Readout Guidance: Learning Control from Diffusion Features

Supplementary Material

8. Readout Guidance

8.1. Guidance Hyperparameters

For all tasks, we use 100 sampling timesteps, which we find produces results that are faithful to the input control while still maintaining a reasonable runtime. Our guidance update is rescaled by $2e^{-2}$ to allow the user-defined guidance weight to occupy a more intuitive range of $[0.0, 1.0]$, where 0 is no additional guidance, and 1 is very strong guidance (values > 1 are also valid, but typically not desirable). Below, we describe the set of hyperparameters that vary between applications shown in the paper. For each of these, the values used for generating our results are shown in Table 1.

text_weight: the weight of text classifier-free guidance [21], which ranges between $[0, \infty]$.

rg_weight: the weight of our guidance signal, which ranges between $[0, \infty]$. As mentioned above and demonstrated in Figure 13, values much larger than 1.0 can result in visual artifacts. The effects of varying this weight in can be seen in Figure 6 of the main text.

rg_ratio: the span of timesteps [start, end] to apply our guidance to, expressed as a fraction of the total number of timesteps. These values range between $[0, 1.0]$. We find that early diffusion steps generally determine structure and composition, while late diffusion steps determine high frequency detail and photorealism. For the appearance preservation and identity consistency tasks, we find it helpful to set the start value > 0.0 to retain structural diversity from a random seed (setting start $= 0.0$ can cause the reference image's structure to be copied in addition to the subject's identity). For all tasks, we set end < 1.0, to avoid the guidance signal competing with photorealism.

eta (η): the stochasticity of the sampling process, which ranges between [0, 1.0]. Setting $\eta = 0.0$ is traditional deterministic DDIM sampling [54] and $\eta = 1.0$ is highly stochastic. Empirically we also find that using $\eta = 1.0$ is especially critical for the spatially aligned control task, where using $\eta = 0.0$ can result in visual artifacts, which we show in Figure 13. We hypothesize that stochasticity is required when regressing against a user input control, which is more out of domain compared to a reference set of diffusion features.

8.2. Update Rule

We observe that our guidance process is sensitive to the magnitude of the gradient signal at each sampling step. In this section, we explore different mechanisms for scaling the gradient signal, inspired by common practices in gradient descent optimization. First, recall that our method derives

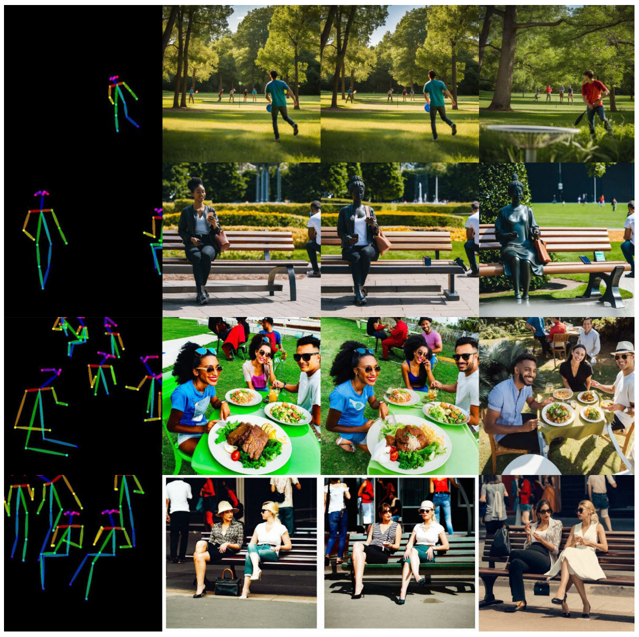
(a) Pose Control (b) w/ τ_t (c) w/o τ_t (d) Normalized

Figure 11. **Update Rule Comparison**: For the same pose control, we compare outputs when using timestep rescaling as was done in original classifier guidance [13], removing the timestep rescaling factor, and normalizing the gradient with the L2-norm.

from the original classifier guidance (CG) update rule [13], which is written as follows:

$$\hat{\epsilon}_t \leftarrow \epsilon_\theta(x_t) - \sqrt{1-\bar{\alpha}_t}\nabla_{x_t}\log p_\phi(y|x_t)$$

where, in our case, we replace the gradient term $\log p_\phi(y|x_t)$ with a distance function over the readout $d(r, f_\psi(x_t))$. For simplicity, we can rewrite this update rule as a standard optimizer update:

$$w_t \leftarrow w_{t-1} - \alpha \cdot \tau_t \cdot \nabla \mathcal{L},$$

where w represents the input/output noise, τ_t is a timestep-dependent rescaling, α is a configurable learning rate, and $\mathcal{L}$ is the loss or distance function. Following this formulation, we explore a couple of alternative update rules for applying guidance, with qualitative comparisons shown in Figure 11. First, since our readout heads are already timestep-conditional, we ablate the necessity of timestep rescaling:

$$w_t \leftarrow w_{t-1} - \alpha \cdot \nabla \mathcal{L},$$

We use the learning rate $\alpha = 200$ for both this variant and the one with τ_t. We also experiment with normalizing the

Task	text_weight	rg_weight	rg_ratio	η
Drag-Based Manipulation (Real Images)	3.5	1.0	[0.0, 0.5]	0.0
Drag-Based Manipulation (Generated Images)	7.5	1.0	[0.0, 0.5]	0.0
Appearance Preservation (Generated Images)	7.5	1.0	[0.05, 0.5]	0.0
Identity Consistency (Generated Images)	7.5	1.0	[0.05, 0.8]	1.0
Spatially Aligned Control (Generated Images)	7.5	0.5	[0.0, 0.5]	1.0

Table 1. **Guidance Hyperparameters:** We recommend the following hyperparameter settings for each task when using Readout Guidance.

gradient by the L2-norm, as might be done in an Adam [30] update (without the historical moving average):

$$w_t \leftarrow w_{t-1} - \alpha \cdot \frac{\nabla \mathcal{L}}{\sqrt{\nabla \mathcal{L}^2 + \epsilon}},$$

where ϵ is a small positive constant to prevent division by zero. For this variant, we use $\alpha = 2e^{-2}$. Unlike the other two update rules, the normalized gradient variant has an effective step size bounded only by the learning rate α, and is therefore invariant to the magnitude of the gradient $\nabla \mathcal{L}$ [30].

Empirically, we find that when using our readout heads, timestep rescaling in the update rule has a negligible effect, often producing outputs that are extremely similar to those without rescaling (Figure 11b and Figure 11c). We further notice that making the readout heads timestep-conditional is very important—removing this conditioning results in improbable outputs that respect the input control but quickly fall off the manifold of natural imagery. A similar failure mode occurs if the magnitude of the loss $\nabla \mathcal{L}$ is too large. For this reason, we find that the normalized gradient update rule (Figure 11d) produces the highest fidelity results, as it causes the update step size to be agnostic to gradient scale. Note the difference between Figure 11c and 11d, where the un-normalized update rules sometimes exhibit contrast and saturation artifacts. All the results in our paper use the normalized variant, although our method is also effective with the other two update rules.

8.3. Sampling Compute

Here we report the runtime and memory consumption on an Nvidia A100 GPU. For the pose control task, sampling with Readout Guidance takes 32.7s on SDv1-5 (vs. 10.7s with only text guidance) and consumes 11.9GB of VRAM (vs. 8.5GB). On SDXL, sampling with Readout Guidance takes 44.0s (vs. 14.2s) and consumes 27.7GB of VRAM (vs. 15.6GB). For the drag-based manipulation task, sampling with Readout Guidance takes 55.4s (vs. 16.6s) on SDv1-5 and consumes 19.2GB of VRAM (vs. 8.3GB). On SDXL, Readout Guidance takes 75.0s (vs. 23.4s) and consumes 37.6GB of VRAM (vs. 15.8GB).

It is important to note that we use a naive initial implementation that has not been optimized for memory usage—*i.e.*, we retain all relevant buffers in memory: all features, gradients, target control signals. A number of optimizations would likely reduce memory consumption to less than half the reported numbers, *e.g.*, not batching the conditional and unconditional inputs together, pre-computing and caching reference features for pairwise tasks, among others. Using the current implementation, the most memory-efficient setting is able to run on readily accessible GPUs such as an Nvidia T4 (16GB VRAM) available via Google Colab, and we anticipate further memory optimization should enable our method's use on even lower-end GPUs.

8.4. Training Compute

Here we share the training time for each readout head on an Nvidia A100 GPU. We train the pose head on PascalVOC with a batch size of 8 image / control pairs for at most 5k steps, which takes 37 min for SDv1-5 and 2 hr 25 min for SDXL. The depth head takes 41 min for SDv1-5 and 2 hr 40 min for SDXL. The edge head takes 41 min for SDv1-5 and 2 hr 38 min for SDXL. We train the appearance similarity head on DAVIS with a batch size of 1 anchor / positive / negative triplet for at most 1k steps, which takes 08 min for SDv1-5 and 24 min for SDXL. We train the correspondence feature head on DAVIS with a batch size of 1 frame / frame pair for at most 10k steps, which takes 48 min for SDv1-5 and 2 hr 15 min for SDXL.

8.5. Readout Head Architecture

In Figure 12 we depict a detailed architecture diagram of our readout heads. We share a common architecture for all readout heads, building off of the aggregation network proposed in Diffusion Hyperfeatures [36]. For each raw decoder feature map, Diffusion Hyperfeatures uses bottleneck layers [19] to standardize the channel count (left column), then aggregates these standardized feature maps with a learned weighted sum (middle column). Our main modification to adapt this architecture for guidance is to make the bottleneck layers timestep-conditional, implemented as the blue linear layer shown in the left column of Figure 12. More specifically, we use the U-Net's pre-trained timestep embedding layers to get a timestep embedding, learn a projection layer, and add this projected embedding to the feature map during the standardization process. Our design was inspired by the Stable Diffusion implementation of timestep conditioning in their ResNet blocks [24]. For the case of the spatially

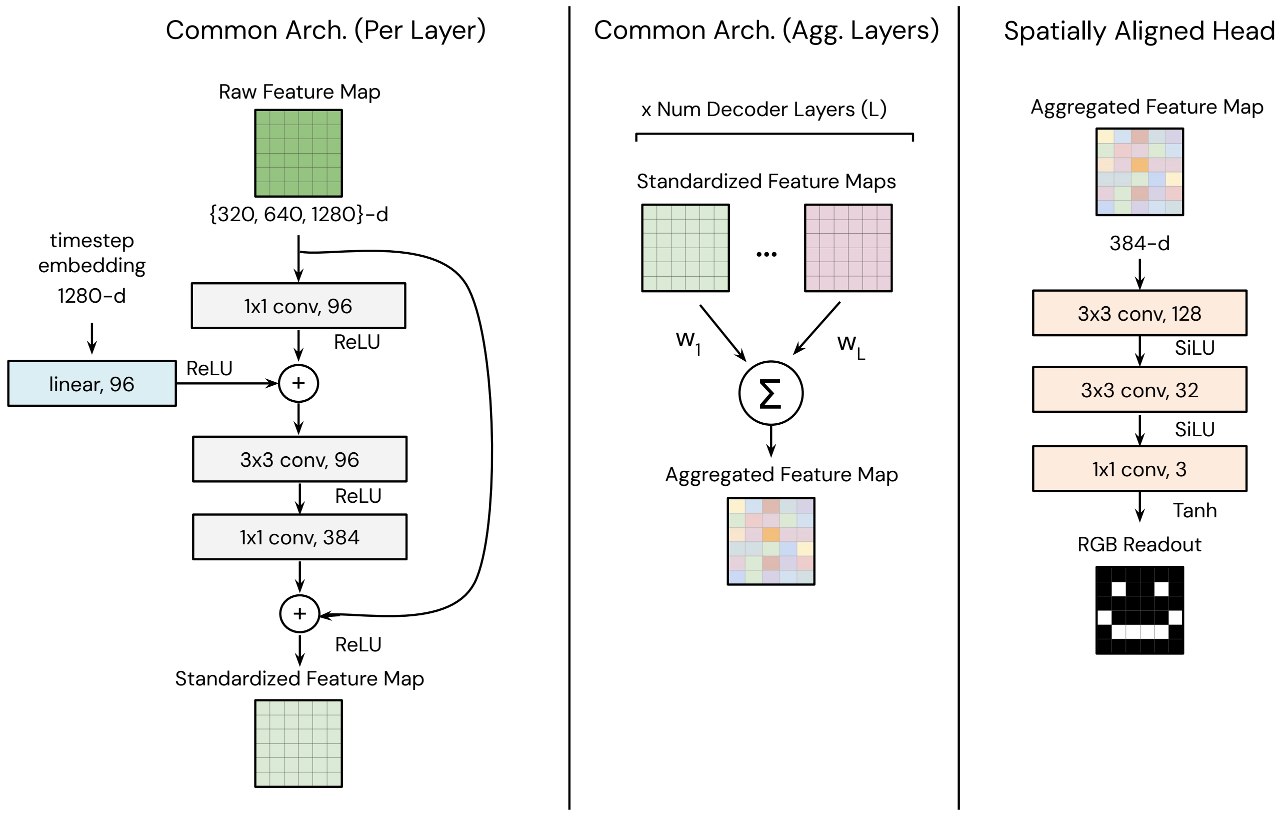

Figure 12. **Architecture Diagram**: We share a common architecture across all readout heads, where we first convert the raw feature map to a feature map of a standardized channel count and spatial resolution (left), aggregate feature maps from all decoder layers (middle), and optionally convert the aggregated feature map into an RGB readout (right).

aligned heads (right column), we make another modification where we add convolutional layers (yellow) that convert the multi-channel aggregated feature map into a three-channel RGB image. Our design was inspired by the output convolutions used in MiDaS [41, 49] and the SiLU (a soft version of ReLU) activations used in ControlNet [70]. We use Tanh for our final activation function to ensure that the output is bounded between $[-1, 1]$, rather than unbounded.

8.6. Improbable Images

In Figure 13 we show a few failure cases. Although these images fully respect the input pose control, they start to diverge from the natural image manifold, tending towards cartoonish colors or surreal imagery with relatively textureless scenes. These types of artifacts are likely the result of a trade-off between the guidance target and photorealism (as defined by the base model's log-likelihood)—and can be resolved by lowering the readout guidance weight, stopping our guidance earlier in the diffusion process, or starting from a different initial seed.

9. Relative Control

9.1. Identity Consistency

In this task, we seek to generate an image (or set of images) of the same person or character. Given an input image, a sampling process (otherwise guided by a text prompt describing a different scene) is encouraged to produce an image that contains a person with the same identity as the input image. To accomplish this, we use a specialized version of the appearance head trained on a dataset of face photographs [27]. Since this dataset consists of mostly tight crops around the face, we apply our guidance constraint only at pixels which we determine to belong to a face (*i.e.*, using an off-the-shelf face detector [58, 64]). While we only apply guidance within the face region, this can cause the overall sampling trajectory to change, which can in turn change the appearance of the background. To counteract this, we also apply our (standard, *i.e.*, not identity-specialized) appearance guidance on the rest of the image, but towards the original sampling trajectory, allowing the background to remain unchanged. In Figure 14 we ablate the effect of removing this appearance similarity guidance (col. 4), showing that the background drifts more in appearance but often the identity is more closely

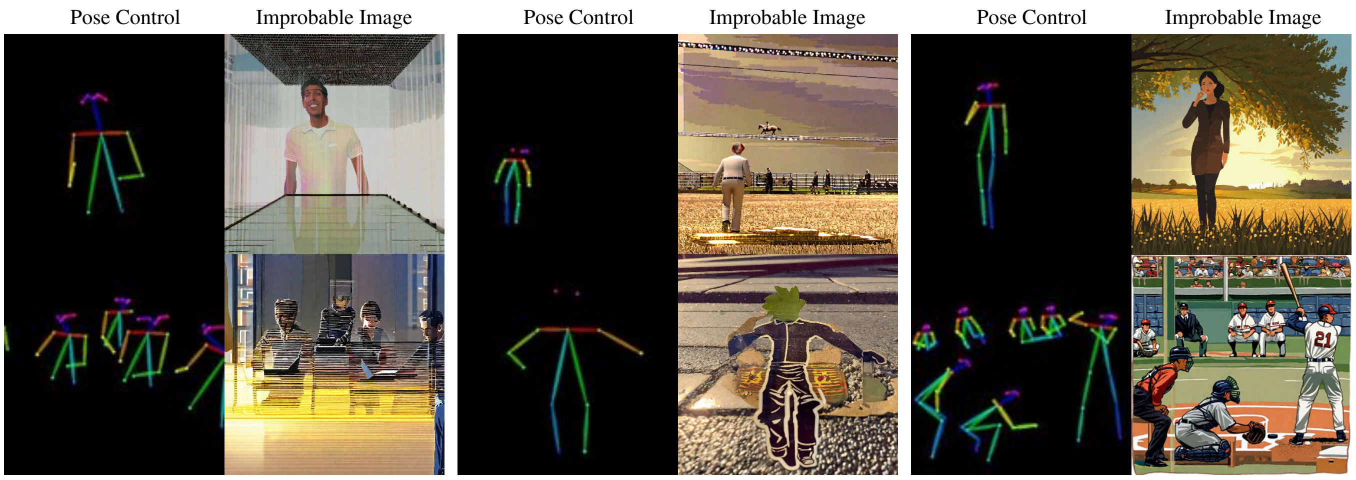

Figure 13. **Improbable Images**: We show images generated with our pose head using a deterministic sampler ($\eta = 0.0$), rather than the recommended stochastic sampler ($\eta = 1.0$). These types of generations also arise when the guidance weight is set too high.

Method	Added Training Data	Added Params / Size	$\text{PCK}_{\alpha=0.05}$ (↑)	$\text{PCK}_{\alpha=0.1}$ (↑)	$\text{PCK}_{\alpha=0.2}$ (↑)
SDv1-5					
No Control	-	-	0.49	1.55	5.42
$\text{RG}_{8.5\text{k}}$	8.5k Images	8.5M / 49MB	19.91	36.10	50.71
ControlNet [70]	200k Images	361M / 1.4GB	42.34	60.22	73.38
ControlNet + $\text{RG}_{8.5\text{k}}$	200k + 8.5k Images	370M / 1.4GB	**55.82**	**77.47**	**86.93**
SDXL					
No Control	-	-	1.19	2.94	10.40
RG_{100}	100 Images	5.9M / 35MB	15.24	28.19	41.62
$\text{RG}_{1\text{k}}$	1k Images	5.9M / 35MB	19.65	29.41	43.17
$\text{RG}_{8.5\text{k}}$	8.5k Images	5.9M / 35MB	24.29	37.07	46.21
T2IAdapter [40]	3M Images	79M / 302MB	24.06	35.92	49.49
T2IAdapter+$\text{RG}_{8.5\text{k}}$	3M + 8.5k Images	85M / 337MB	**54.54**	**72.04**	**82.18**

Table 2. **Pose Control Comparison:** We compute the percentage of correct keypoints (PCK) at varying thresholds α between the input pose and the pose of the generated image on 100 random images of humans from the MSCOCO [34] validation set.

Method	CMMD (↓)
SDXL	
No Control	1.374
RG_{100}	1.305
RG_{1k}	1.243
$\text{RG}_{8.5k}$	1.134
T2IAdapter	0.817
T2IAdapter + $\text{RG}_{8.5k}$	0.766

Table 3. **Pose Control Image Quality:** We also compute the CLIP Maximum Mean Discrepancy (CMMD) [25] between the distribution of real images and generated images on the same set of 100 MSCOCO images described in Table 2.

matched. We also compare to the result of the concurrent work IP-Adapter [31, 67] (col. 5), which trains an adapter to condition on the CLIP image embedding [47] of a reference image. Although IP-Adapter can also be used to borrow a reference identity (face), unlike our method, it cannot inject this identity into a particular sampled image.

10. Spatially Aligned Control

10.1. Readout Evolution

In Figure 15 we visualize the evolution of our readouts over the sampling process. The typical visualization of the denoising process, or the evolution of x_t (row 1), can be relatively uninterpretable across a majority of the process. A more informative visualization is typically the intermediate prediction of the clean image (row 2). Our readouts (row 3-5) offer an additional tool for probing the diffusion process, derived from the features themselves rather than the sampling formulation. We find that the image composition is largely determined early in the diffusion process, where our readouts show a central figure in the foreground within the first step of sampling and a coarse silhouette within the first 10%.

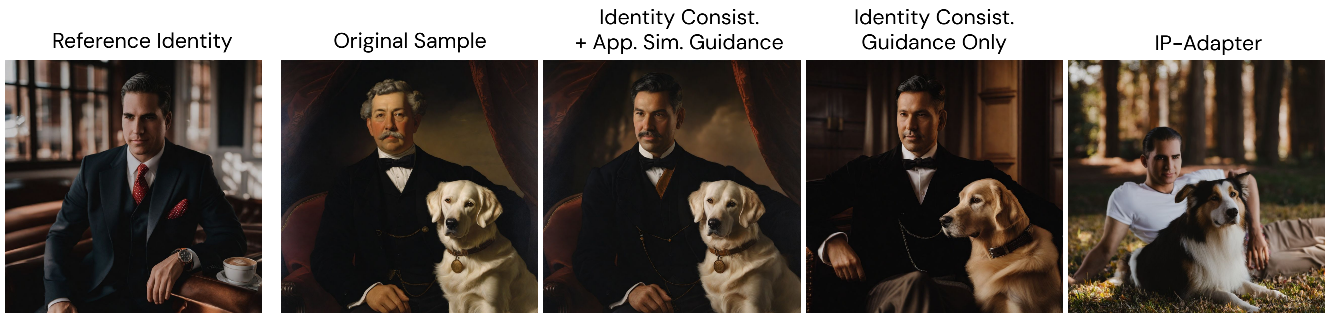

Figure 14. **Identity Consistency Ablation**: Our full method uses the identity consistency head on the face and appearance similarity head to preserve the background. We also ablate using only the identity consistency head. Concurrent work IP-Adapter [67] does not support injecting identity into a specific sampled image.

Evidently, our readouts are useful for guidance because they are informative across the entire diffusion process, even at early noisy timesteps.

10.2. Quantitative Comparison

In Table 2 we compare against fine-tuned conditional models on the task of pose control. We opt for using SDXL as our base model, since it significantly outperforms SDv1-5 in visual quality. Unfortunately, ControlNet [70] only has officially released weights on SDv1-5, so our SDXL comparisons are only with officially released checkpoints from T2IAdapter [40]. We do, however, provide additional supplementary comparisons to ControlNet on SDv1-5. In this evaluation, we use OpenPose [8] to extract the pose of the generated images from each method then compute the percentage of correct keypoints (PCK) against the input control. When a pose prediction has multiple instances, we take the highest scoring PCK across all combinations of instances between the input and synthesized image. We then average the PCKs across all samples. We compute the PCK at varying error thresholds, $\alpha \in \{0.05, 0.1, 0.2\}$. We also report the amount of added training data that was used to fine-tune the control method, as well as the added parameter count. Note that the parameter count of our method scales with the number of decoder layers (SDXL has 9 vs SDv1-5 has 12 layers), so our SDXL heads are actually more parameter-efficient than our SDv1-5 heads. SDXL trades off fewer decoder blocks with additional mid blocks, which we found in early experiments did not improve the quality of our spatially-aligned readouts. Finally, we see that our pose head trained on the vanilla base model can be transferred to guide a fine-tuned conditional model, further improving the performance of ControlNet [70] and T2IAdapter [40] by 13% and 30% PCK ($\alpha = 0.05$) respectively. The compatibility between Readout Guidance and these fine-tuned conditional models derives from the fact that all methods keep the base model frozen, and therefore operate on the same common feature space as the original base model.

We also calculate image quality as measured by the CLIP Maximum Mean Discrepancy [25]. We compare the distributions of the real MSCOCO images, from which we derived the pose control and text prompt, with the generated images from each method. We opt to use CMMD as it is unbiased with respect to the evaluation sample size, unlike FID which is not meaningful on small image sets. As seen in Table 3, our method actually slightly improves image quality when compared to the base model (No Control vs. $\text{RG}_{8.5k}$, T2IAdapter vs. T2IAdapter + $\text{RG}_{8.5k}$). We also see that training on more data leads to increasingly better image quality (RG_{100} vs. RG_{1k} vs. $\text{RG}_{8.5k}$). This likely explains why there is a leap in image quality when also using T2IAdapter, as it was trained on orders of magnitude more data ($\sim$3M images).

10.3. Additional Examples

We show additional examples of spatially aligned control from our full method trained on all images in PascalVOC [16]. We show additional examples of pose control in Figure 16 and Figure 17, depth control in Figure 18 and Figure 19, and edge control in Figure 20 and Figure 21. In Figure 17, we see that our pose readout can sometimes hallucinate extra detail, for example detecting fingers (col 2, row 4) or detecting the dog (col 1, row 2), even when they are not covered by the pose representation. Similarly, in Figure 18, while our depth readout can reliably distinguish the foreground and background, it can struggle with producing a smooth and uniform depth across the entire object. We show additional qualitative examples using our guidance on top of ControlNet [70] (Figure 22) and T2I-Adapter [40] (Figure 23). We also compare the variants of our model trained on 100, 1k, and 8.5k images in Figure 24. For simpler poses (col 1, row 8), all models perform equally well whereas for more difficult examples (col 1, row 4) our method trained on all 8.5k images performs the best.

11. Text Prompts

We provide the text prompts used to generate the images in the main text below.

- Figure 2: *"professional headshot of a woman with the eiffel tower in the background"*, *"photo of a squirrel with a hamburger at its feet"*, *"photo of a beagle hovering mid-air"*, *"a bear dancing on times square"*
- Figure 4: *"two men on a tennis court shaking hands over the net "*, *"a group of people playing with Frisbee's on the grass"*, *"a giraffe standing in a straw field next to shrubbery"*, *"two birds standing next to each other on a branch"*, *"a mountain goat stands on top of a rock on a hill"*, *"a man riding a horse followed by a dog"*
- Figure 6: *"A dog is walking down the street"*, *"An astronaut is skiing down the hill"*
- Figure 7: *"oil painting half body portrait of a man on a city street in copenhagen"*, *"portrait of a man and his dog"*, *"photo of a female firefighter in the forest"*, *"photo of a woman at the zoo with a seal"*
- Figure 5: *"a photo of a tiger"*, *"a photo of a raccoon"*, *"a photo of a rabbit"*, *"a photo of a man holding a crocodile"*, *"an alpine ibex with horns"* [1], *"wolf looking around"* [2]
- Figure 8: *"a woman that is sitting on a couch holding a remote"*, *"a young lady is playing a baseball bat game"*, *"a man wearing an apron peering into the bottom of an open fridge"*, *"a person on a surfboard on the water"*
- Figure 9: *"a person riding a scooter with folded cardboard"*, *"a group of people in a field playing frisbee"*

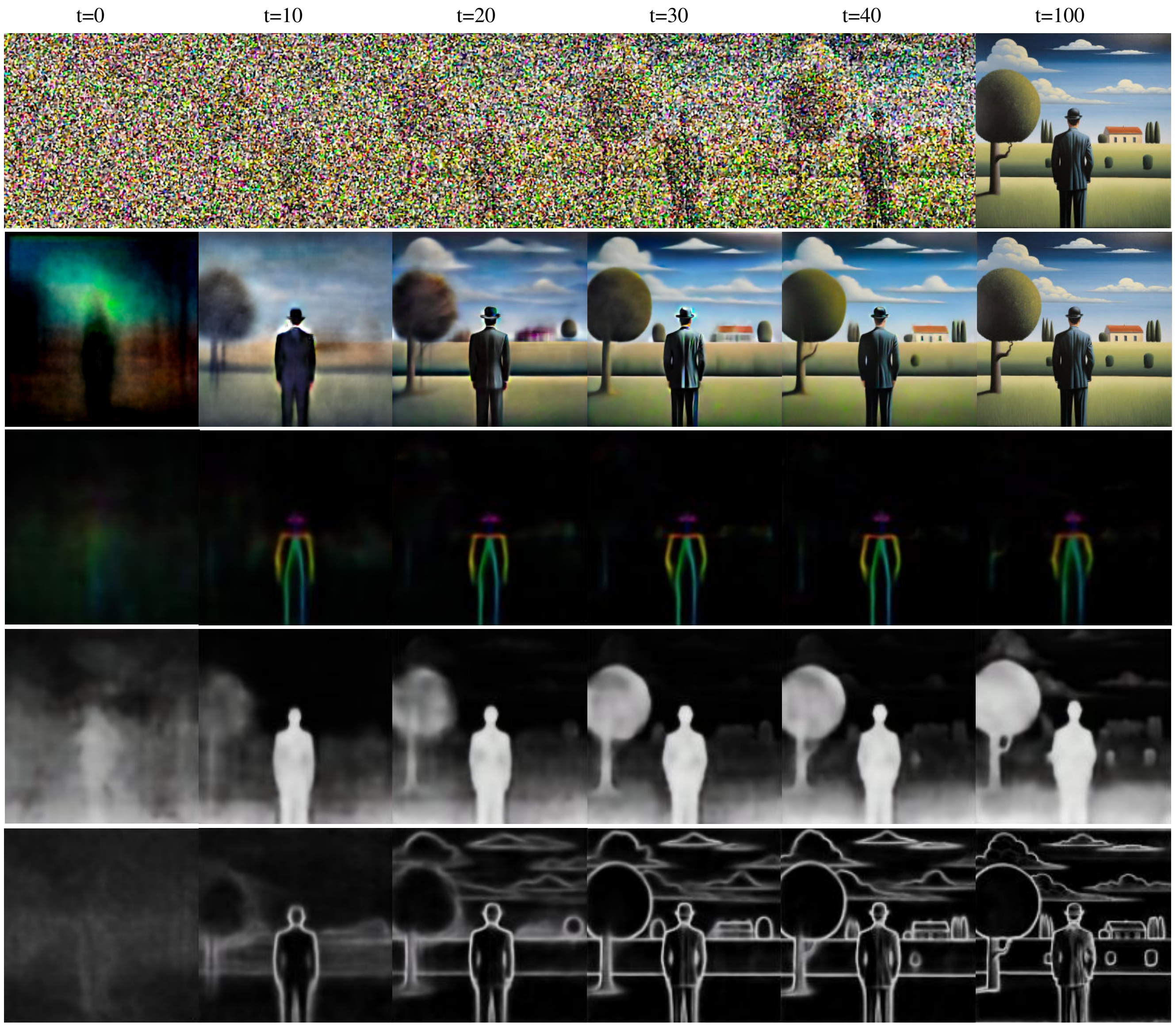

Figure 15. **Readout Evolution**: We synthesize an image with the prompt *"painting of man standing outdoors in the style of magritte"*, and show (from top to bottom) the x_t, predicted clean image [54], pose readout, depth readout, and edge readout across the diffusion process. A large portion of the image composition is already determined in the first 10% of the diffusion process ($t = 10$), as reflected by the crispness of our readouts which become further refined over time.

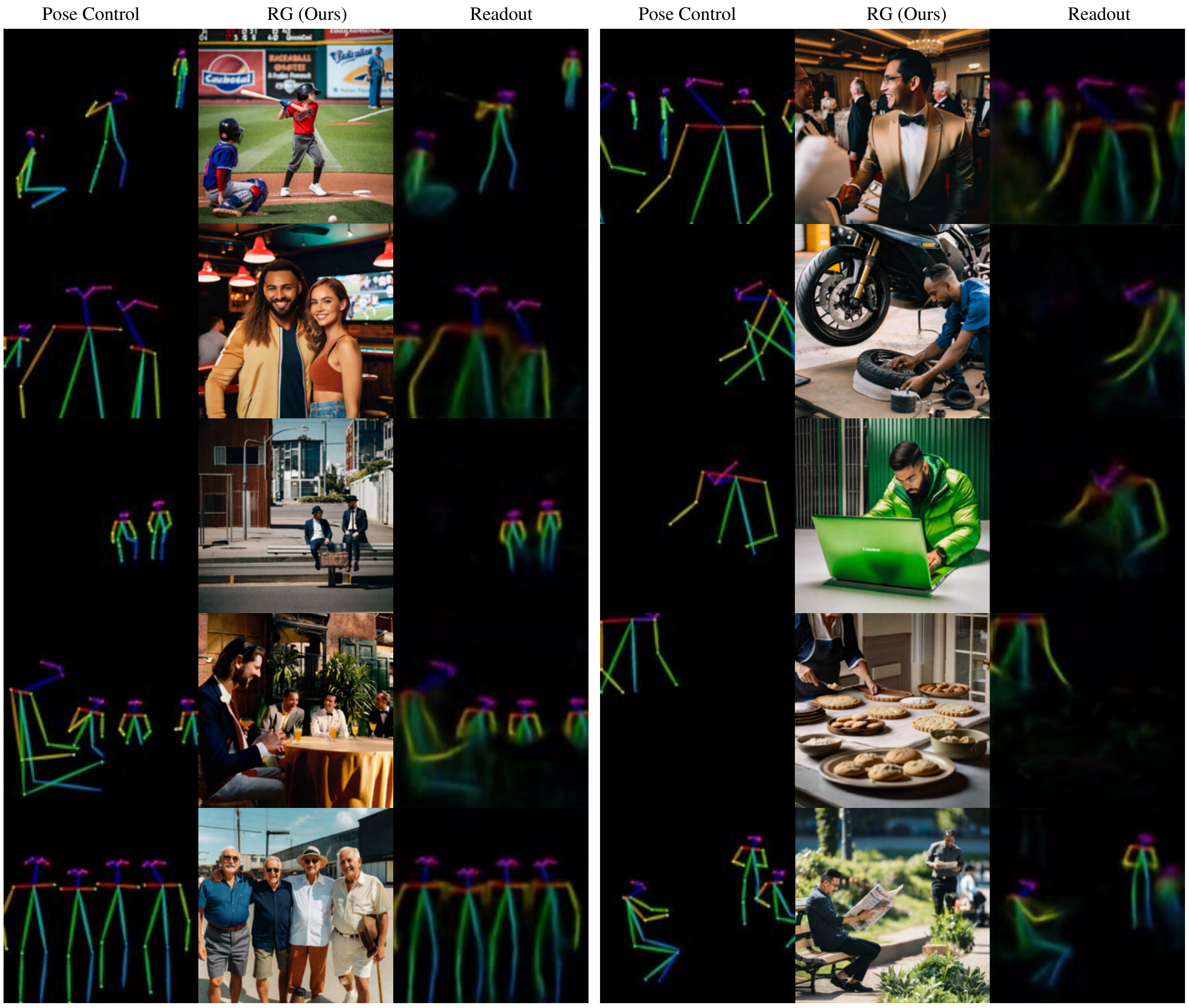

Figure 16. **Pose Control**: Additional examples with our pose head.

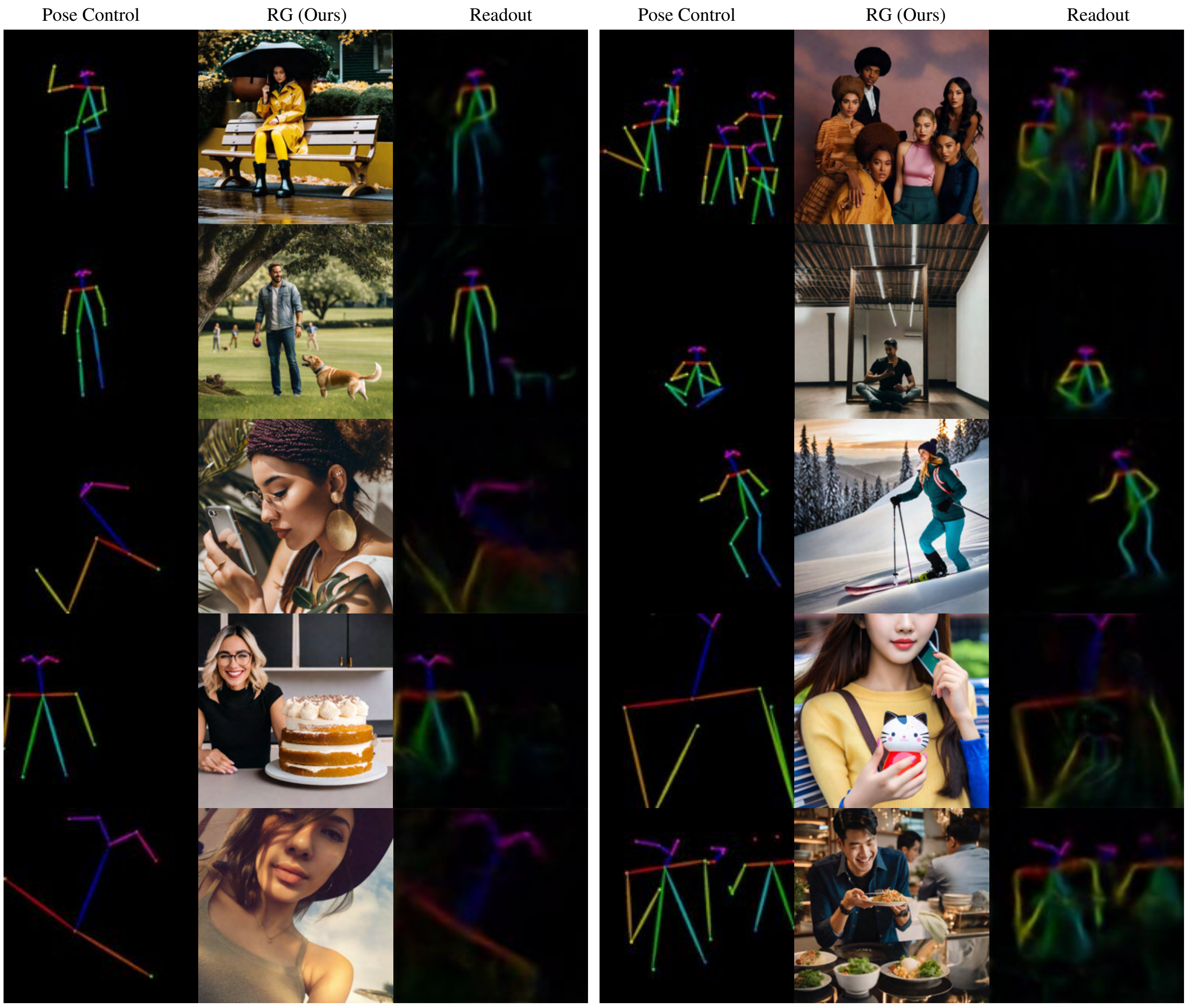

Figure 17. **Pose Control**: Additional examples with our pose head.

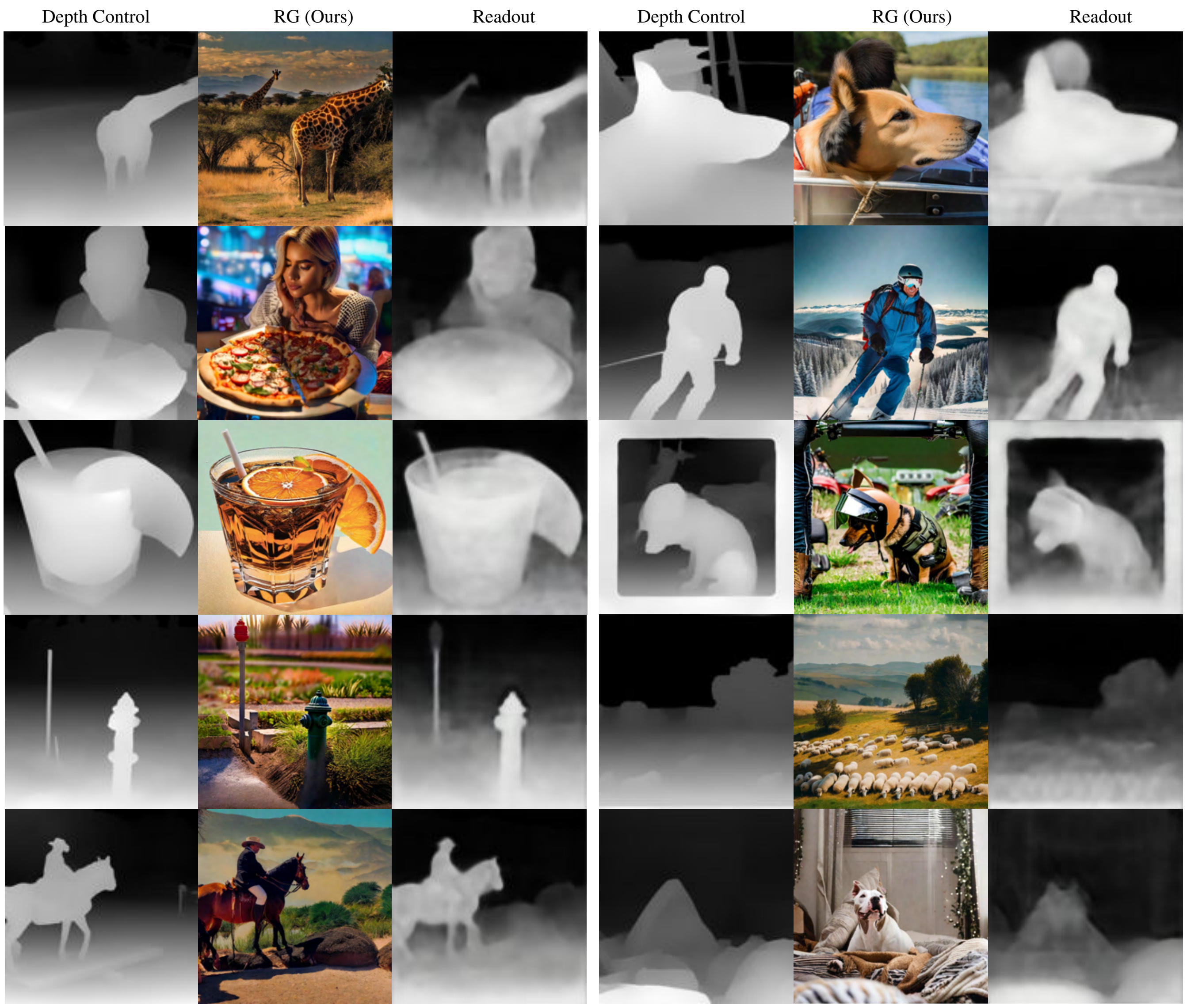

Figure 18. **Depth Control**: Additional examples with our depth head.

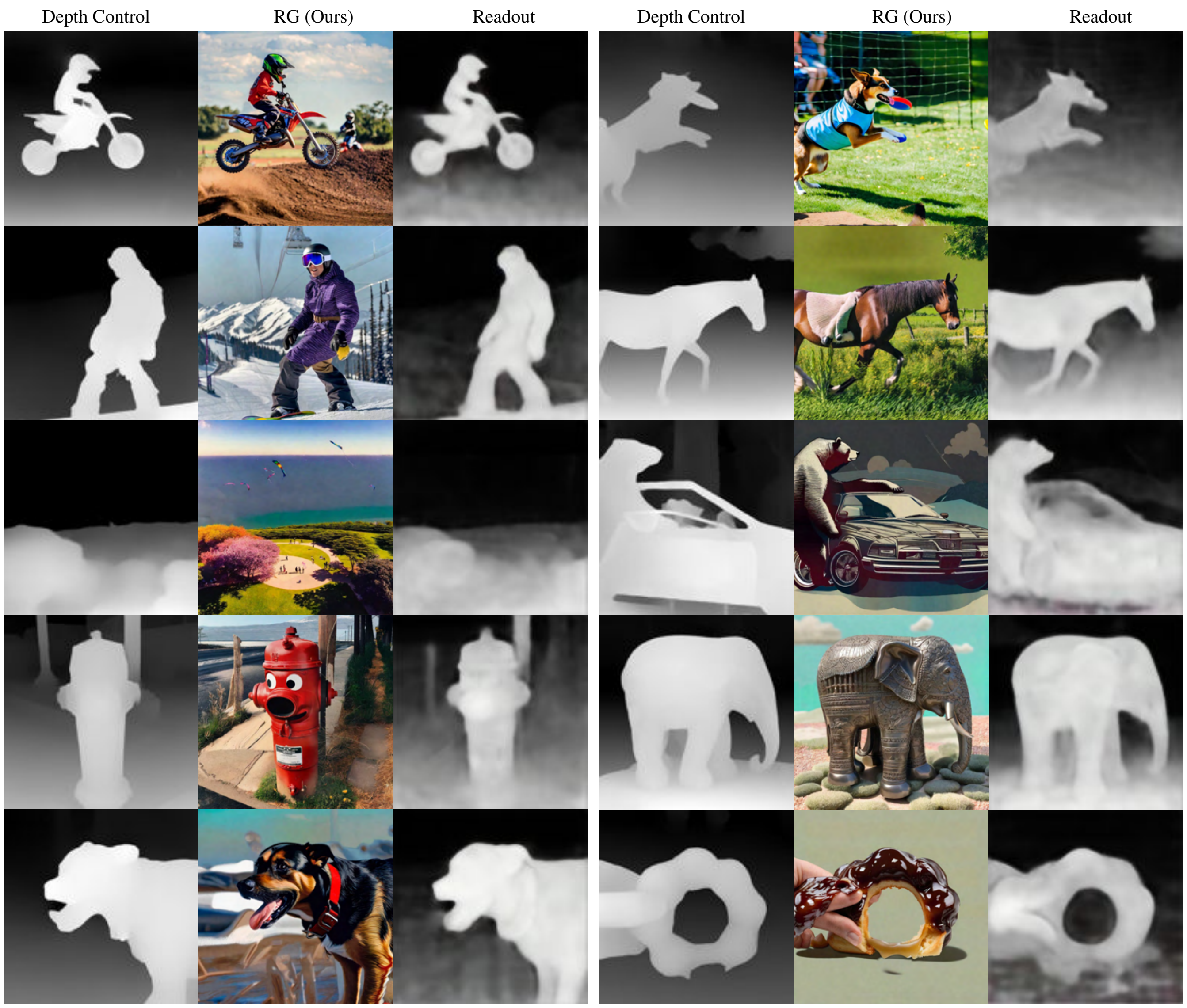

Figure 19. **Depth Control**: Additional examples with our depth head.

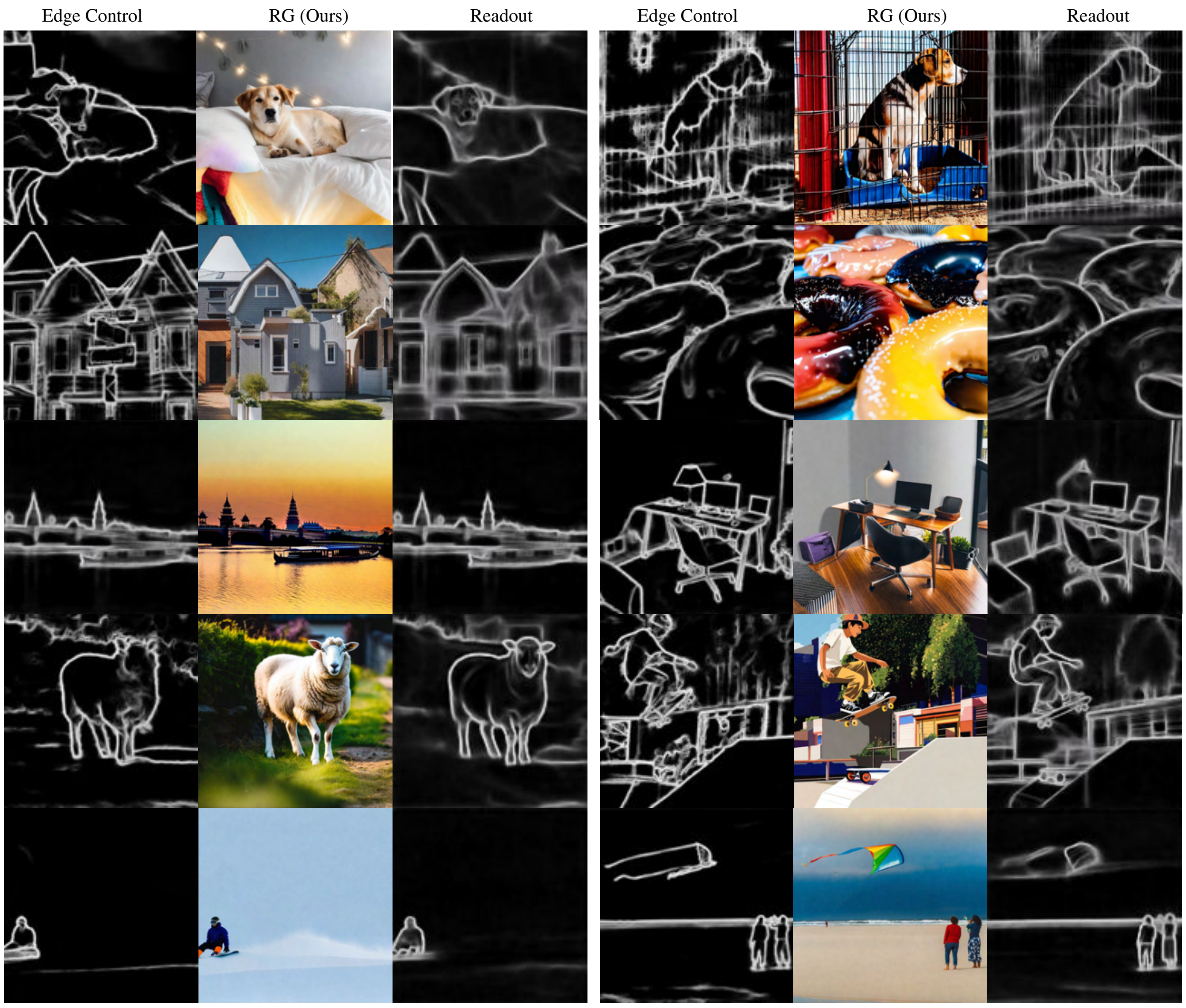

Figure 20. **Edge Control**: Additional examples with our edge head.

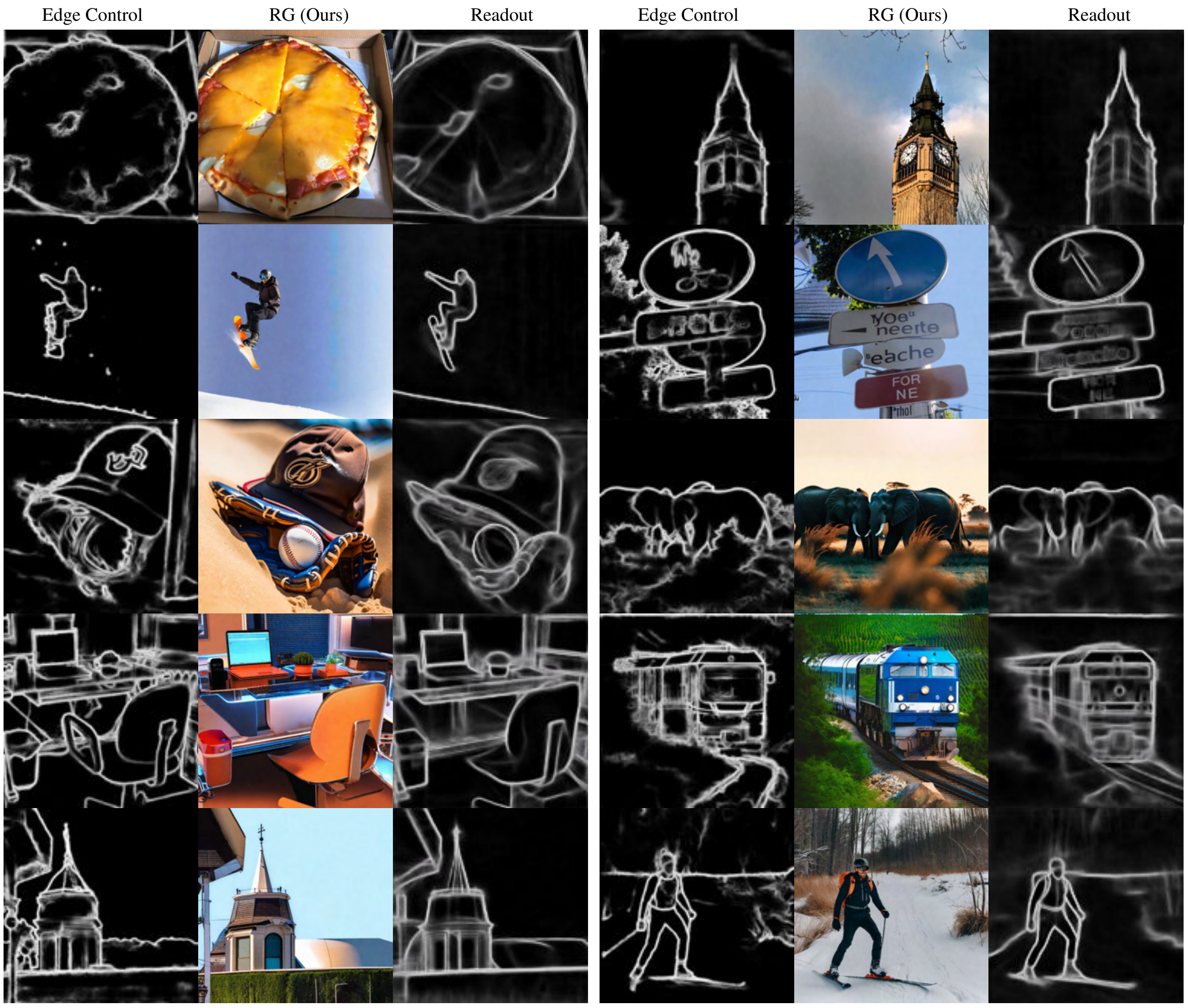

Figure 21. **Edge Control**: Additional examples with our edge head.

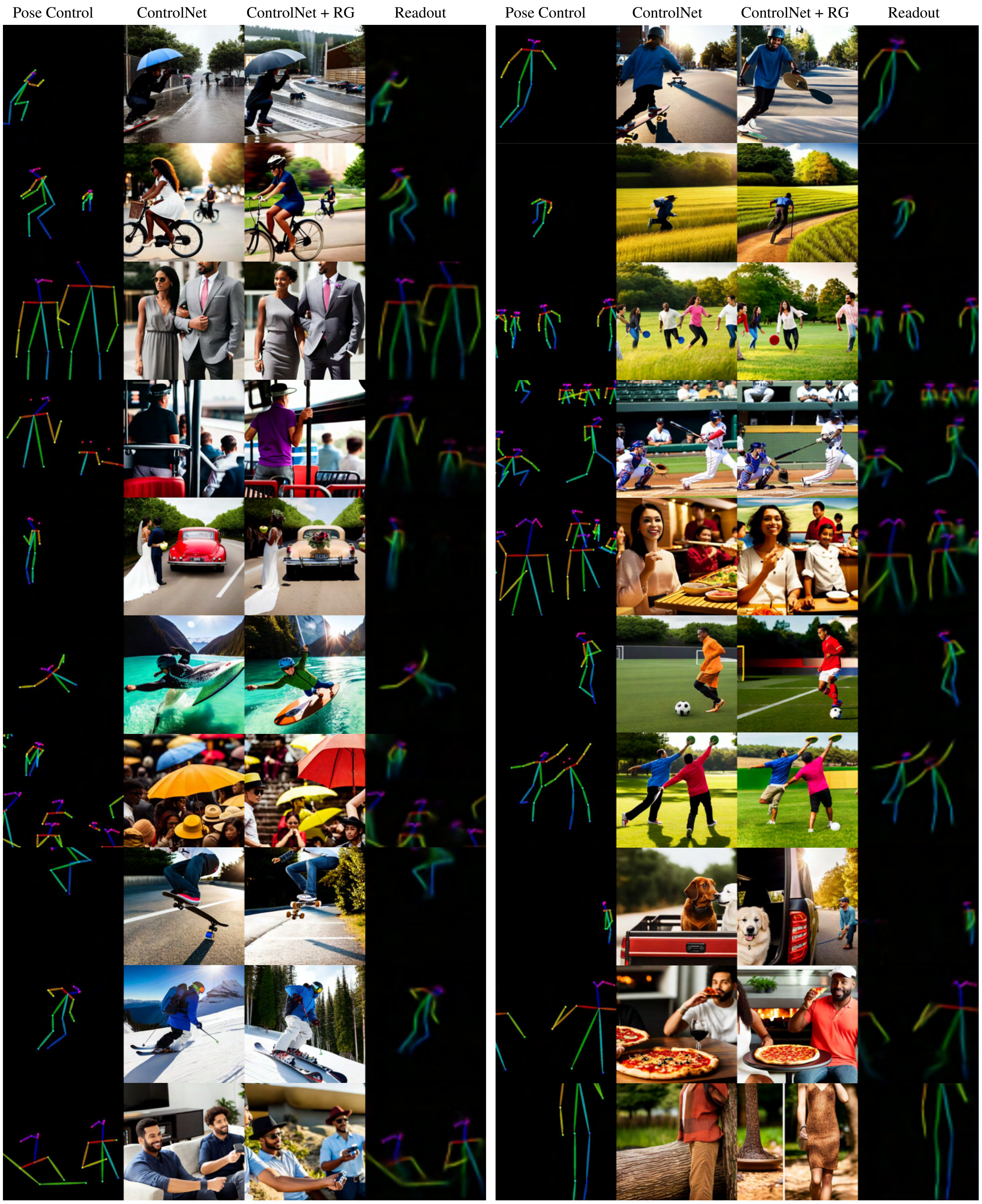

Figure 22. **Control Refinement**: Additional examples of Readout Guidance combined with ControlNet [70].

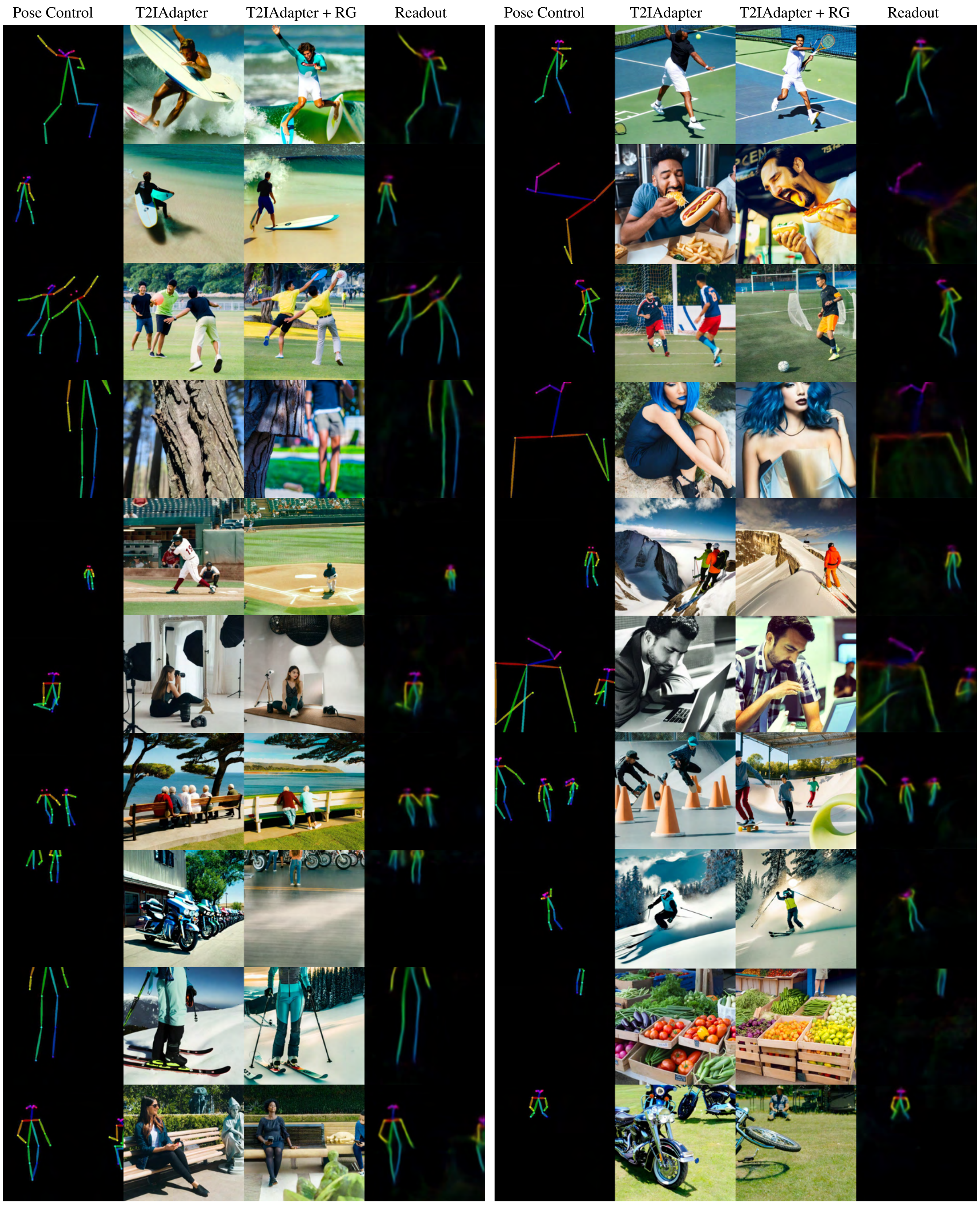

Figure 23. **Control Refinement**: Additional examples of Readout Guidance combined with T2IAdapter [40].

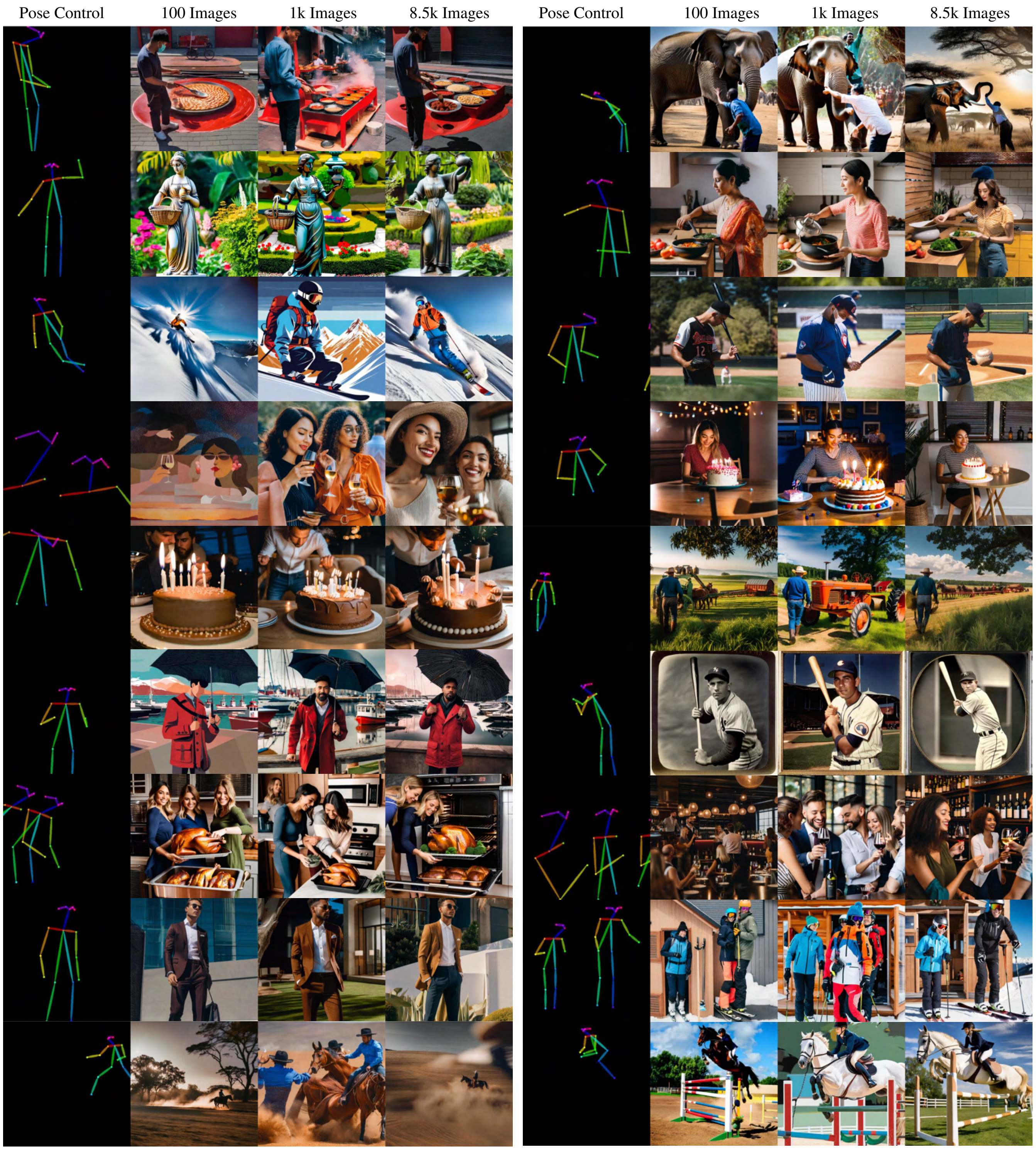

Figure 24. **Limited Data**: Additional examples after training our pose head on varying numbers of images.