$. Although uniquely adapted, each $\mathbf{y}_i$ is again only affected by $\mathbf{x}_j$ when $i = j$. Hence, similar to batching (Figure 1.c), each teammate operates in isolation and it is oblivious to what the other teammates are doing. In contrast, our Teamwork framework forms a low-rank representation of the *whole* offset matrix $\Delta\mathbf{W}$. Crucially, because the resulting $\Delta\mathbf{W}$ is not block-diagonal anymore, each output $\mathbf{y}_i$ is now affected by all inputs $\mathbf{x}_j$, allowing for feature information to flow between different teammates (i.e., coordination).

We can also draw parallels between Teamwork and Joint Attention [Zhang and Agrawala 2024]. Joint attention enables coordination between different instances of a(n adapted) diffusion model by replacing the self-attention layers with joint-attention layers

across all models (Figure 1.d). However, Joint Attention suffers from a quadratic computation complexity with respect to the number of models (as well as the resolution). Furthermore, Joint Attention only offers coordination and a separate adaptation method is needed (e.g., regular LoRA on each instance). Teamwork is in some sense the dual of Joint Attention; instead of exchanging information via the attention layers, Teamwork instead shares information via the linear features of the model (Figure 1.e). This offers three advantages: (1) coordination and adaption are performed jointly, (2) the number of coordination layers quadruples (i.e., four linear layers per attention layer), and (3) the computational cost is identical to applying a regular LoRA to each instance (i.e., linear with the number of teammates; see Section 3.4).

3.2 Input Expansion

We can also leverage Teamwork to widen the number input channels (i.e., add condition images). Similar as with output expansion, we add a teammate for each triplet of input conditions; we differentiate between both types of teammates as *input-teammates* and *output-teammates*. The key idea is that diffusion models extract sophisticated features that are not only useful for image synthesis, but that are also meaningful for conditioning the diffusion process; these features are present irrespective of the degree of noise in the input. Thus input-teammates do not function as diffusion models, but as vision models. Therefore, we pass the noise-free input image at *every* diffusion step instead of starting from noisy latents, and propagate the resulting features. This is equivalent to ignoring the output of the input-teammates at every diffusion step. Consequently, no loss is defined on the input-teammates. However, this does not mean that the input-teammates are not adapted; the loss from the output-teammates is still propagated via the dense offset weights $\Delta\mathbf{W}$ to the input-teammates.

3.3 Dynamic Activation of Teammates

Teamwork also supports dynamic (de)activation of teammates, both during inference and training. The latter is particularly useful when training on heterogenous datasets that have different subsets of the channels available. Similar to RGB→X [Zeng et al. 2024], rather than selecting the largest common subset for training, we can during training dynamically activate the channels present for each training exemplar, thereby maximally leveraging the information in the combined training set. Practically, dynamic activation is achieved by only including the low rank factors $\mathbf{A}_i$ and $\mathbf{B}_i$ for active teammates (and corresponding $\mathbf{x}_i$) when evaluating $\Delta\mathbf{W}\mathbf{x}$ (Equation (3)).

3.4 Practical Considerations

Materialization. Materializing $\mathbf{W} + \Delta\mathbf{W}$ in regular LoRA is common practice because it slightly reduces computational evaluation costs from $O(mn + r(m + n))$ to $O(mn)$. However, for Teamwork with T teammates, we can exploit that the frozen component is a block-matrix, yielding an unmaterialized evaluation cost of $O(Tmn + Tr(m + n))$, compared to $O(T^2mn)$ for evaluation on a materialized $\mathbf{W} + \Delta\mathbf{W}$, and hence materialization is more expensive for modest Teamwork-rank. Furthermore, the memory cost of materialization for Teamwork is more significant due to the dense off-diagonal blocks.

Batch Trick. To simplify implementation of Teamwork, we exploit the inherent capability of batching to run multiple instances of a diffusion model simultaneously, and expand it by performing Teamwork-aware (low-rank) operations across the batch dimensions. As a result, linear layers perform Teamwork-aware operations across batch-dimension, while other layers broadcast across teammates without performing any coordination. However, implementing this batch trick limits the micro-batch size to 1, necessitating the use of gradient accumulation during training. Practically, we found that gradient accumulation did not pose a major disadvantage as common team sizes (5-10) can easily saturate a single GPU's memory, necessitating distributed gradient accumulation.

4 Results

Teamwork is designed as a general framework for input and output expansion of pretrained diffusion models. To demonstrate the versatility and flexibility of our framework, we apply Teamwork to five different generative and inverse rendering tasks (inpainting, single image SVBRDF estimation, intrinsic image decomposition, neural shading, and intrinsic image synthesis) that require an expansion of the input and/or output channels. All training (and timing) is performed on a single NVIDIA A40 with 48GB of VRAM. We use the Prodigy optimizer [Mishchenko and Defazio 2024] to train Teamwork using 16× gradient accumulation and using a cosine learning rate schedule with the standard loss for which the base model was trained (i.e., diffusion loss for Stable Diffusion XL [Stability AI 2023b] or flow-matching loss for Stable Diffusion 3 [Esser et al. 2025] and Flux [Black Forest Labs 2024]).

Inpainting. We employ Stable Diffusion 3 [Esser et al. 2025] as the base diffusion model and adapt it for the generative task of inpainting [Rombach et al. 2022]. We use three teammates; two input-teammates: one for the image and one for the mask (copied to the three RGB channels), and one output-teammate for the resulting inpainted image. We compare Teamwork to different input expansion techniques such as ControlLoRA [Hecong 2023], Joint Attention [Zhang and Agrawala 2024] for coordinating between separately LoRA-adapted teammates, and a variant that employs both Joint Attention and Teamwork for coordination as both methods can be employed simultaneously. All models are trained on 64k images from the PixelProse dataset [Singla et al. 2024] at 1024 pixel resolution. Training took 31h for ControlLoRA and Teamwork, and 100h for the Joint Attention and combined model due the $O(T^2)$ time-complexity (versus $O(T)$ for Teamwork). In addition, we also compare against a pretrained ControlNet-based inpainting variant of Stable Diffusion 3 [Stability AI 2024] which was trained with a larger batch size (192) and on a much larger training set of 3,840,000 images from Laoin2B [Sisap 2024] and a proprietary dataset, and thus likely required significantly more resources for training. We quantitatively compare (Table 1) the different models on a random test split of the PixelProse dataset for four different metrics: SI-FID [Shaham et al. 2019] (2nd column; lower is better), CLIPScore [Hessel et al. 2021] on the inpainted image of the reference (3rd column; higher

Table 1. Teamwork performs quantitatively similarly compared to other Stable Diffusion 3 based inpainting methods on a random test split from the PixelProse dataset [Singla et al. 2024] for four different metrics: SI-FID [Shaham et al. 2019], CLIPScore [Hessel et al. 2021] with respect to the text prompt and input image, and IQA [Wang et al. 2023].

Method	SI-FID ↓	CLIPScore (Img) ↑	CLIPScore (Txt) ↑	CLIP IQA ↑
ControlNet	13.987	**90.18**	31.53	**0.530**
ControlLoRA	17.856	86.97	31.27	0.469
Joint-Attn.	14.147	89.94	31.02	0.486
Teamwork	**13.236**	89.65	31.05	0.490
Combined	16.473	89.88	**31.59**	0.515

Fig. 2. Teamwork performs qualitative similarly to different Stable Diffusion 3 based inpainting methods.

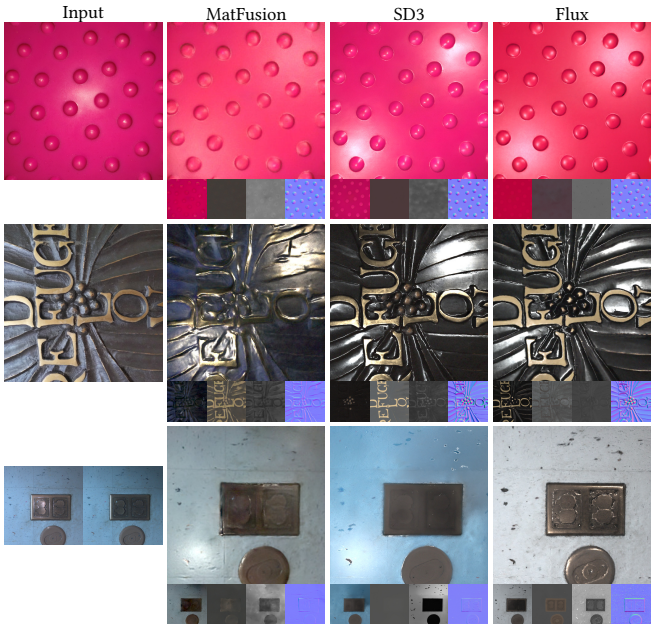


Fig. 3. Qualitative comparison of SVBRDFs estimated from real-world captured photographs of Teamwork variants and MatFusion [Sartor and Peers 2023] under colocated (1st row), environment (2nd row), and flash/no-flash (3rd row) lighting.

is better) and with respect to the input prompt (4th column; higher is better), and CLIP IQA [Wang et al. 2023] on the inpainted model (last column; higher is better). All methods perform similarly, both quantitatively (Table 1) and qualitatively (Figure 2), demonstrating that Teamwork offers a viable and competitive alternative for input channel expansion with reduced training costs.

Single Image SVBRDF Estimation. SVBRDF estimation aims to recover spatially-varying reflectance distribution (SVBRDF) parameters (e.g., diffuse albedo, specular albedo, specular roughness, and surface normals) from a single photograph of a planar material sample under controlled or uncontrolled lighting. SVBRDF estimation is traditionally formulated as an inverse rendering problem. Single image SVBRDF estimation is an inherently underconstrained problem, and generative diffusion-based SVBRDF estimation methods (MatFusion [Sartor and Peers 2023] and ControlMat [Vecchio et al. 2024]) have recently been shown to address the resulting ambiguities better than purely regressive techniques. However, to support the expanded number of output channels (10 channels in total) these prior diffusion-based SVBRDF solutions are trained from scratch.

We train three Teamwork variants for SVBRDF estimation under three different lighting conditions mirroring MatFusion [Sartor and Peers 2023]: colocated flash lighting, unknown environment lighting, and flash/no-flash image pairs. Each Teamwork model includes at minimum 3 input channels (photograph under target lighting) and 12 output channels (3 for diffuse albedo (gamma 2.2 encoded), 3 for specular albedo (gamma 2.2 encoded), 3 for normals, and 3 for roughness encoded as a gray-scale image), i.e., 4 output teammates. For the colocated model, we follow MatFusion and add a second input-teammate containing the halfway vector between the light and view vector per pixel, yielding a total of 2 input and 4 output-teammates. For the environment lighting variant, we add an extra output-teammate for the shading-only image that corresponds to (the normalized) ratio of the photograph over the sum of the diffuse and specular albedo, yielding a total of 1 input and 5 output teammates. Finally, the flash/no-flash model features 2 input-teammates (one for the flash and one for the no-flash photograph), and the same 5 output-teammates as the environment lighting variant. We train each Teamwork variant on 64k exemplars from the MatFusion SVBRDF training dataset. While MatFusion is limited to 256×256 resolution due to the computational cost of training a diffusion model from scratch, we train our models at 512×512 resolution.

We demonstrate Teamwork's flexibility by using different base diffusion models (i.e., Stable Diffusion XL [Stability AI 2023b], Stable Diffusion 3 [Esser et al. 2025], and Flux [Black Forest Labs 2024]). Qualitative comparison of the three variants against their respective MatFusion [Sartor and Peers 2023] counterparts are shown in Figure 3, Figure 4, and the supplemental material. Table 2 quantitatively compares the different methods against MatFusion, as well as selected variants that use Joint Attention for coordination instead. From the quantitative comparison, we can see that the Teamwork variants outperform MatFusion as well as the Joint Attention variants in terms of rerender error, and that all models perform competitively on accuracy for each of the reflectance components. Not only do the Teamwork variants perform better than MatFusion models despite being trained on significantly fewer training exemplars and using fewer training iterations (only 4,000 versus

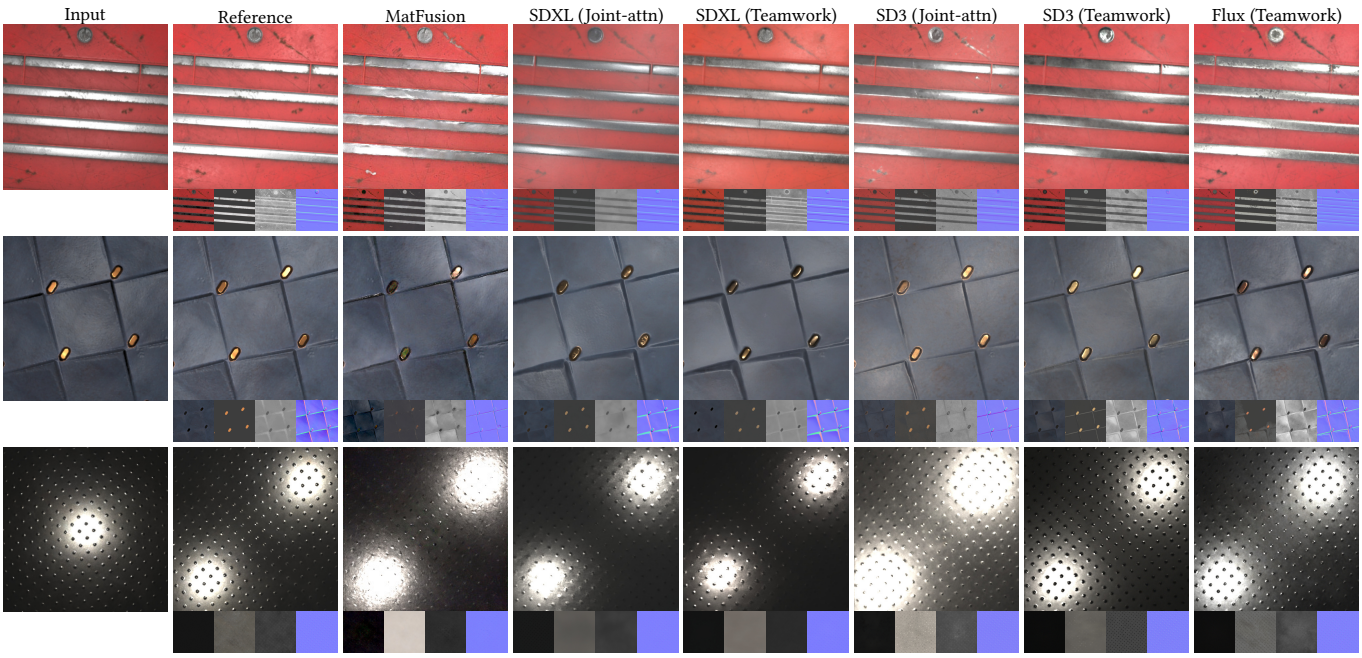


Fig. 4. Qualitative comparison of Teamwork variants and MatFusion [Sartor and Peers 2023] on simulated captures under colocated lighting for three synthetic SVBRDFs. For each SVBRDF we show a rerendering under novel lighting and the estimated diffuse albedo, specular albedo, roughness, and normal maps.

Table 2. Quantitative comparison of different Teamwork variants for single image SVBRDF estimation under colocated, environment, and flash/no-flash lighting. For reference, we compare against the diffusion-based MatFusion [Sartor and Peers 2023] models using the average RMSE on the estimated parameter maps and the average LPIPS [Zhang et al. 2018] rerender error under 128 randomly placed point lights on the MatFusion test set.

Lighting	Base Model	Coord. Method	Render Error	RMSE Diff.	RMSE Spec.	RMSE Rough.	RMSE Norm.
Coloc.	MatFus.	N.A.	0.214	0.071	0.0958	0.115	0.0562
	SDXL	Attn	0.216	**0.0471**	0.0861	**0.105**	0.0454
	SD3	Attn	0.214	0.0682	0.117	0.145	0.0481
	SDXL	Team.	0.204	0.0572	0.0876	**0.105**	**0.0416**
	SD3	Team.	**0.188**	0.0609	**0.0835**	0.12	0.0464
	Flux	Team.	0.195	0.0653	0.101	0.141	0.0454
Environ.	MatFus.	N.A.	0.337	0.197	0.133	0.258	0.087
	SD3	Team.	**0.266**	**0.143**	**0.105**	**0.183**	**0.0538**
	Flux	Team.	0.273	0.17	**0.105**	0.2	0.0551
Flash / No-flash pair	MatFus.	N.A.	0.3709	0.187	0.120	0.143	0.062
	SD3	Team.	**0.2518**	**0.150**	**0.100**	**0.116**	**0.050**
	Flux	Team.	0.3184	0.159	0.112	0.235	0.065

672,000 for MatFusion), they are also significantly faster to train: 21/18/85 GPU hours for the Stable Diffusion XL, Stable Diffusion 3, and Flux base-model respectively at 512 resolution versus ∼400 GPU hours for MatFusion (refinement time from the MatFusion base model) at 256 resolution; a 5∼20× speed-up. For reference, the Joint Attention variant required 22/41 GPU hours to train for the Stable Diffusion XL and Stable Diffusion 3 base models respectively. Note that the impact on the training cost for Stable Diffusion XL with Joint Attention versus Teamwork is less significant because Stable Diffusion XL features relatively few attention layers compared to convolutional layers. More modern diffusion architectures, such as Stable Diffusion 3 and Flux, depend almost exclusively on attention layers, and hence incur a more significant training cost increase for Joint Attention.

Intrinsic Image Decomposition. Intrinsic image decomposition aims to separate an image into various shading components, hence it constitutes a one-to-many image translation. While originally designed to decompose into just two components (i.e., reflectance and shading), more recently methods consider an expanded definition of intrinsic components [Kocsis et al. 2024; Zeng et al. 2024]; we follow this later inverse rendering view of intrinsic decomposition. We adapt Stable Diffusion 3 [Esser et al. 2025] to include 1 input-teammate (for the photograph to be decomposed) and 9 different output-teammates at 1024 resolution for: diffuse albedo, specular albedo, the summed (diffuse and specular) albedo, specular roughness, normals, depth, diffuse shading, (diffuse plus specular) shading, and the specular residual defined as the difference between the image and the diffuse albedo times the diffuse shading.

To demonstrate Teamwork's ability to support heterogeneous training datasets, we randomly select 256k training exemplars from: InteriorVerse [Zhu et al. 2022], HyperSim [Roberts et al. 2021], CGIntrinsics [Li and Snavely 2018] (using the CGIntrinsic's albedo as total reflectance), Infinigen [Raistrick et al. 2023], and renderings of random objects selected from the ABC dataset [Koch et al. 2019] textured with random MatFusion SVBRDFs [Sartor and Peers 2023] and lit by a random light probe from https://polyhaven.com/hdri; not all reflectance components are present in each dataset (see supplemental material for a summary of the available components). The resulting Teamwork model performs comparably (Table 3, Part

Table 3. Quantitative comparison of Intrinsic Image Diffusion [Kocsis et al. 2024], RGB→X [Zeng et al. 2024] (pretrained and InteriorVerse retrained variants) and a Stable Diffusion 3 based Teamwork model (a variant trained on a heterogeneous training set and a variant trained only on InteriorVerse) over the InteriorVerse test set. To compensate for the albedo-intensity ambiguity, we first apply a least-squares optimization to find an optimal scale per channel before computing the respective errors for each method.

	Method	Diffuse		Specular		Roughness		Albedo		Shading		Normal
		RMSE	LPIPS	RMSE	LPIPS	RMSE	LPIPS	RMSE	LPIPS	RMSE	LPIPS	RMSE
I	Kocsis et al.	0.135	0.205	0.126	0.242	0.232	0.384	0.110	0.166	0.144	0.230	✗
	RGB→X	0.229	0.518	0.272	0.576	0.304	0.654	0.148	0.378	0.065	0.137	0.196
	Teamwork	0.176	0.248	0.151	0.288	0.291	0.424	0.140	0.208	0.064	0.140	0.121
II	RGB→X (InteriorVerse)	0.165	0.300	0.143	0.293	0.249	0.371	0.139	0.273	0.073	0.211	0.133
	RGB→X SD3 (InteriorVerse)	0.141	0.223	0.132	0.274	0.212	0.314	0.127	0.203	0.058	0.159	0.116
	Teamwork (InteriorVerse)	**0.116**	**0.156**	**0.119**	**0.197**	**0.189**	**0.259**	**0.107**	**0.136**	**0.045**	**0.097**	**0.093**
III	No Coordination	0.205	0.353	0.169	0.370	0.329	0.508	0.170	0.334	0.083	0.225	0.171
	Sequential Teamwork	0.236	0.372	0.145	0.359	0.313	0.612	0.190	0.265	0.070	0.172	0.141
	InteriorVerse-restricted Teamwork	0.128	0.184	0.150	0.229	0.243	0.321	0.125	0.167	0.054	0.119	0.107
	Teamwork (Dropout)	0.195	0.310	0.154	0.293	0.321	0.461	0.162	0.261	0.091	0.189	0.135
	Sequential Teamwork (Dropout)	0.221	0.293	0.152	0.292	0.303	0.450	0.173	0.248	0.066	0.155	0.132

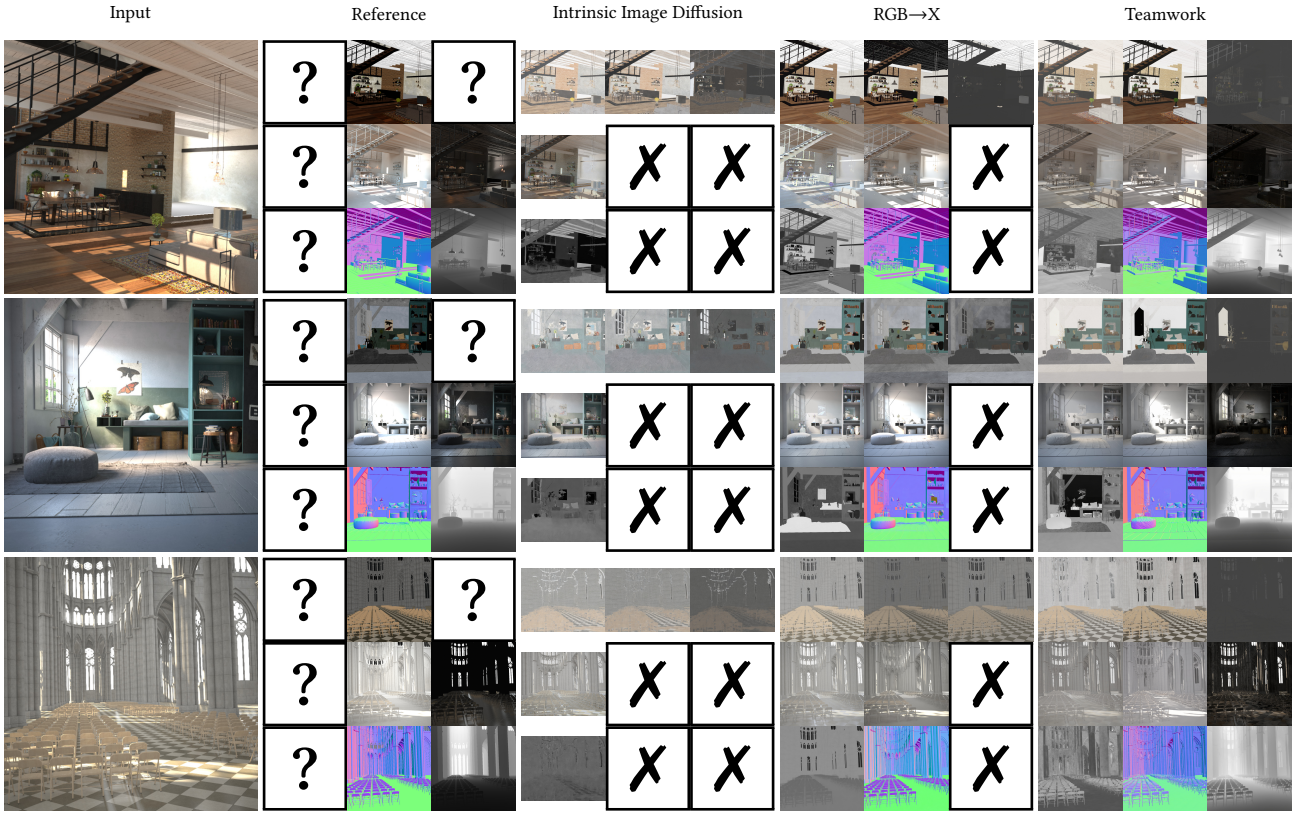


Fig. 5. Qualitative comparison (on examples from HyperSim [Roberts et al. 2021]) of the pretrained Instrinsic Image Diffusion [Kocsis et al. 2024] and pretrained RGB→X [Zeng et al. 2024] against a Stable Diffusion 3 based Teamwork variant trained on the heterogeneous training set. For each method the resulting intrinsic components (if available) are organized as: (1st row): summed albedo, diffuse albedo, and specular albedo; (2nd row): shading, diffuse shading, specular residual; and (3rd row): roughness, normals, and depth.

I) to the publicly-available Intrinsic Image Diffusion [Kocsis et al. 2024] and RGB→X [Zeng et al. 2024] models over the InteriorVerse test set. Figure 5 further qualitatively demonstrates intrinsic image decompositions on three selected images from HyperSim [Roberts et al. 2021]; see supplemental material for additional decompositions on other datasets.

While informative, the previous comparison is not conclusive, as all three models are trained on different datasets. Therefore, we re-train RGB→X and Teamwork on 256k random samples drawn from the InteriorVerse training set only (corresponding to ∼6 epochs over the full InteriorVerse training set); the pretrained Intrinsic Image Diffusion model [Kocsis et al. 2024] is already exclusively

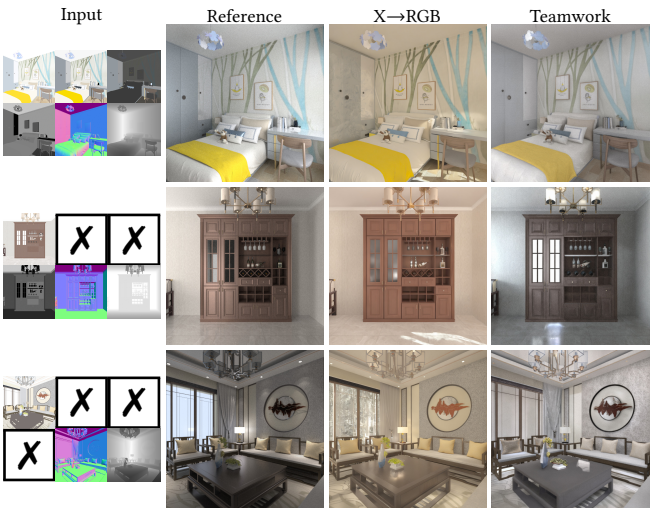


Fig. 6. Neural shading from subsets of intrinsic parameters (1st column - first row: summed albedo, diffuse albedo, specular albedo; second row: roughness, normals, depth). While an exact match to the reference (2nd column) is unlikely due to the uncontrolled lighting, Teamwork (4th column) produces qualitatively similar results to X→ RGB [Zeng et al. 2024] (3rd column).

Fig. 7. Left: results from a prompt-only conditioned intrinsic image synthesis Teamwork model trained on 128k exemplars from InteriorVerse with Gemma-3 generated prompts. Right: results from models trained without coordination (and evaluated with the same seed) lack coherence.

trained on InteriorVerse and thus not retrained. Furthermore, as RGB→X uses Stable Diffusion 2.1 as a base-model while Teamwork leverages the more powerful Stable Diffusion 3 as a base-model, we also train an Stable Diffusion 3 variant of RGB→X. As can be seen in Table 3 (Part II) the retrained RGB→X models both show improvements compared to its more general pretrained counterpart over the InteriorVerse test set with the Stable Diffusion 3 variant slightly outperforming the Stable Diffusion 2.1 variant. However, the performance of the InteriorVerse-specialized Teamwork outperforms prior methods by a significant margin.

Neural Shading. Inspired by X→RGB [Zeng et al. 2024] we also create an intrinsic neural shader that generates from a set of intrinsic components, a synthetic image under unknown random lighting that conforms to the provided components. We adapt Stable Diffusion 3 [Esser et al. 2025] for this task, using the same channels as for the intrinsic image decomposition Teamwork model, except with different input/output roles, i.e., using shading and render channels as output (4 teammates) and all other intrinsic channels as input (6 teammates). The neural shading Teamwork model is trained with 64k random exemplars from InteriorVerse, HyperSim, and CGIntrinsics. In contrast to X→RGB, we do not overload a black image to indicate the absence of an input channel, but instead rely on Teamwork's dynamic (de)activation of teammates. Figure 6 qualitatively compares Teamwork with X→RGB; a quantitative comparison is difficult due to the unknown lighting in the synthesized images. Qualitatively, we see that Teamwork performs similarly to X→RGB.

Intrinsic Image Synthesis. As a final application, we train a novel generative Teamwork model that directly synthesizes intrinsic components from a given prompt. We employ the same teammates as for intrinsic image decomposition and neural rendering, except that all 10 are now output-teammates. We train this generative model on 128k exemplars drawn from InteriorVerse with Gemma-3 [Kamath et al. 2025] generated prompts. The intrinsic synthesis results shown in Figure 7 (first two columns) are qualitatively similar to the intrinsic components from the intrinsic decomposition in Figure 5.

Table 4. Quantitative ablation of Teamwork hyper-parameters on the Stable Diffusion 3 based environment SVBRDF estimation model trained with 16 accumulations per optimizer step, and using rank=16 and steps=4k if not otherwise specified.

Parameter	Render	RMSE			
	Error	Diff.	Spec.	Rough.	Norm.
rank=8	0.286	0.149	0.108	**0.175**	0.0574
rank=16	**0.266**	**0.143**	0.105	0.183	**0.0538**
rank=64	0.273	0.166	0.101	0.229	0.0612
rank=128	0.279	0.151	**0.0991**	0.2	0.0557
steps=2k	0.286	0.147	0.107	**0.179**	0.0566
steps=4k	0.266	0.143	**0.105**	0.183	0.0538
steps=8k	**0.253**	**0.124**	0.108	0.182	**0.0525**

5 Discussion

Hyper-parameter Ablation. To validate Teamwork's hyper-parameter sensitivity with respect to rank and number of training steps, we perform two 1D parameter-scans on the Stable Diffusion 3 based environment SVBRDF estimation model (Table 4). We train each model with 16× gradient accumulation (in lieu of batching) with 4k optimization steps (i.e., 16 × 4k = 64k training exemplars) and rank 16 unless specified otherwise. While the optimal parameters are task specific, we observe that Teamwork is robust across a wide range of hyper-parameters. In general, a too low rank restricts the adaptation capabilities whereas a too high rank offers too much freedom, and thus increases the risk of finding a sub-optimal local minimum. More training steps/exemplars is generally more beneficial, albeit with diminishing returns.

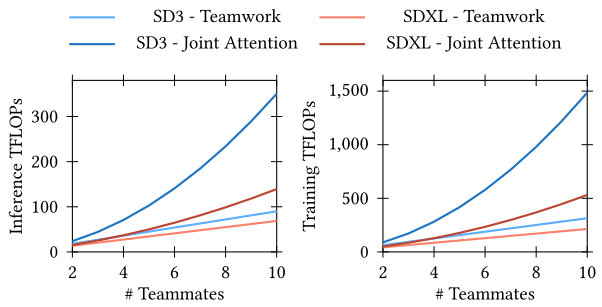


Fig. 8. Comparison of measured computational inference and training FLOPs (for a single diffusion step) of Teamwork (linear) versus Joint Attention (quadratic) for two different base diffusion models with respect to an increasing number of teammates.

Teamwork vs. Joint Attention. The Joint Attention inpainting (Table 1) and SVBRDF estimation (Table 2) variants both underperform compared to Teamwork. We posit that this is mainly due to two reasons. First, from Figure 1 we can see that Teamwork contains 4× as many coordination layers than Joint Attention, and thus Teamwork can more effectively coordinate. Second, Teamwork strongly prefers pixel-aligned input and output images. In contrast, Joint Attention has no such built-in prior and thus needs to learn the positional embeddings and ignore spurious correlations. We expect Joint Attention to perform better on non-pixel aligned inputs.

An additional key benefit of Teamwork over Joint Attention is scalability with respect to resolution and number of teammates. Figure 8 compares inference and training FLOPs (measured using `torch.utils.flop_counter`) for Stable Diffusion XL and Stable Diffusion 3 Teamwork and Joint Attention variants for an increasing number of teammates over a single diffusion step with a 1024 resolution, a rank of 16, and `bf16` datatype. The empirically measured FLOPs confirm the theoretical linear and quadratic complexity of Teamwork and Joint Attention respectively.

Importance of Coordination. A key component of Teamwork is the coordination between teammates. To demonstrate the benefit and necessity of coordination, we train a *set* of Stable Diffusion 3 based intrinsic image decomposition models using ControlLoRA for input expansion (trained on the heterogeneous training set) that each produce one of the intrinsic components without coordination, and evaluate the models on the InteriorVerse test set (Table 3, Part III, first row). Compared to the Teamwork model (Table 3, Part I), we observe a decrease in performance, demonstrating the importance of coordination. However, a low error does not guarantee that the errors between the different outputs are coherent. To demonstrate that Teamwork also helps to improve coherency between output channels, we measure the joint *albedo* × *shading* reconstruction error with respect to the input image, yielding a 0.1127 LPIPS and 0.1272 RMSE for Teamwork versus 0.2443 and 0.2617 respectively for the no-coordination model.

As a final experiment to demonstrate both quality and consistency, we train a *set* of prompt-conditioned intrinsic image synthesis networks that each produce a single component (without coordination) and use a common seed (as in RGB→X) when inferencing each intrinsic component. Figure 7 (last two columns) shows that while each synthesized output is of high quality, the maps are not mutually coherent, resulting in severe ghosting artifacts when recomposing the intrinsic components. In contrast, the Teamwork generative model (first two columns) produces mutually coherent intrinsic components.

Table 5. Quantitative evaluation of the generalization capabilities over HyperSim [Roberts et al. 2021] and Infinigen [Raistrick et al. 2023] of Intrinsic Image Diffusion [Kocsis et al. 2024], RGB→X [Zeng et al. 2024] and a Teamwork model; the latter two are trained on 256k exemplars from InteriorVerse to maximize the difference between training and test sets. Only outputs common between the different models and test sets are included below.

Method	HyperSim			Infinigen (Outdoor)		
	Diffuse		Normal	Diffuse		Normal
	RMSE	LPIPS	RMSE	RMSE	LPIPS	RMSE
Kocsis et al.	0.237	0.418	✗	0.230	0.615	✗
RGB→X	0.340	0.484	0.157	**0.223**	0.596	0.276
Teamwork	**0.215**	**0.356**	**0.125**	0.259	**0.549**	**0.265**

Dynamic (de)activation. To understand the impact of (de)activating teammates on the accuracy of the model, we compare the performance of the intrinsic image decomposition Teamwork model (trained on the heterogeneous training set) on different subsets of (activated) teammates. We consider the following subsets: (a) evaluating each teammate separately (i.e., one active teammate at the time – Table 3, Part III, second line), (b) evaluating only teammates restricted to components present in InteriorVerse (i.e., 7 active output teammates – Table 3, Part III, third line), and (c) evaluating all 9 output teammates jointly (Table 3, Part I, third line). We observe that evaluating each component in isolation incurs a significant loss of accuracy as this effectively disables coordination between the teammates. However, activating only the InteriorVerse components yields lower error than activating all components. We posit that Teamwork attempts to balance errors on the subsets seen during training – InteriorVerse is part of the training set, and thus the Teamwork model is partially optimized for the subset of InteriorVerse components. To validate this thesis, we also train a Teamwork variant with drop-out (i.e., each teammate is deactivated with a 20% probability), and compare its performance on activating each component sequentially versus all components at once (Table 3, Part III, last two lines). Due to drop-out, the resulting model is forced to rely less on coordination between the teammates. Consequently the performance gap between sequential and full inference is significantly reduced. However, the overall performance of the drop-out trained Teamwork is also reduced and is more similar to the no-coordination model.

Generalization Capabilities. Teamwork's ability to leverage pretrained diffusion models can aid in generalizing beyond the training data. To evaluate Teamwork's generalization capabilities, we compare in Table 5 the InteriorVerse trained Intrinsic Image Diffusion [Kocsis et al. 2024], RGB→X [Zeng et al. 2024], and the intrinsic image decomposition Teamwork model on (common intrinsic components from) the HyperSim [Roberts et al. 2021] and Infinigen [Raistrick et al. 2023] test sets. Note that existing real-world intrinsic datasets [Wu et al. 2023] only assume diffuse albedo

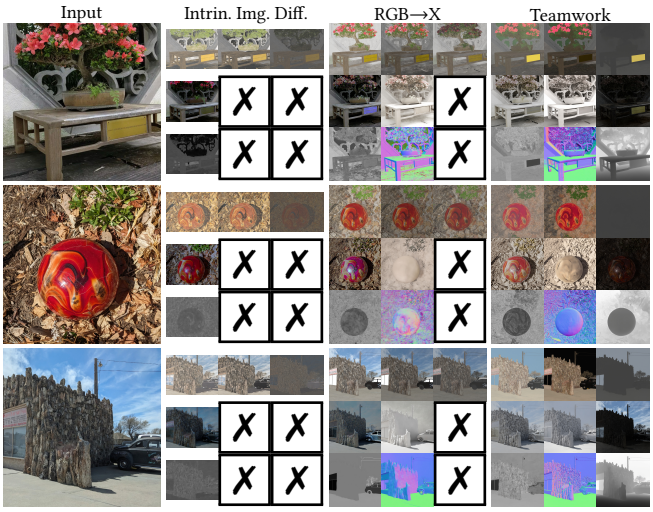


Fig. 9. Qualitative comparison of intrinsic image decompositions of the pretrained Intrinsic Image Diffusion [Kocsis et al. 2024], RGB→X [Zeng et al. 2024], and Teamwork (trained on the heterogeneous data) on real-world photographs.

and are therefore not suited for validating models with an expanded set of intrinsic components. While HyperSim also features indoor scenes, Infinigen contains outdoor scenes, and thus is more distinct from the InteriorVerse training data. Over both test sets, our Teamwork variant exhibits better generalization capabilities than competing diffusion-based intrinsic decomposition methods. Figure 9 further qualitatively demonstrates Teamwork's generalization capabilities on real-world photographs. We observe that Intrinsic Image Diffusion sometimes fails to extract shadows and reflections, while RGB→X struggles to produce consistent normals. Refer to the supplemental material for more real-world side-by-side comparisons.

6 Conclusion

In this paper, we presented Teamwork, an efficient and flexible unified framework for adapting and expanding the number of input and output channels of a pretrained large image diffusion model. Teamwork leverages a novel variation of LoRA to coordinate and adapt between multiple instances of a base diffusion model. We introduced a novel way to add additional control signals to a model as well as an easy method of dynamically activating different teammates. We demonstrated that Teamwork performs similarly or better than prior work that relies on bespoke solutions for a variety of diffusion-derived graphics tasks such as inpainting, SVBRDF estimation, intrinsic decomposition, neural shading, and intrinsic image synthesis.

Acknowledgments

This research was supported in part by NSF grant IIS-1909028.

References

Omer Bar-Tal, Hila Chefer, Omer Tov, Charles Herrmann, Roni Paiss, Shiran Zada, Ariel Ephrat, Junhwa Hur, Guanghui Liu, Amit Raj, Yuanzhen Li, Michael Rubinstein, Tomer Michaeli, Oliver Wang, Deqing Sun, Tali Dekel, and Inbar Mosseri. 2024. Lumiere: A Space-Time Diffusion Model for Video Generation. In *SIGGRAPH Asia 2024 Conference Papers*. Article 94, 11 pages.

Omer Bar-Tal, Lior Yariv, Yaron Lipman, and Tali Dekel. 2023. MultiDiffusion: fusing diffusion paths for controlled image generation. In *ICML*. Article 74, 16 pages.

Dan Biderman, Jacob Portes, Jose Javier Gonzalez Ortiz, Mansheej Paul, Philip Greengard, Connor Jennings, Daniel King, Sam Havens, Vitaliy Chiley, Jonathan Frankle, et al. 2024. Lora learns less and forgets less. *TMLR* (2024).

Black Forest Labs. 2024. FLUX.1. https://blackforestlabs.io/flux-1/.

Andreas Blattmann, Robin Rombach, Huan Ling, Tim Dockhorn, Seung Wook Kim, Sanja Fidler, and Karsten Kreis. 2023. Align your latents: High-resolution video synthesis with latent diffusion models. In *CVPR*. 22563–22575.

Prafulla Dhariwal and Alexander Nichol. 2021. Diffusion Models Beat GANs on Image Synthesis. In *NeurIPS*, Vol. 34. 8780–8794.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, Dustin Podell, Tim Dockhorn, Zion English, and Robin Rombach. 2025. Scaling rectified flow transformers for high-resolution image synthesis. In *ICML*. Article 503.

Rohit Girdhar, Mannat Singh, Andrew Brown, Quentin Duval, Samaneh Azadi, Sai Saketh Rambhatla, Akbar Shah, Xi Yin, Devi Parikh, and Ishan Misra. 2024. Factorizing Text-to-Video Generation by Explicit Image Conditioning. In *ECCV*. 205–224.

Agrim Gupta, Lijun Yu, Kihyuk Sohn, Xiuye Gu, Meera Hahn, Fei-Fei Li, Irfan Essa, Lu Jiang, and José Lezama. 2025. Photorealistic video generation with diffusion models. In *ECCV*. 393–411.

Wu Hecong. 2023. *ControlLoRA: A Lightweight Neural Network To Control Stable Diffusion Spatial Information.* https://github.com/HighCWu/ControlLoRA

Amir Hertz, Andrey Voynov, Shlomi Fruchter, and Daniel Cohen-Or. 2023. Style aligned image generation via shared attention. In *CVPR*.

Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. 2021. CLIPScore: A Reference-free Evaluation Metric for Image Captioning. In *EMNLP*.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising Diffusion Probabilistic Models. In *NeurIPS*, Vol. 33. 6840–6851.

Jonathan Ho, Chitwan Saharia, William Chan, David J Fleet, Mohammad Norouzi, and Tim Salimans. 2022a. Cascaded Diffusion Models for High Fidelity Image Generation. *Journal of Machine Learning Research* (2022).

Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J Fleet. 2022b. Video diffusion models. *NeurIPS* 35 (2022), 8633–8646.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *ICML*. 2790–2799.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. Lora: Low-rank adaptation of large language models. In *ICLR*.

Zahra Kadkhodaie and Eero Simoncelli. 2021. Stochastic Solutions for Linear Inverse Problems using the Prior Implicit in a Denoiser. In *NeurIPS*, Vol. 34. 13242–13254.

Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, et al. 2025. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786* (2025).

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. 2022. Elucidating the Design Space of Diffusion-Based Generative Models. In *NeurIPS*.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. 2017. Overcoming catastrophic forgetting in neural networks. *PNAS* 114, 13 (2017), 3521–3526.

Sebastian Koch, Albert Matveev, Zhongshi Jiang, Francis Williams, Alexey Artemov, Evgeny Burnaev, Marc Alexa, Denis Zorin, and Daniele Panozzo. 2019. ABC: A Big CAD Model Dataset For Geometric Deep Learning. In *CVPR*.

Peter Kocsis, Vincent Sitzmann, and Matthias Nießner. 2024. Intrinsic Image Diffusion for Single-view Material Estimation. In *CVPR*.

Yuheng Li, Haotian Liu, Qingyang Wu, Fangzhou Mu, Jianwei Yang, Jianfeng Gao, Chunyuan Li, and Yong Jae Lee. 2023. GLIGEN: Open-Set Grounded Text-to-Image Generation . In *CVPR*.

Zhengqi Li and Noah Snavely. 2018. Cgintrinsics: Better intrinsic image decomposition through physically-based rendering. In *ECCV*. 371–387.

Zhaojiang Lin, Andrea Madotto, and Pascale Fung. 2020. Exploring Versatile Generative Language Model Via Parameter-Efficient Transfer Learning. In *Findings of the Association for Computational Linguistics: EMNLP 2020*. 441–459.

Jundan Luo, Duygu Ceylan, Jae Shin Yoon, Nanxuan Zhao, Julien Philip, Anna Frühstück, Wenbin Li, Christian Richardt, and Tuanfeng Wang. 2024. IntrinsicDiffusion: Joint Intrinsic Layers from Latent Diffusion Models. In *ACM SIGGRAPH 2024 Conference Papers*. Article 74.

Konstantin Mishchenko and Aaron Defazio. 2024. Prodigy: an expeditiously adaptive parameter-free learner. In *ICML*. Article 1458.

O. Nalbach, E. Arabadzhiyska, D. Mehta, H.-P. Seidel, and T. Ritschel. 2017. Deep Shading: Convolutional Neural Networks for Screen Space Shading. *Comput. Graph.*

Forum 36, 4 (July 2017), 65–78.

Alexander Quinn Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob Mcgrew, Ilya Sutskever, and Mark Chen. 2022. GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models. In *ICML*. 16784–16804.

Alexander Raistrick, Lahav Lipson, Zeyu Ma, Lingjie Mei, Mingzhe Wang, Yiming Zuo, Karhan Kayan, Hongyu Wen, Beining Han, Yihan Wang, Alejandro Newell, Hei Law, Ankit Goyal, Kaiyu Yang, and Jia Deng. 2023. Infinite Photorealistic Worlds Using Procedural Generation. In *CVPR*. 12630–12641.

Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. 2022. Hierarchical Text-Conditional Image Generation with CLIP Latents. *arXiv preprint arXiv:2204.06125* (2022).

Mike Roberts, Jason Ramapuram, Anurag Ranjan, Atulit Kumar, Miguel Angel Bautista, Nathan Paczan, Russ Webb, and Joshua M. Susskind. 2021. Hypersim: A Photorealistic Synthetic Dataset for Holistic Indoor Scene Understanding. In *ICCV*.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-resolution image synthesis with latent diffusion models. In *CVPR*. 10684–10695.

Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. 2022. Photorealistic text-to-image diffusion models with deep language understanding. *NeurIPS* 35 (2022), 36479–36494.

C. Saharia, J. Ho, W. Chan, T. Salimans, D. J. Fleet, and M. Norouzi. 2023. Image Super-Resolution via Iterative Refinement. *IEEE TPAMI* 45, 04 (apr 2023), 4713–4726.

Sam Sartor and Pieter Peers. 2023. MatFusion: A Generative Diffusion Model for SVBRDF Capture. In *SIGGRAPH Asia 2023 Conference Papers*.

Hiroshi Sasaki, Chris G Willcocks, and Toby P Breckon. 2021. Unit-ddpm: Unpaired image translation with denoising diffusion probabilistic models. *arXiv preprint arXiv:2104.05358* (2021).

Tamar Rott Shaham, Tali Dekel, and Tomer Michaeli. 2019. Singan: Learning a generative model from a single natural image. In *CVPR*. 4570–4580.

Reece Shuttleworth, Jacob Andreas, Antonio Torralba, and Pratyusha Sharma. 2024. Lora vs full fine-tuning: An illusion of equivalence. *arXiv preprint arXiv:2410.21228* (2024).

Uriel Singer, Adam Polyak, Thomas Hayes, Xi Yin, Jie An, Songyang Zhang, Qiyuan Hu, Harry Yang, Oron Ashual, Oran Gafni, et al. 2023. Make-A-Video: Text-to-Video Generation without Text-Video Data. In *ICLR*.

Vasu Singla, Kaiyu Yue, Sukriti Paul, Reza Shirkavand, Mayuka Jayawardhana, Alireza Ganjdanesh, Heng Huang, Abhinav Bhatele, Gowthami Somepalli, and Tom Goldstein. 2024. From Pixels to Prose: A Large Dataset of Dense Image Captions. *arXiv preprint arXiv:2406.10328* (2024).

Sisap. 2024. Laion2B. https://sisap-challenges.github.io/2024/datasets/.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. 2021. Score-Based Generative Modeling through Stochastic Differential Equations. In *ICLR*.

Stability AI. 2022. Stable Diffusion V2.1. https://huggingface.co/stabilityai/stable-diffusion-2-1.

Stability AI. 2023a. DeepFloyd. https://github.com/deep-floyd/IF.

Stability AI. 2023b. Stable Diffusion XL. https://huggingface.co/docs/diffusers/en/using-diffusers/sdxl.

Stability AI. 2024. Stable Diffusion V3 - Inpainting. https://huggingface.co/alimama-creative/SD3-Controlnet-InpaintingControlNet.

Giuseppe Vecchio, Rosalie Martin, Arthur Roullier, Adrien Kaiser, Romain Rouffet, Valentin Deschaintre, and Tamy Boubekeur. 2024. ControlMat: A Controlled Generative Approach to Material Capture. *ACM Trans. Graph.* 43, 5, Article 164 (Sept. 2024).

Patrick von Platen, Suraj Patil, Anton Lozhkov, Pedro Cuenca, Nathan Lambert, Kashif Rasul, Mishig Davaadorj, Dhruv Nair, Sayak Paul, William Berman, Yiyi Xu, Steven Liu, and Thomas Wolf. 2022. Diffusers: State-of-the-art diffusion models. https://github.com/huggingface/diffusers.

Jianyi Wang, Kelvin C.K. Chan, and Chen Change Loy. 2023. Exploring CLIP for assessing the look and feel of images. In *AAAI*. Article 284.

Jiaye Wu, Sanjoy Chowdhury, Hariharmano Shanmugaraja, David Jacobs, and Soumyadip Sengupta. 2023. Measured albedo in the wild: Filling the gap in intrinsics evaluation. In *ICCP*.

Zheng Zeng, Valentin Deschaintre, Iliyan Georgiev, Yannick Hold-Geoffroy, Yiwei Hu, Fujun Luan, Ling-Qi Yan, and Miloš Hašan. 2024. RGB ↔ X: Image decomposition and synthesis using material-and lighting-aware diffusion models. In *ACM SIGGRAPH 2024 Conference Papers*. Article 75.

Friedemann Zenke, Ben Poole, and Surya Ganguli. 2017. Continual learning through synaptic intelligence. In *ICML*. 3987–3995.

Lvmin Zhang and Maneesh Agrawala. 2024. Transparent Image Layer Diffusion using Latent Transparency. *ACM Trans. Graph.* 43, 4, Article 100 (July 2024).

Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. 2023. Adding conditional control to text-to-image diffusion models. In *CVPR*. 3836–3847.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. 2018. The Unreasonable Effectiveness of Deep Features as a Perceptual Metric. In *CVPR*.

Jingsen Zhu, Fujun Luan, Yuchi Huo, Zihao Lin, Zhihua Zhong, Dianbing Xi, Rui Wang, Hujun Bao, Jiaxiang Zheng, and Rui Tang. 2022. Learning-Based Inverse Rendering of Complex Indoor Scenes with Differentiable Monte Carlo Raytracing. In *SIGGRAPH Asia 2022 Conference Papers*. ACM, Article 6.