LaVin-DiT: Large Vision Diffusion Transformer

Supplementary Material

A. More Technical Details of LaVin-DiT

A.1. Details of 3D RoPE

Recall that we represent task context and query as a unified sequence of frames, which is a 3D representation. Afterward, we extend RoPE from 1D to 3D format to capture the essential structure of visual data. Specifically, each token in an input sequence is associated with a 3D coordinate (t, x, y), representing its position in temporal and spatial dimensions. The 3D RoPE encodes positional information by decomposing it into three separate 1D RoPEs along the temporal and spatial axes, allowing the model to capture relative positional relationships across all dimensions inherently.

Technically, for each axis $a \in \{t, x, y\}$, we define a rotation matrix $R_p^{(a)}$ that operates on a dedicated subspace of an embedding vector z. The embedding vector is partitioned accordingly: $z = [z^{(t)},\ z^{(x)},\ z^{(y)}]$, where each subvector $z^{(a)} \in \mathbb{R}^{d_a}$ corresponds to axis a and $d = d_t + d_x + d_y$. The rotation matrix $R_p^{(a)}$ is constructed in a block-wise manner, rotating each pair of dimensions $(2i, 2i+1)$ by an angle $\theta_{p,i}^{(a)} = p^{(a)} \cdot \omega_i^{(a)}$, where $\omega_i^{(a)} = \omega_{\text{base}}^{-2i/d_a}$ and ω_{base} is a predefined constant:

$$R_p^{(a)} = \begin{bmatrix} R_p^{(a,0)} & & \\ & \ddots & \\ & & R_p^{(a,d_a/2-1)} \end{bmatrix}, \quad \text{where} \tag{1}$$

$$R_p^{(a,i)} = \begin{bmatrix} \cos\left(\theta_{p,i}^{(a)}\right) & -\sin\left(\theta_{p,i}^{(a)}\right) \\ \sin\left(\theta_{p,i}^{(a)}\right) & \cos\left(\theta_{p,i}^{(a)}\right) \end{bmatrix}. \tag{2}$$

When computing self-attention, the rotated query q and key k are obtained by applying the rotation matrices: $q'^{(a)} = R_p^{(a)} q^{(a)}$ and $k'^{(a)} = R_p^{(a)} k^{(a)}$. The full rotated query and key are then $q' = [q'^{(t)},\ q'^{(x)},\ q'^{(y)}]$ and $k' = [k'^{(t)},\ k'^{(x)},\ k'^{(y)}]$. When computing the attention between tokens at positions j and k, the dot product incorporates the rotations from all axes:

$$(q_j'^{\top}) k_k' = \sum_{a \in \{t,x,y\}} \left(q^{(a)}\right)^{\top} R_j^{(a)\top} R_k^{(a)} k^{(a)}. \tag{3}$$

The key property of rotation matrices is that the product of two rotation matrices corresponds to a rotation by the difference of their angles:

$$R_j^{(a)\top} R_k^{(a)} = R_{j-k}^{(a)}, \tag{4}$$

where $R_{p-q}^{(a)}$ is the rotation matrix for the relative position $j^{(a)} - k^{(a)}$, constructed as:

$$R_{j-k}^{(a)} = \begin{bmatrix} R_{j-k}^{(a,0)} & & \\ & \ddots & \\ & & R_{j-k}^{(a,N_a-1)} \end{bmatrix}, \quad \text{where} \tag{5}$$

$$R_{j-k}^{(a,i)} = \begin{bmatrix} \cos\left(\Delta_{jk}^{(a)} \omega_i^{(a)}\right) & -\sin\left(\Delta_{jk}^{(a)} \omega_i^{(a)}\right) \\ \sin\left(\Delta_{jk}^{(a)} \omega_i^{(a)}\right) & \cos\left(\Delta_{jk}^{(a)} \omega_i^{(a)}\right) \end{bmatrix}, \tag{6}$$

$$\Delta_{jk}^{(a)} = j^{(a)} - k^{(a)}. \tag{7}$$

This block-wise matrix format explicitly shows that the attention score depends on the relative positions $j^{(a)} - k^{(a)}$ along each axis a.

A.2. Algorithm Flows of LaVin-DiT

In this section, we present algorithm flows of the proposed LaVin-DiT. It is built upon the flow matching framework [8]. The training and inference procedures are provided in Algorithm 1 and Algorithm 2, respectively.

Algorithm 1 LaVin-DiT Training Procedure

Require: ST-VAE encoder $\text{Enc}(\cdot)$, dataset $\mathcal{D} = \{\boldsymbol{x}_i\}_{i=1}^K$, initialized parameters $\boldsymbol{\theta}$ of vector field $v_{\boldsymbol{\theta}}(\boldsymbol{z}, t)$, total iterations T, learning rate η.

for $n = 1$ to T **do**
  Sample $\boldsymbol{x} \sim \mathcal{D},\ \boldsymbol{c} \sim \mathcal{D}$
  Compute latents: $\boldsymbol{z}_0 \leftarrow \text{Enc}(\boldsymbol{x}),\ \boldsymbol{z}_c \leftarrow \text{Enc}(\boldsymbol{c})$
  Initialize random latent: $\boldsymbol{z}_1 \sim \mathcal{N}(0, 1)$
  Sample time step: $t \sim \text{LogitNormal}(0, 1)$
  Interpolate: $\boldsymbol{z}_t \leftarrow (1-t)\boldsymbol{z}_1 + t\boldsymbol{z}_0$
  Target vector: $\boldsymbol{u} \leftarrow \boldsymbol{z}_0 - \boldsymbol{z}_1$
  Predicted vector: $\boldsymbol{v} \leftarrow v_{\boldsymbol{\theta}}(\boldsymbol{z}_t,\ \boldsymbol{z}_c,\ t)$
  Compute loss: $\mathcal{L} \leftarrow \mathbb{E}[|\boldsymbol{v} - \boldsymbol{u}|_2^2]$
  Update parameters: $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta \nabla_{\boldsymbol{\theta}} \mathcal{L}$
end for

Training procedure. As illustrated in Algorithm 1, the primary goal is to learn a vector field $v_{\boldsymbol{\theta}}(\boldsymbol{z}, t)$ that maps the latent space dynamics conditioned on the target latent $\boldsymbol{z}_0$, the task context latent $\boldsymbol{z}_c$, and a time step t. The training process iteratively refines the parameters $\boldsymbol{\theta}$ to minimize the discrepancy between the predicted and ground-truth latent trajectories.

Inference procedure. This process, described in Algorithm 2, employs the learned vector field $v_{\boldsymbol{\theta}}$ to sample in the

latent space. Starting with an initial latent $\boldsymbol{z}_1 \sim \mathcal{N}(0, 1)$, the method denoises iteratively using the Euler method.

Algorithm 2 LaVin-DiT Inference Procedure

Require: Trained vector field $v_{\boldsymbol{\theta}}(z, t)$, ST-VAE encoder $\mathrm{Enc}(\cdot)$, ST-VAE decoder $\mathrm{Dec}(\cdot)$, timesteps N, dataset $\mathcal{D} = \{\boldsymbol{x}_i\}_{i=1}^{K}$.

Set step size $\Delta t \leftarrow \frac{1}{N}$, initialize $t^{(N)} \leftarrow 1$
Sample initial latent: $\boldsymbol{z}_1 \sim \mathcal{N}(0, 1)$
Encode condition: $\boldsymbol{z}_c \leftarrow \mathrm{Enc}(\boldsymbol{c}),\ \boldsymbol{c} \sim \mathcal{D}$
for $k = N$ down to 1 **do**
    Update time: $t^{(k-1)} \leftarrow t^{(k)} - \Delta t$
    Compute vector field: $\boldsymbol{v}^{(k)} \leftarrow v_{\boldsymbol{\theta}}(\boldsymbol{z}^{(k)},\ \boldsymbol{z}_c,\ t^{(k)})$
    Update latent: $\boldsymbol{z}^{(k-1)} \leftarrow \boldsymbol{z}^{(k)} - \Delta t \cdot \boldsymbol{v}^{(k)}$
end for
Decode sample: $\hat{\boldsymbol{y}} \leftarrow \mathrm{Dec}(\boldsymbol{z}_0)$

Table 1. Configurations of LaVin-DiT with different numbers of parameters.

	LaVin-DiT		
	0.1B	**1.0B**	**3.4B**
Latent channels	16	16	16
Patch size	2×2	2×2	2×2
Hidden channels	512	1024	2304
Num. layers	12	28	22
Num. heads	8	16	32
K.V. groups	-	-	4
Drop path	0.0	0.1	0.1
Uncond. ratio	0.1	0.1	0.1
Grad. clip	1.0	1.0	1.0
EMA moment.	0.9999	0.9999	0.9999
Extra norm.	-	S-Norm.	S-Norm.
Position embed.	3D-RoPE	3D-RoPE	3D-RoPE

B. Supplementary Experimental Settings

B.1. Large-Scale Multi-Task Dataset Composition

Recall that we build a large-scale multi-task dataset to unify diverse computer vision tasks. We integrate multiple public image-level and video-level task benchmarks into a large-scale dataset for training. Details are listed in Table 2.

B.2. Evaluation Metrics

In this work, we provide quantitative results for 10 tasks (The others are presented with visualization results). Here we introduce the evaluation metrics for these 10 tasks.

Colorization. We randomly sample 1,000 images from ImageNet-1K validation set [4] and convert them into grayscale. We adopt LPIPS [14] and mean squared error (MSE) as metrics.

Inpainting. We randomly sample 1,000 images from ImageNet-1K validation set [4] and mask out a 128×128 region for each image. We adopt the LPIPS [14] and Frechet Inception Distance (FID) as metrics.

Depth Estimation. We evaluate our model on NYUv2 test set [12], including 654 images. Following the protocol of affine-invariant depth evaluation [9], we first align the prediction to the ground truth with the least squares fitting. Afterwards, we adopt Absolute Mean Relative Error (AbsRel) and Mean Squared Error (MSE) as metrics.

Surface Normal Estimation. We evaluate our model on NYUv2 test set [12]. Following the protocol used in [1], we calculate the angular error between the prediction and the ground-truth normal maps and use the mean angular error as the metric.

Depth-to-Image Generation. We adopt all samples in the NYUv2 dataset [12], including 1,449 images. Given the pseudo label generated via Depth-anything V2 or Stable-Normal (turbo), we generate the corresponding RGB image and use the LPIPS [14] and Frechet Inception Distance (FID) as metrics.

Normal-to-Image Generation. The metrics are the same those in Depth-to-Image Generation.

Single Object Detection. We evaluate the model on the Pascal-5i dataset [10] and adopt the mean intersection-over-union (mIoU) as the metric.

Foreground Segmentation. We evaluate our model on the Pascal-5i dataset [10], including 4 different test splits. Following the protocol in [2], we extract binary masks from our predictions and report the mIoU.

Deraining. We randomly sample 1,000 images from ImageNet-1K validation set [4] and apply the raining filter on them. We adopt the Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index Measure (SSIM) as metrics.

De-motion Blur. We randomly sample 1,000 images from the ImageNet-1K validation set [4] and apply motion blur on these images. We adopt the PSNR and SSIM as metrics.

B.3. Architecture Details of LaVin-DiT

Here we detail the architecture of the LaVin-DiT models. Table 1 outlines the configurations for three parameter scales: 0.1B, 1.0B, and 3.4B. Each configuration is characterized by key architectural hyperparameters, including the number of latent channels, patch size, hidden channels, and the number of layers. Additionally, the configurations specify the number of attention heads, key-value groups, drop path rates, and unconditional ratios. To further enhance model training, we incorporate advanced techniques such as gradient clipping and the Exponential Moving Average (EMA). All models utilize 3D-RoPE to ensure consistent spatial and temporal encoding across scales. For large mod-

Table 2. Summary of the large-scale multi-task dataset used in LaVin-DiT, including the number of examples and annotation types for each component dataset. Tasks range from visual understanding and generation.

Task	Dataset	Number of Samples	Annotation Type
Single Object Detection	COCO 2017 train [7]	117,266	Ground Truth
	Object365 train [11]	1,728,778	Ground Truth
Instance Segmentation	COCO 2017 train [7]	117,266	Ground Truth
	ADE20K train+val [15]	19,020	Ground Truth
	Cityscapes train+val [3]	3,457	Ground Truth
Panoptic Segmentation	COCO 2017 train [7]	117,266	Ground Truth
	ADE20K train+val [15]	19,020	Ground Truth
	Cityscapes train+val [3]	3,457	Ground Truth
Pose Estimation	COCO 2017 train [7]	64,115	Ground Truth
Pose-to-Image Generation	COCO 2017 train [7]	64,115	Ground Truth
Depth Estimation	ImageNet1K train [4]	1,281,167	Depth-anything V2
Depth-to-Image Generation	ImageNet1K train [4]	1,281,167	Depth-anything V2
Surface Normal Estimation	COCO 2017 train [7]	117,266	Stable-Normal (turbo)
	ADE20K train+val [15]	19,020	Stable-Normal (turbo)
	Cityscapes train+val [3]	3,457	Stable-Normal (turbo)
Normal-to-Image Generation	COCO 2017 train [7]	117,266	Stable-Normal (turbo)
	ADE20K train+val [15]	19,020	Stable-Normal (turbo)
	Cityscapes train+val [3]	3,457	Stable-Normal (turbo)
Edge Detection	ImageNet1K [4] train	1,281,167	Canny (OpenCV)
	COCO 2017 train [7]	117,266	Canny (OpenCV)
Inpainting	ImageNet1K train [4]	1,281,167	Crop (OpenCV)
	COCO 2017 train [7]	117,266	Crop (OpenCV)
Colorization	ImageNet1K train [4]	1,281,167	Grayscale (OpenCV)
	COCO 2017 train [7]	117,266	Grayscale (OpenCV)
De-glass Blur	ImageNet1K train [4]	1,281,167	Albumentations
	COCO 2017 train [7]	117,266	Albumentations
De-motion Blur	ImageNet1K train [4]	1,281,167	Albumentations
	COCO 2017 train [7]	117,266	Albumentations
De-raining	ImageNet1K train [4]	1,281,167	Albumentations
	COCO 2017 train [7]	117,266	Albumentations
Frame Prediction	UCF101 train [13]	7,629	N/A
	Kinetic 700 train+val [6]	570,465	N/A
	Kubric train [5]	48,689	N/A
Video Depth Estimation	Kubric train [5]	48,689	Ground Truth
Depth-to-Video Generation	Kubric train [5]	48,689	Ground Truth
Video Surface Normal Estimation	Kubric train [5]	48,689	Ground Truth
Normal-to-Video Generation	Kubric train [5]	48,689	Ground Truth
Video Optical Flow Estimation	Kubric train [5]	48,689	Ground Truth
Video Instance Segmentation	Kubric train [5]	48,689	Ground Truth

els, we employ sandwich normalization to improve training stability.

C. Supplementary Qualitative Results

We show more visualization results for each task, including object detection (Figure 1), foreground segmentation (Figure 2), panoptic segmentation (Figure 3), pose estimation (Figure 4), pose-to-image generation (Figure 5), depth estimation (Figure 6), depth-to-image generation (Figure 7), surface normal estimation (Figure 8), normal-to-image generation (Figure 9), edge detection (Figure 10), inpainting (Figure 11), colorization (Figure 12), de-glass blur (Figure 13), de-motion blur (Figure 14), de-raining (Figure 15), frame prediction (Figure 16), video depth estimation (Figure 17), depth-to-video generation (Figure 18), video sur-

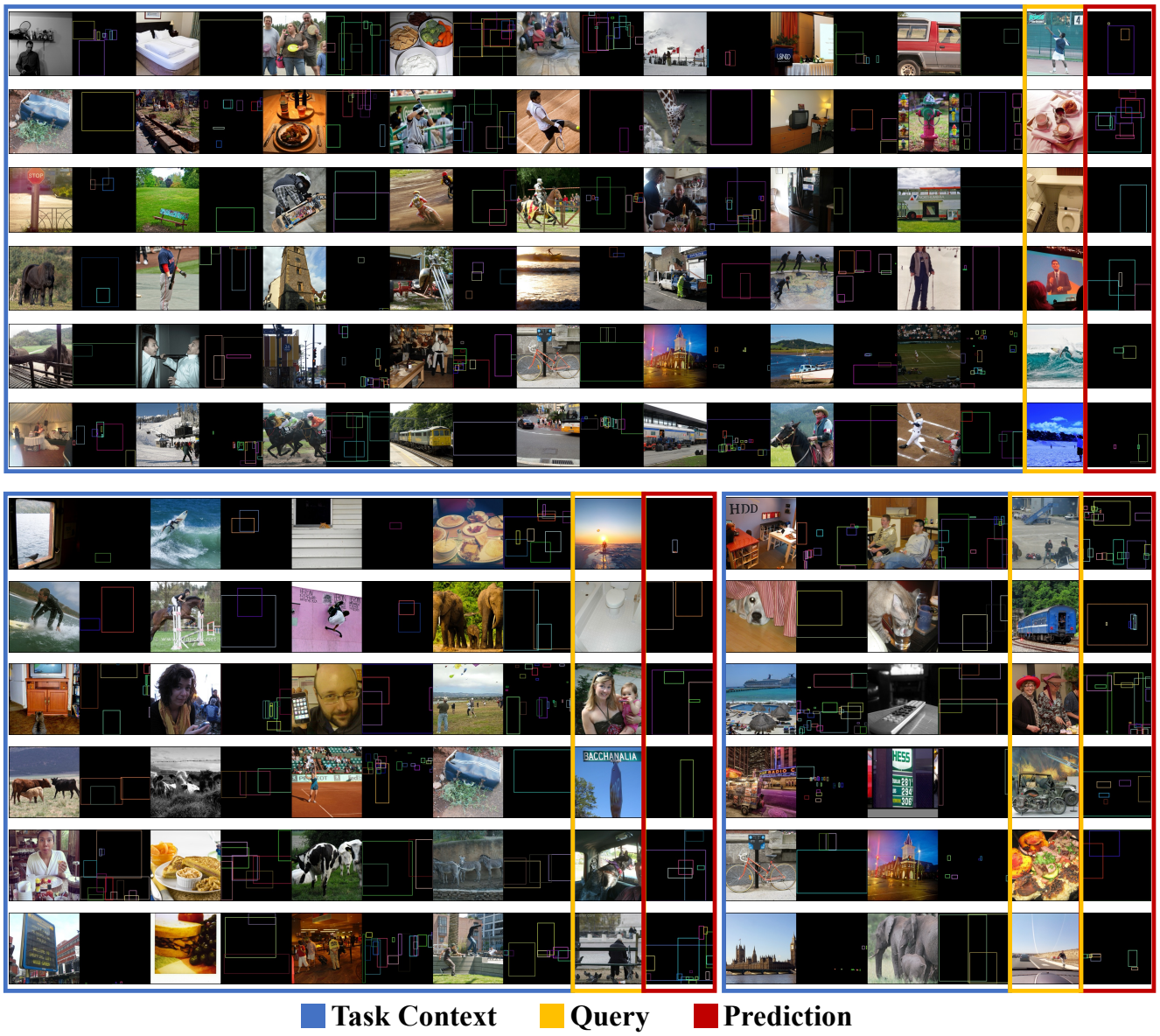

Figure 1. **Qualitative results on object detection.** Each row contains a sequence of images interleaved with annotations, followed by a query. The last image is predicted by the model (marked in red). *Best viewed in color.*

face normal estimation (Figure 19), normal-to-video generation (Figure 20), video optical flow estimation (Figure 21), and video instance segmentation (Figure 22).

D. Potential Applications

LaVin-DiT opens transformative possibilities for tackling open-world computer vision challenges by unifying diverse vision tasks within a single generative framework. For instance, it can seamlessly generalize across tasks such as text-to-image generation, text-to-video generation, video understanding, 3D reconstruction (Figure 23), and 2D/3D visual editing without supervised fine-tuning. By leveraging its spatial-temporal variational autoencoder and joint diffusion transformer, LaVin-DiT excels at capturing the complexity of high-dimensional visual data while maintaining task-specific alignment through in-context learning. This capability positions LaVin-DiT as a foundation model capable of addressing dynamic realistic vision problems, including autonomous driving perception, robotic scene understanding, and interactive AI systems in mixed-reality environments, significantly advancing the frontier of adaptable and scalable AI systems.

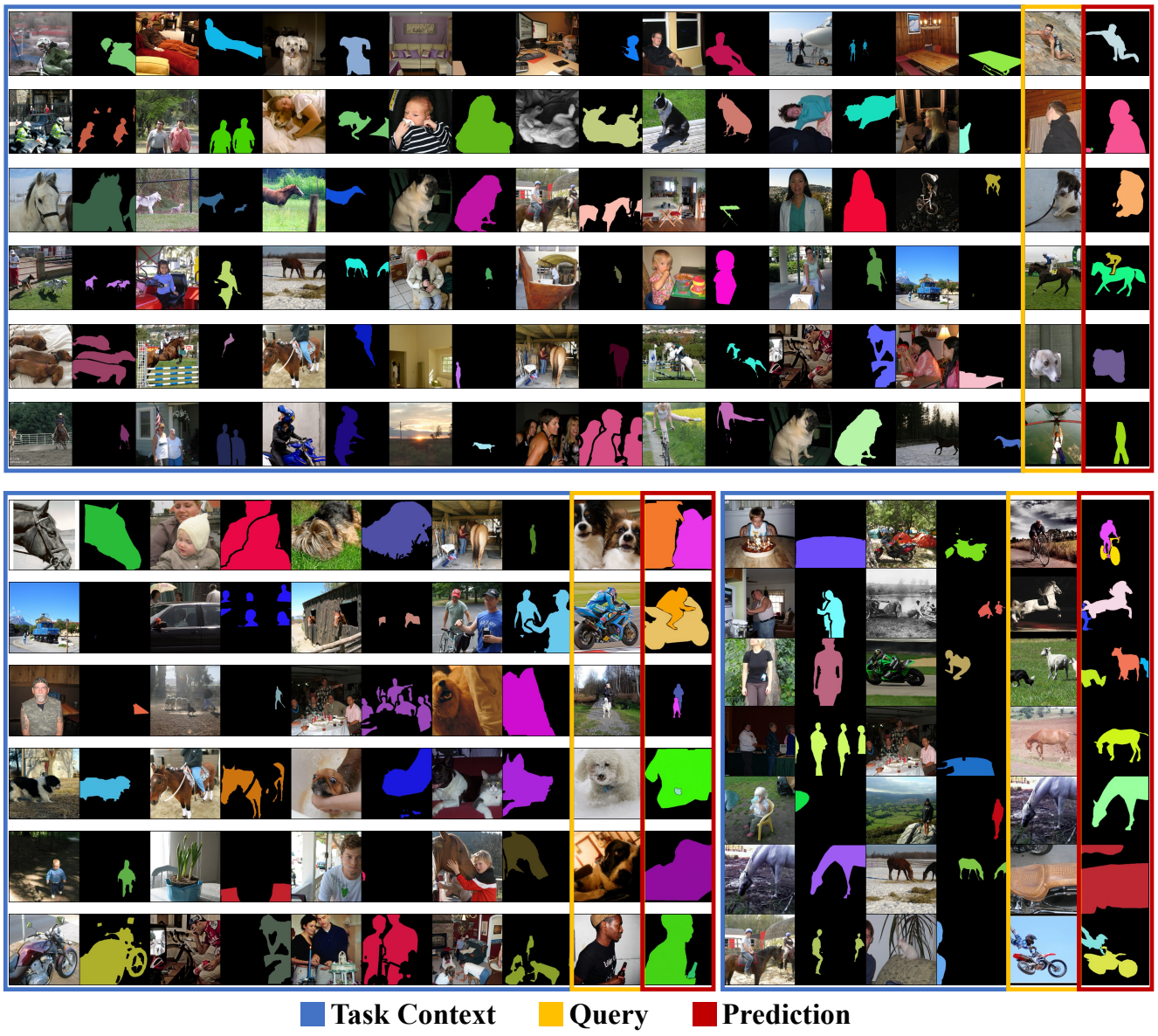

Figure 2. **Qualitative results on foreground segmentation.** Each row contains a sequence of images interleaved with annotations, followed by a query. The last image is predicted by the model (marked in red). *Best viewed in color.*

Figure 3. **Qualitative results on panoptic segmentation.** Each row contains a sequence of images interleaved with annotations, followed by a query. The last image is predicted by the model (marked in red). *Best viewed in color.*

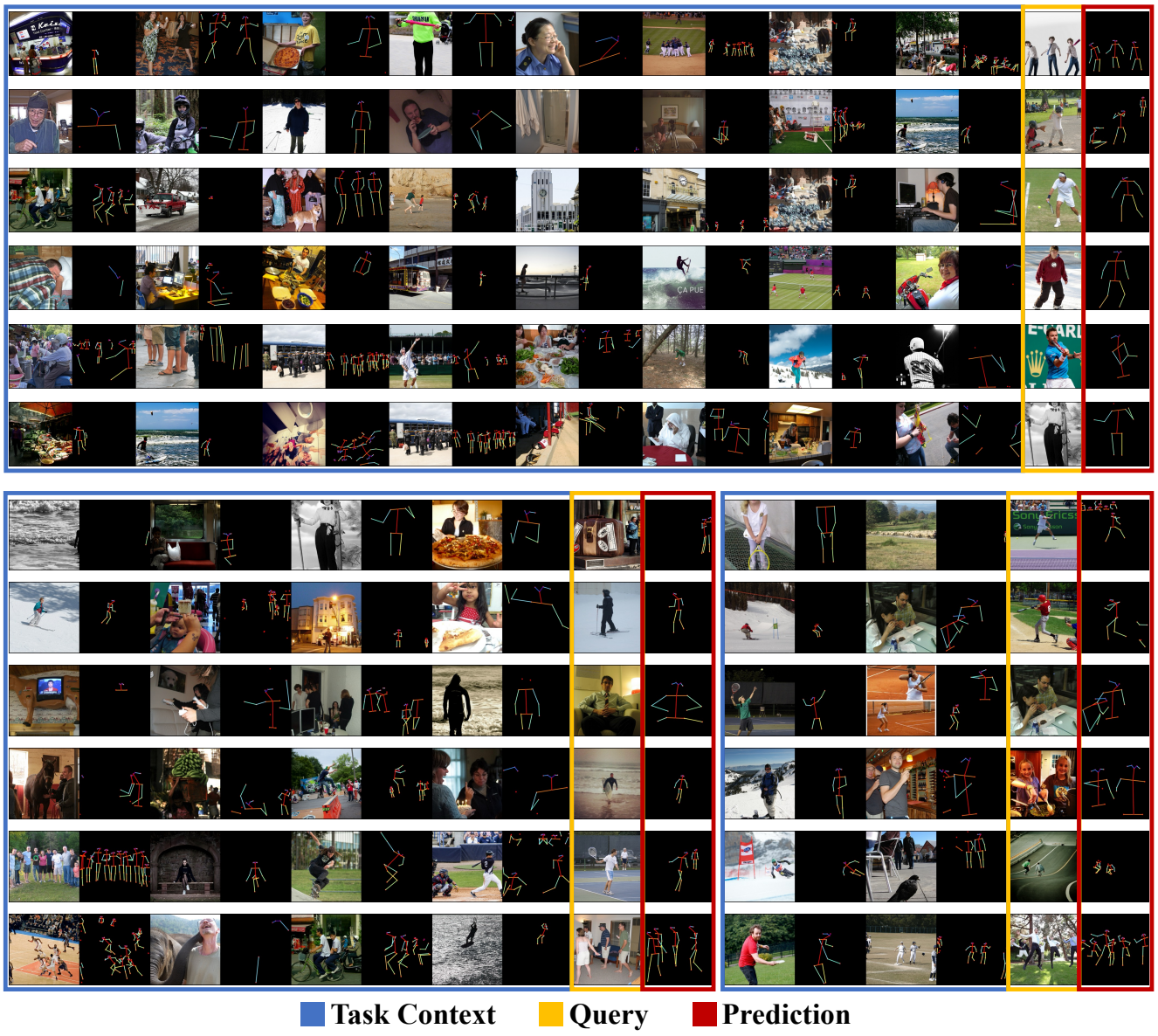

Figure 4. **Qualitative results on pose estimation.** Each row contains a sequence of images interleaved with annotations, followed by a query. The last image is predicted by the model (marked in red). *Best viewed in color.*

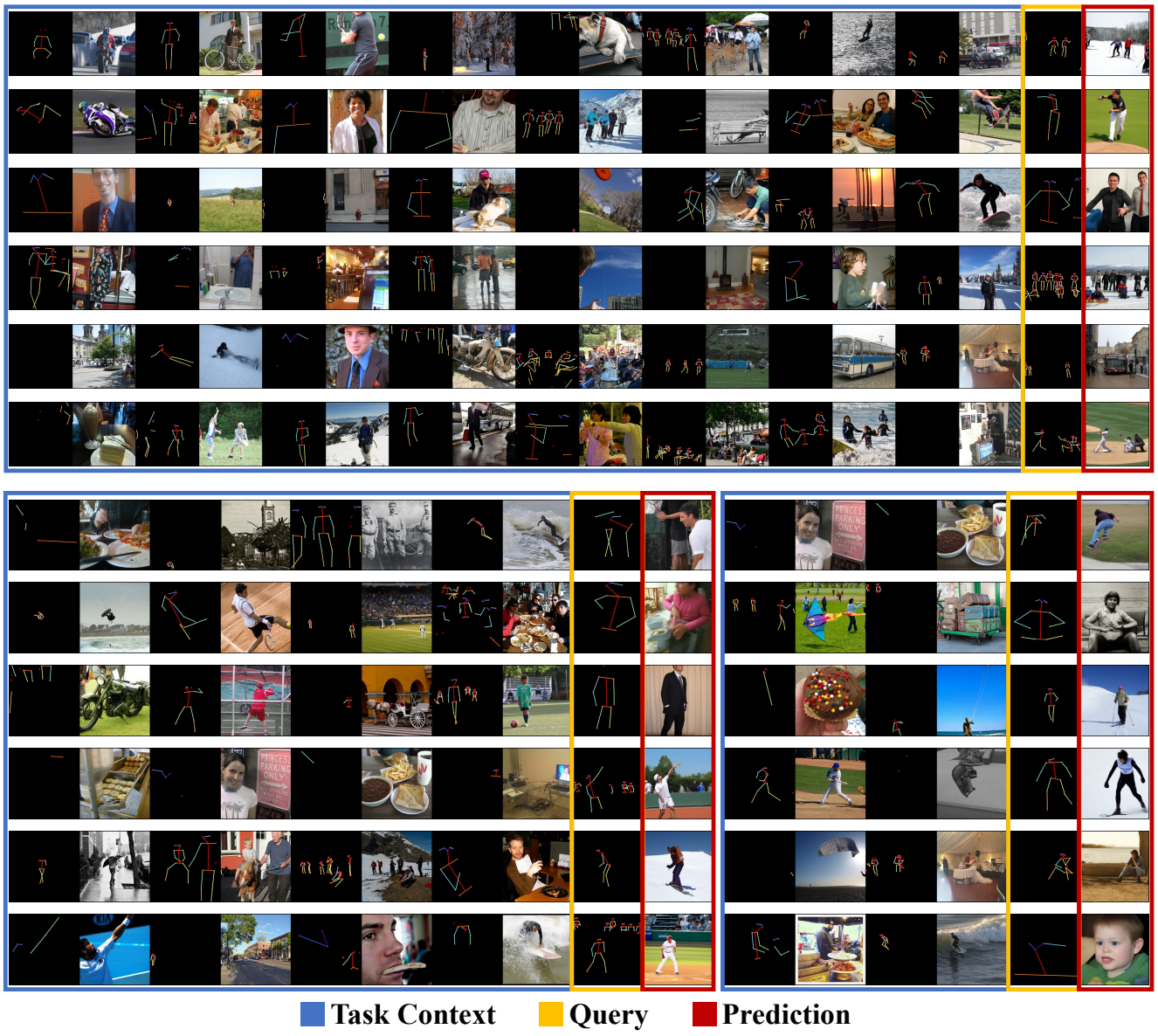

Figure 5. **Qualitative results on pose-to-image generation.** Each row contains a sequence of images interleaved with annotations, followed by a query. The last image is predicted by the model (marked in red). *Best viewed in color.*

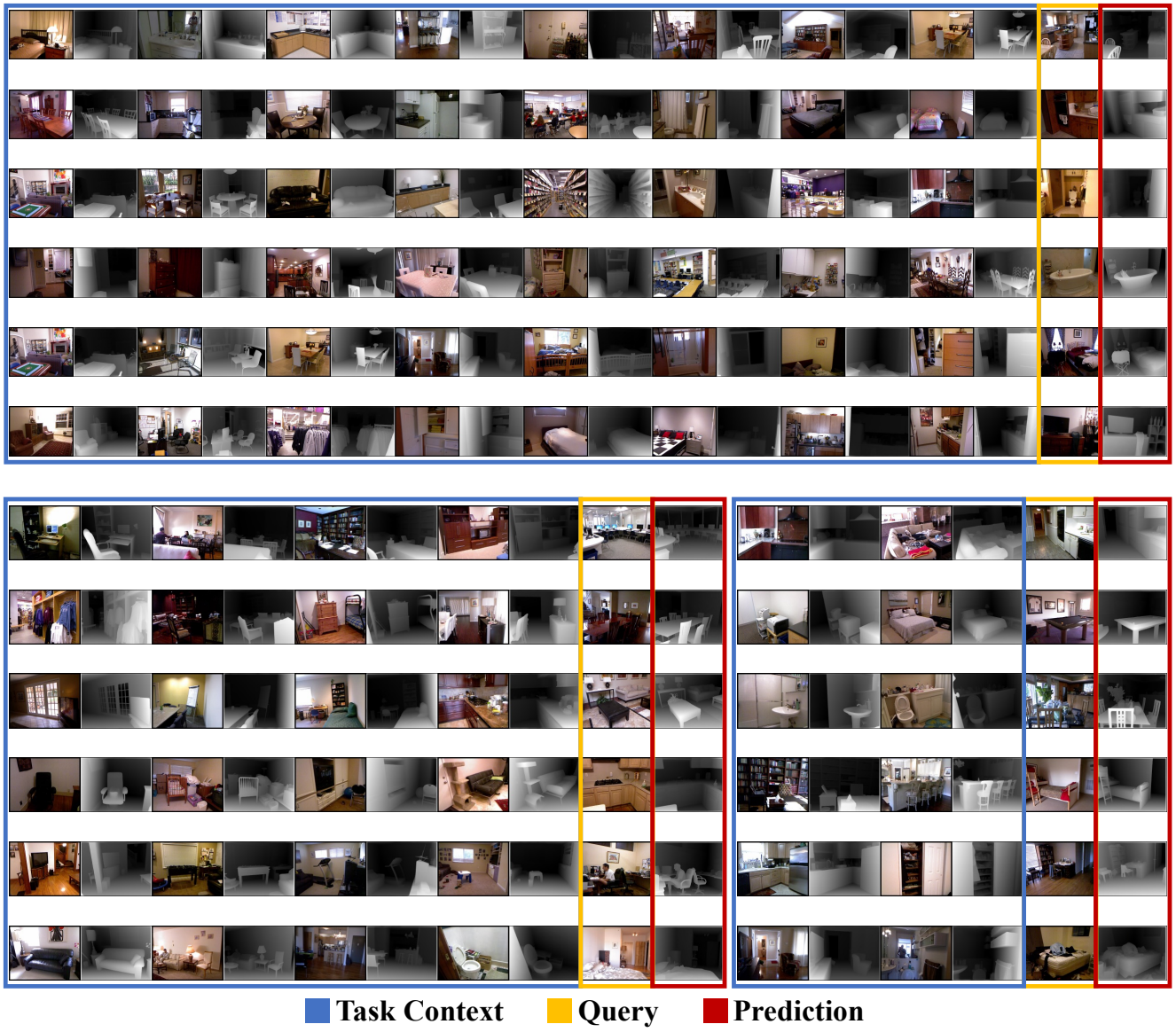

Figure 6. **Qualitative results on depth estimation.** Each row contains a sequence of images interleaved with annotations, followed by a query. The last image is predicted by the model (marked in red). *Best viewed in color.*

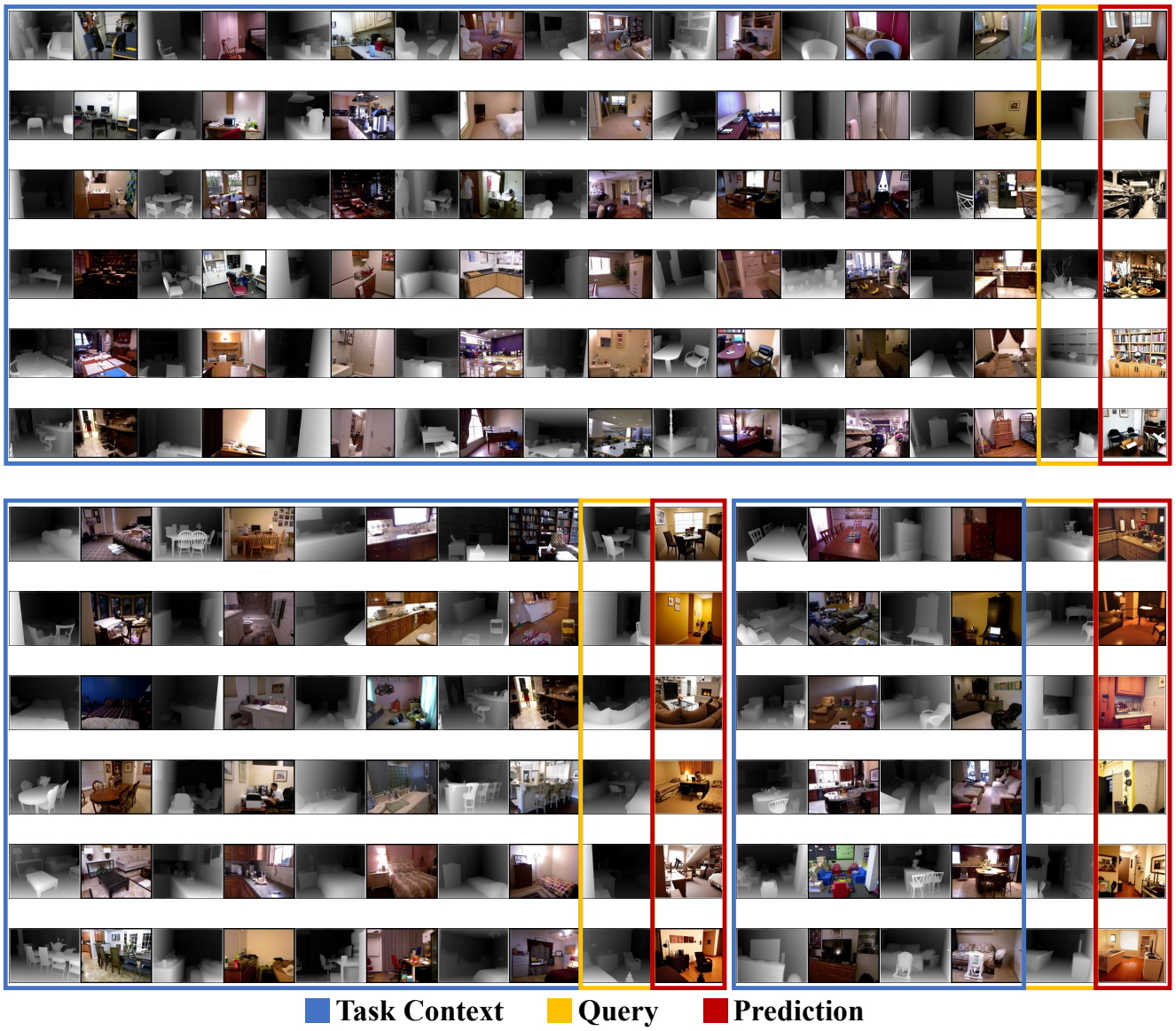

Figure 7. **Qualitative results on depth-to-image generation.** Each row contains a sequence of images interleaved with annotations, followed by a query. The last image is predicted by the model (marked in red). *Best viewed in color.*

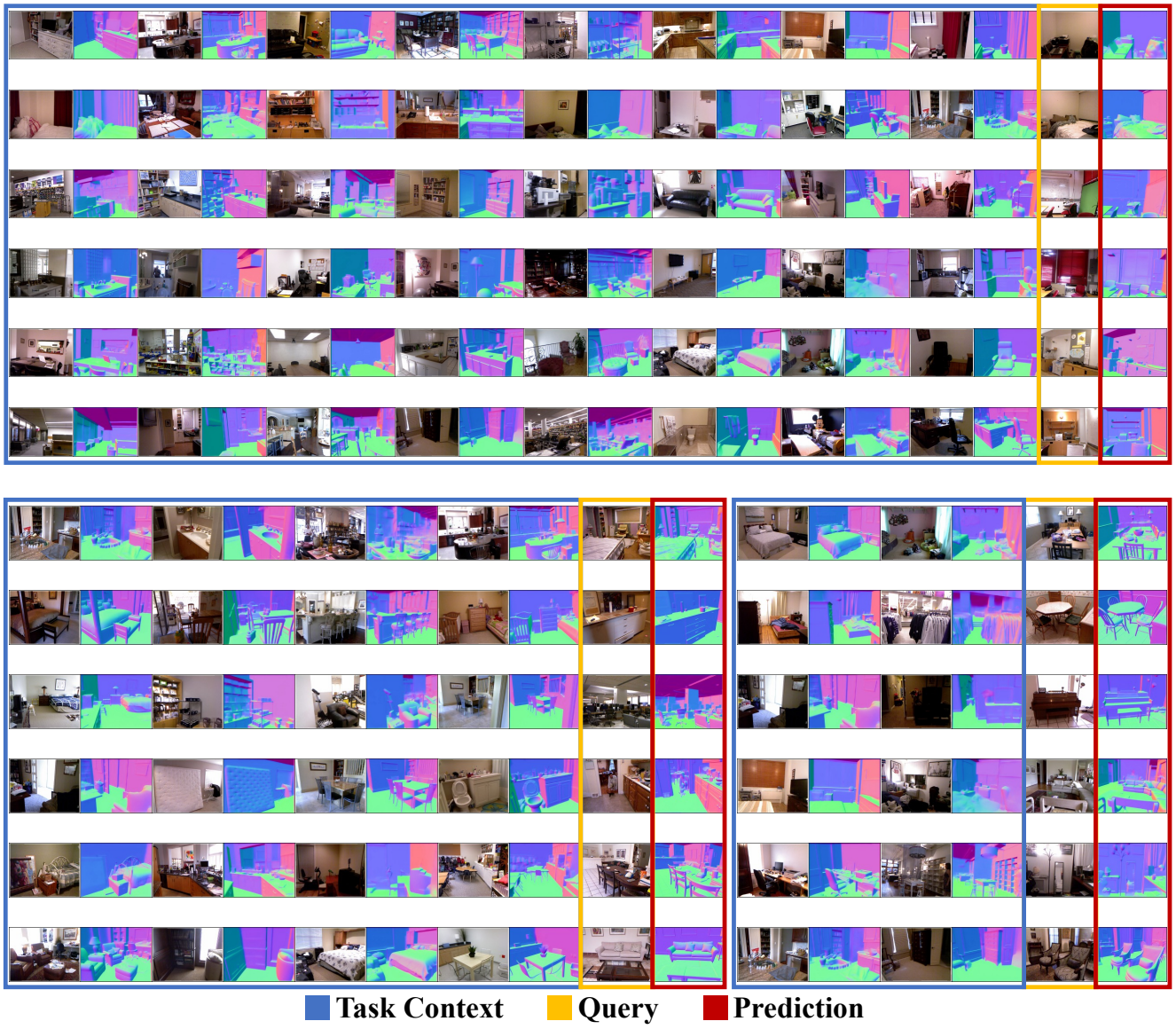

Figure 8. **Qualitative results on surface normal estimation.** Each row contains a sequence of images interleaved with annotations, followed by a query. The last image is predicted by the model (marked in red). *Best viewed in color.*

Figure 9. **Qualitative results on normal-to-image generation.** Each row contains a sequence of images interleaved with annotations, followed by a query. The last image is predicted by the model (marked in red). *Best viewed in color.*

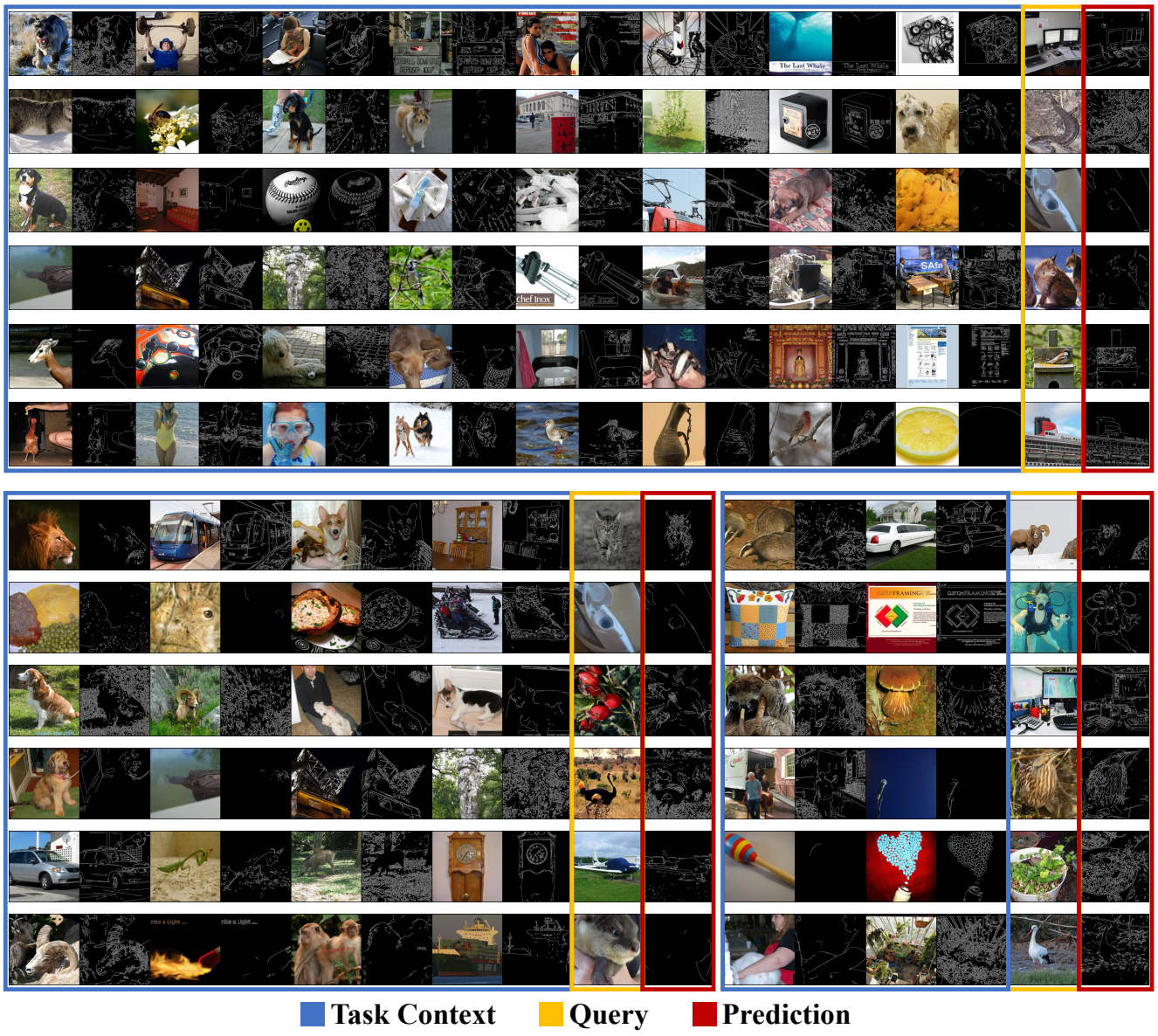

Figure 10. **Qualitative results on edge detection.** Each row contains a sequence of images interleaved with annotations, followed by a query. The last image is predicted by the model (marked in red). *Best viewed in color.*

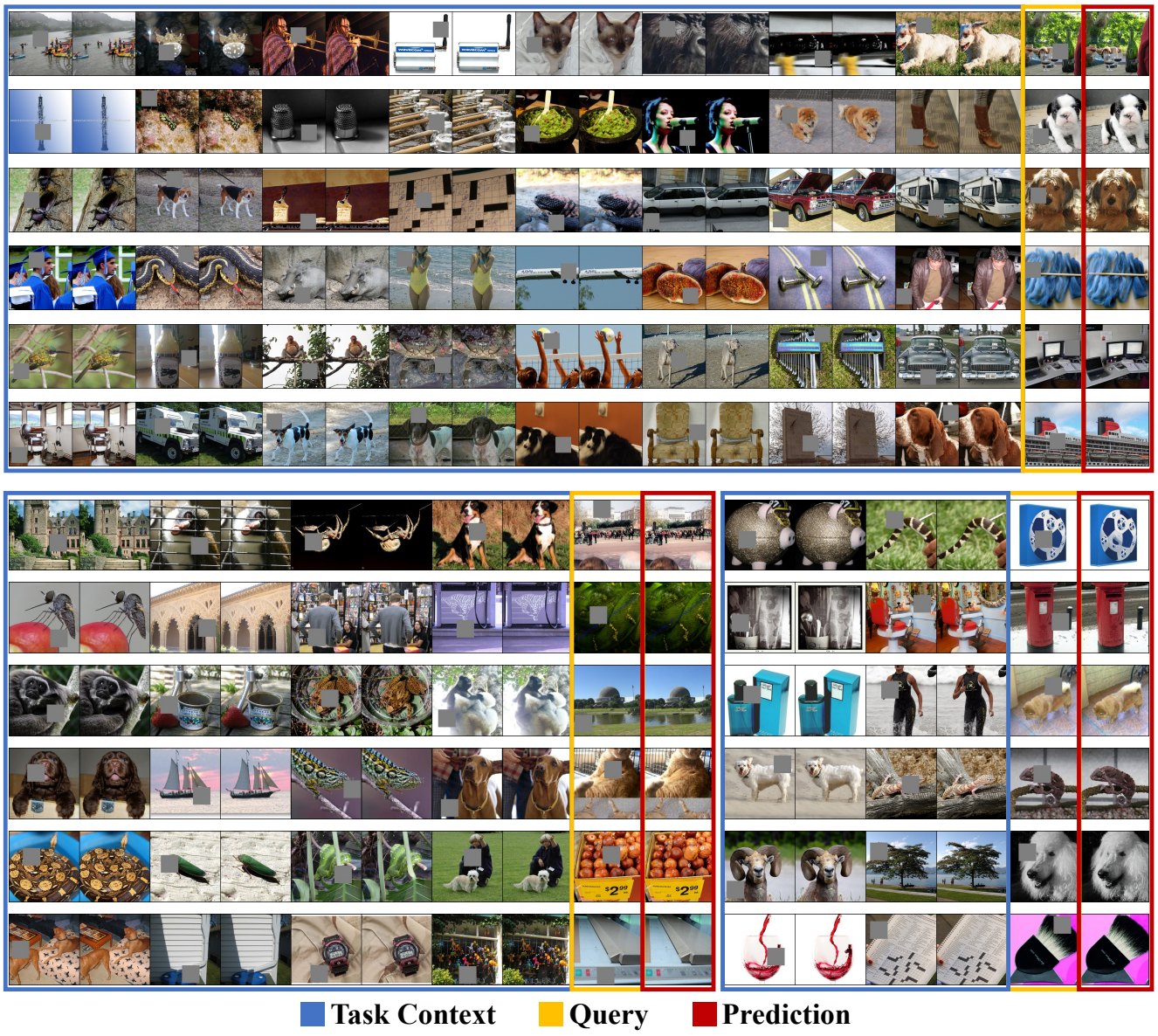

Figure 11. **Qualitative results on inpainting.** Each row contains a sequence of images interleaved with annotations, followed by a query. The last image is predicted by the model (marked in red). *Best viewed in color.*

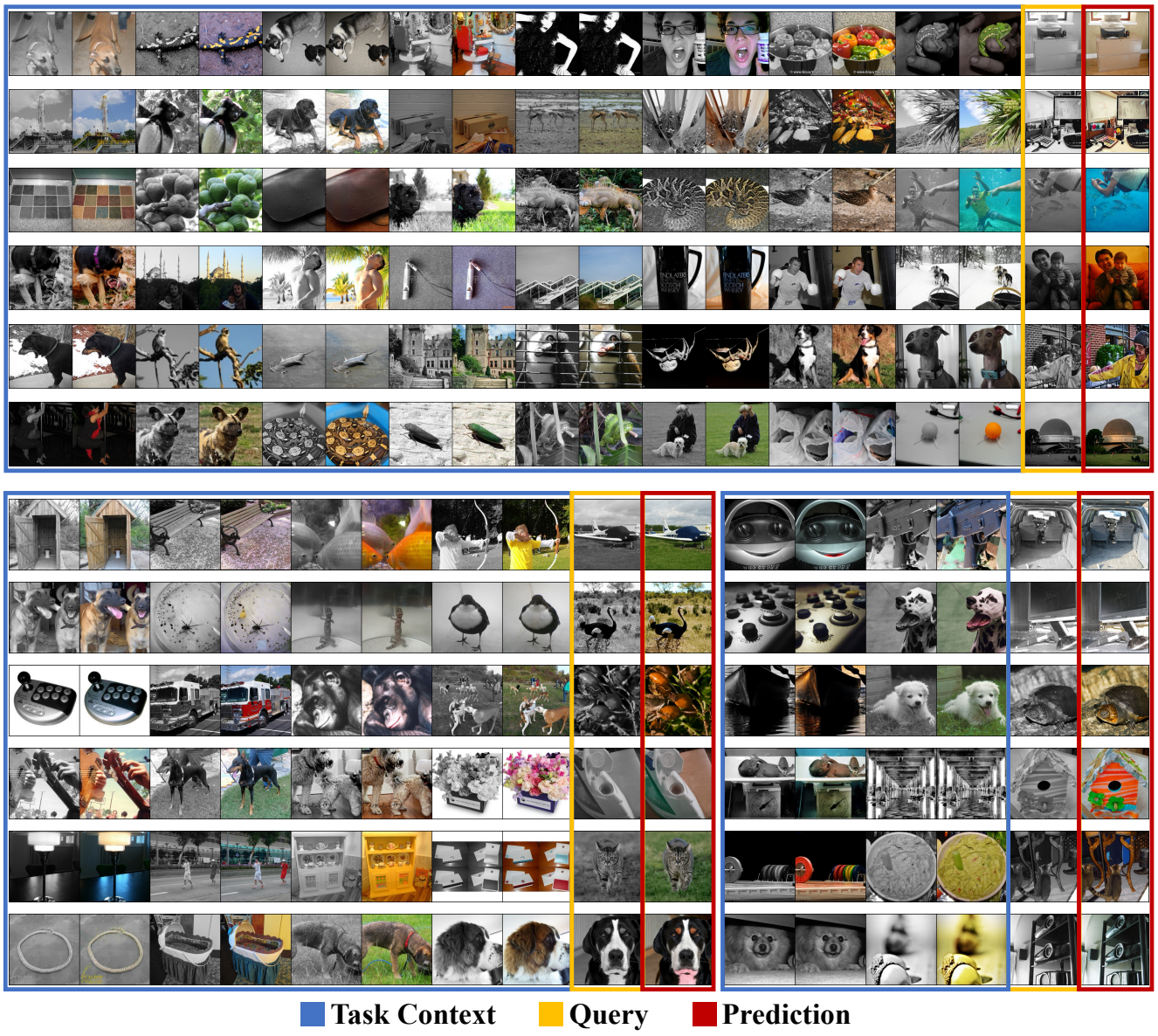

Figure 12. **Qualitative results on image colorization.** Each row contains a sequence of images interleaved with annotations, followed by a query. The last image is predicted by the model (marked in red). *Best viewed in color.*

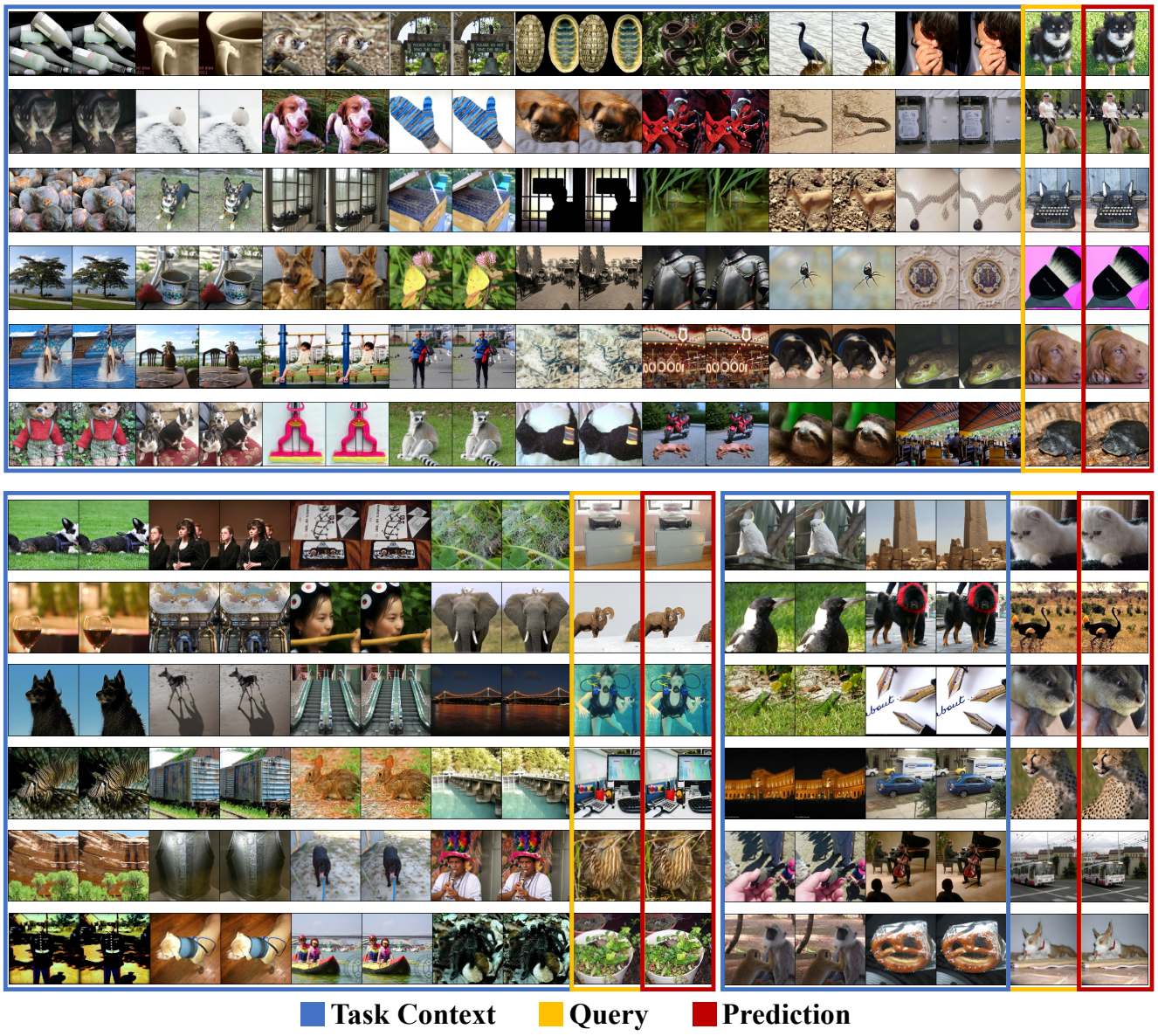

Figure 13. **Qualitative results on de-glass blur.** Each row contains a sequence of images interleaved with annotations, followed by a query. The last image is predicted by the model (marked in red). *Best viewed in color.*

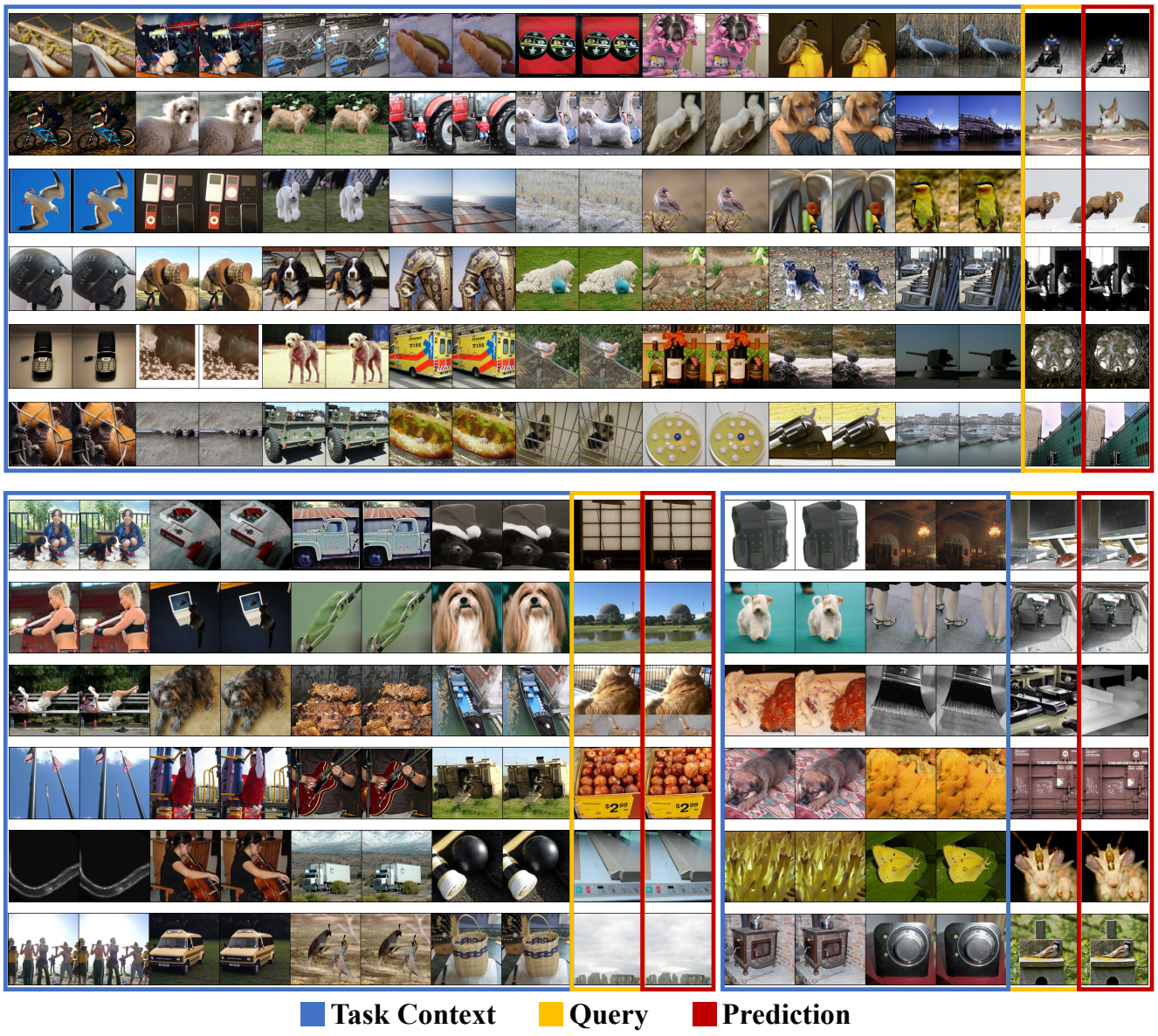

Figure 14. **Qualitative results on de-motion blur.** Each row contains a sequence of images interleaved with annotations, followed by a query. The last image is predicted by the model (marked in red). *Best viewed in color.*

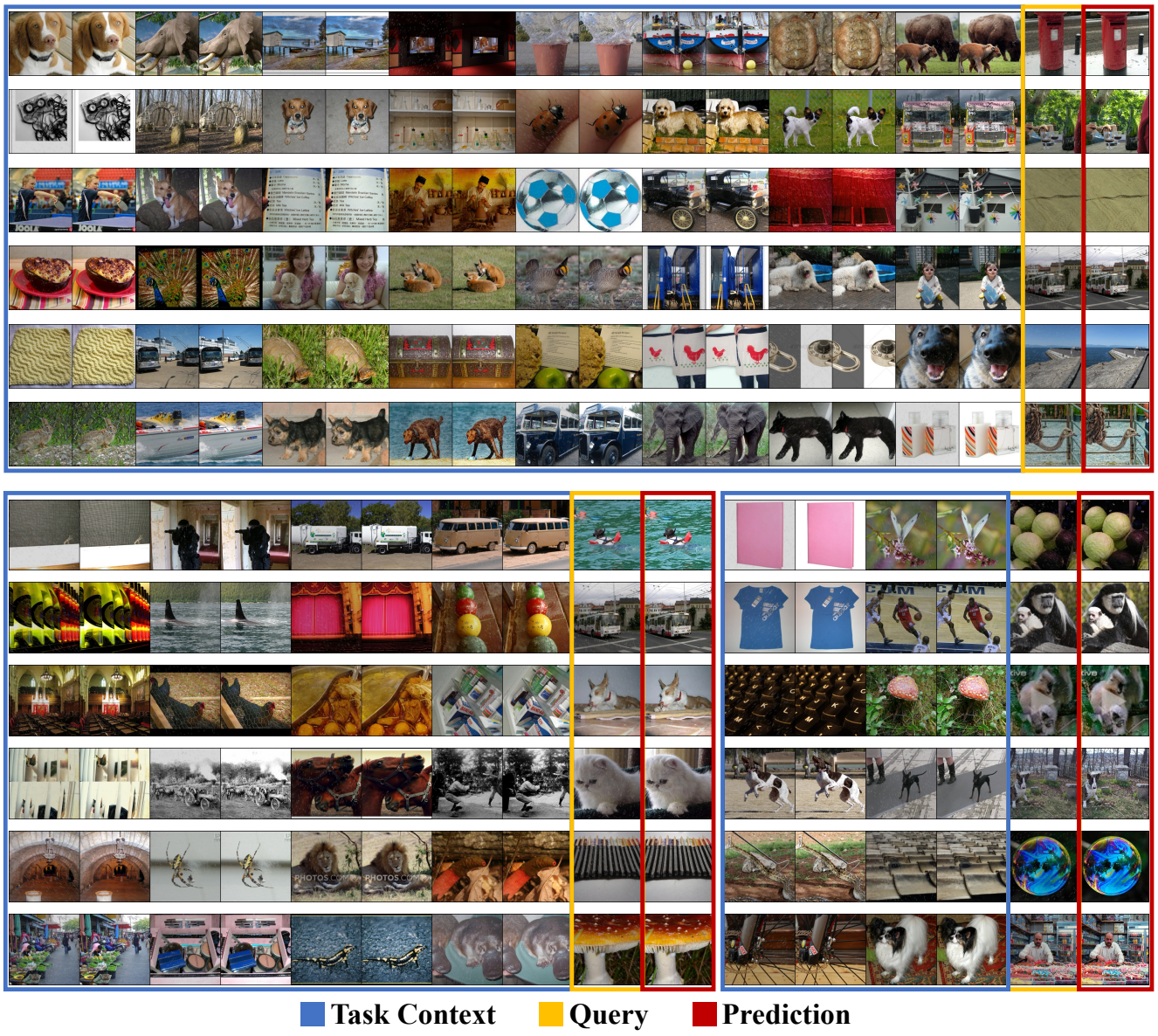

Figure 15. **Qualitative results on de-raining.** Each row contains a sequence of images interleaved with annotations, followed by a query. The last image is predicted by the model (marked in red). *Best viewed in color.*

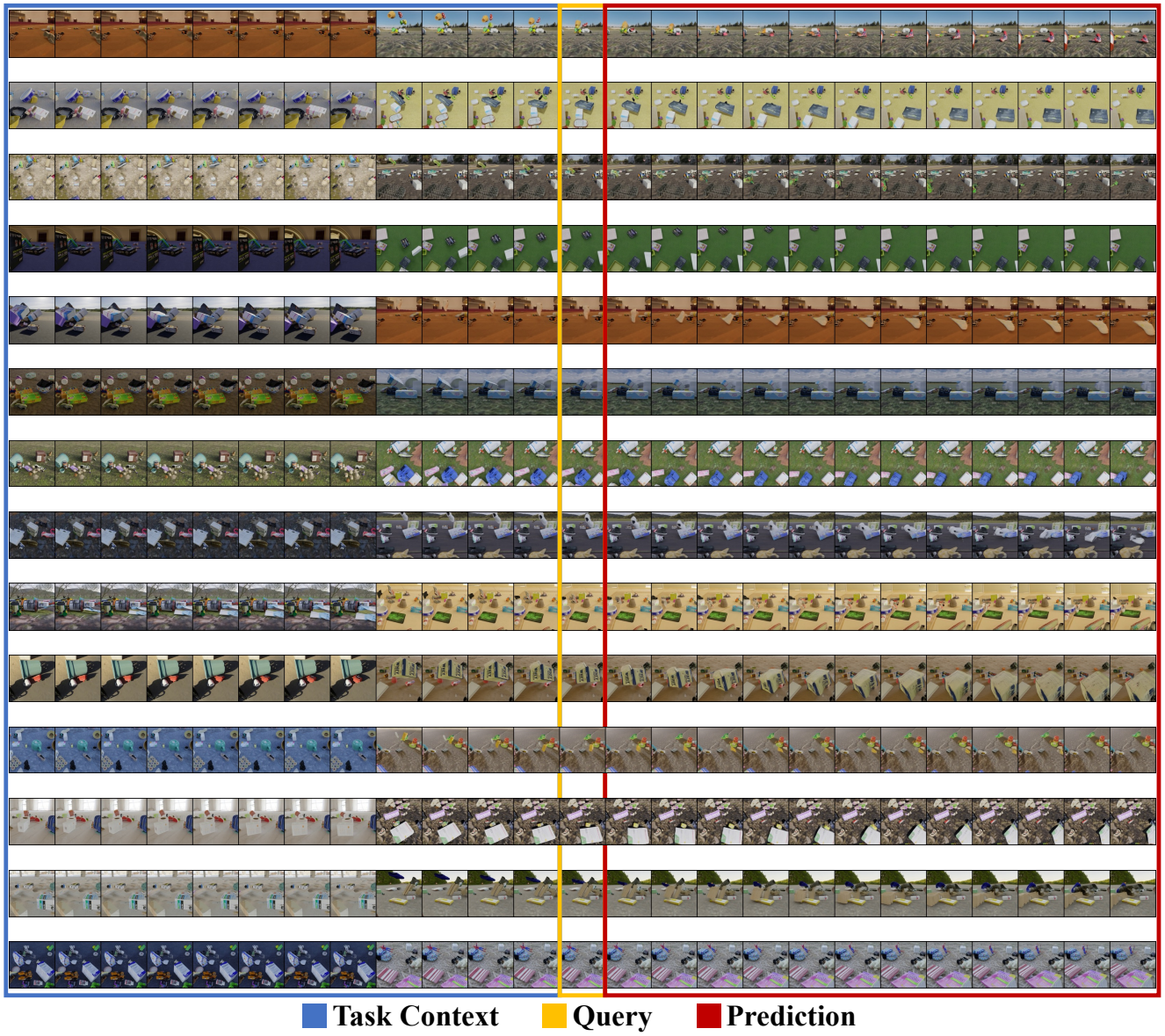

Figure 16. **Qualitative results on frame prediction.** Each row includes a video sequence with a series of target frames as task context (marked in blue), followed by a query frame (marked in yellow). A set of frames in the red box indicates the model's predictions. Due to the length of the sequence, a portion of the task context is hidden. *Best viewed in color.*

Figure 17. **Qualitative results on video depth estimation.** Each row includes a video sequence with a series of target frames as task context (marked in blue), followed by a query frame (marked in yellow). A set of frames in the red box indicates the model's predictions. Due to the length of the sequence, a portion of the task context is hidden. *Best viewed in color.*

Figure 18. **Qualitative results on depth-to-video generation.** Each row includes a video sequence with a series of target frames as task context (marked in blue), followed by a query frame (marked in yellow). A set of frames in the red box indicates the model's predictions. Due to the length of the sequence, a portion of the task context is hidden. *Best viewed in color.*

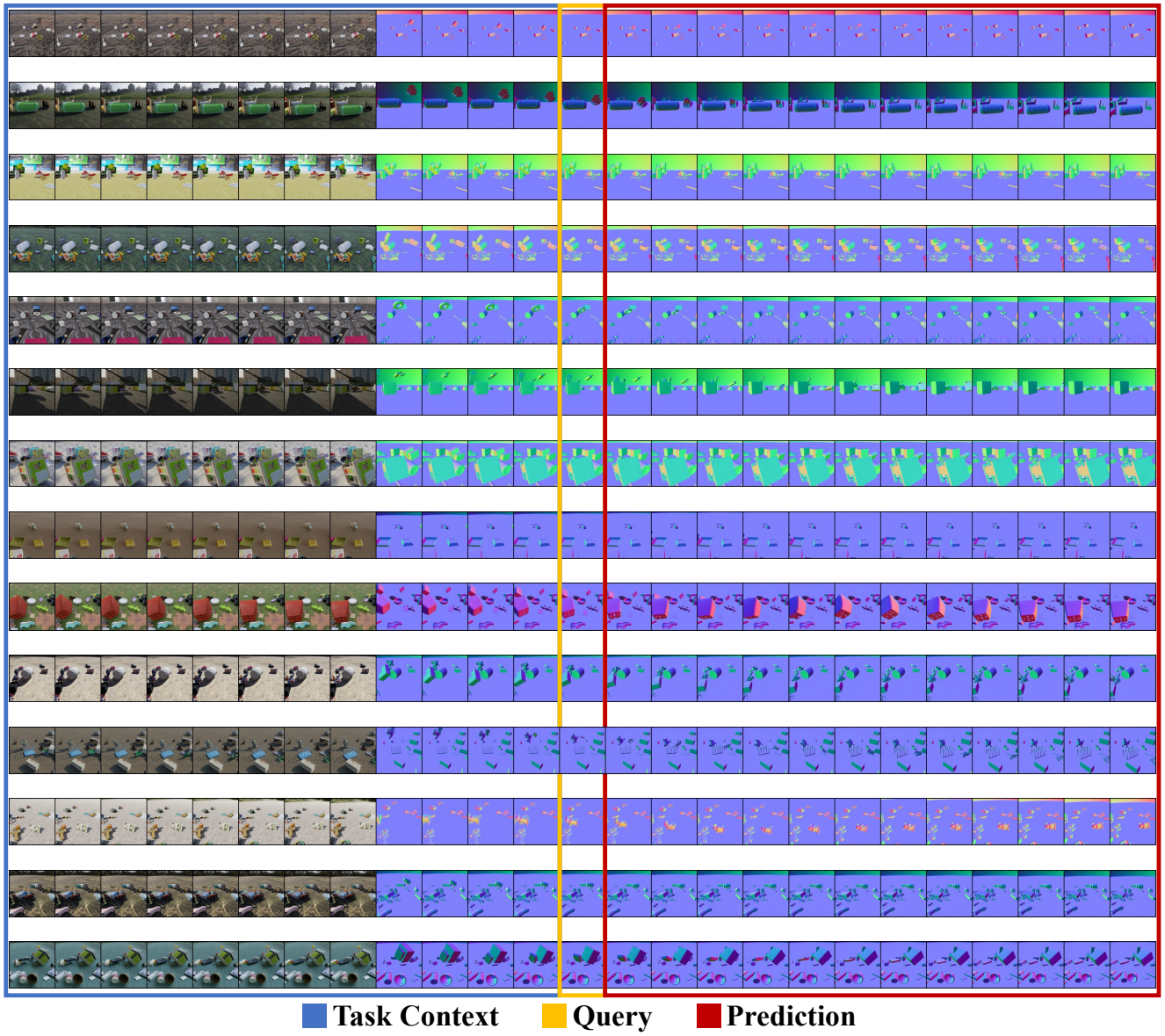

Figure 19. **Qualitative results on video surface normal estimation.** Each row includes a video sequence with a series of target frames as task context (marked in blue), followed by a query frame (marked in yellow). A set of frames in the red box indicates the model's predictions. Due to the length of the sequence, a portion of the task context is hidden. *Best viewed in color.*

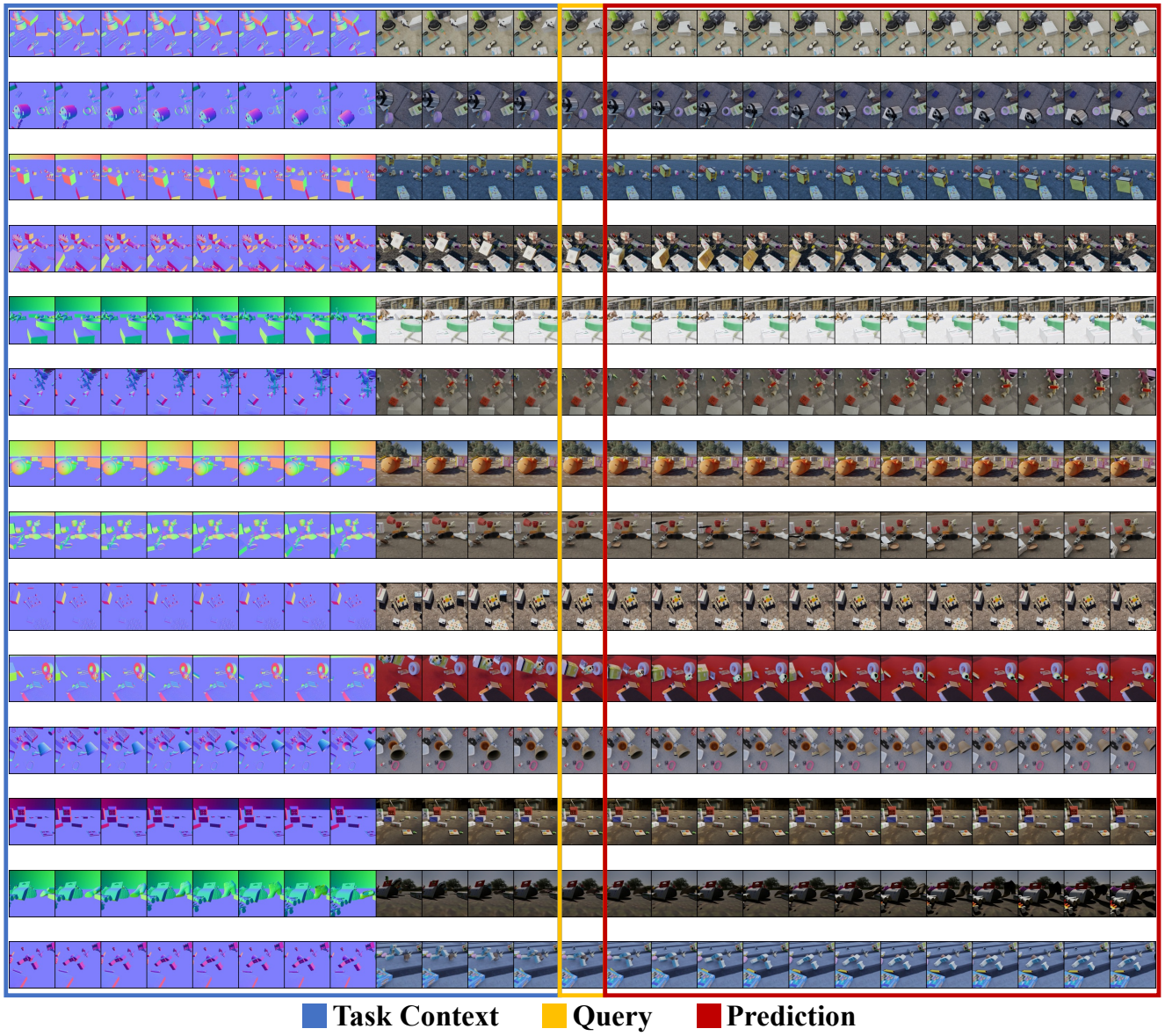

Figure 20. **Qualitative results on normal-to-video generation.** Each row includes a video sequence with a series of target frames as task context (marked in blue), followed by a query frame (marked in yellow). A set of frames in the red box indicates the model's predictions. Due to the length of the sequence, a portion of the task context is hidden. *Best viewed in color.*

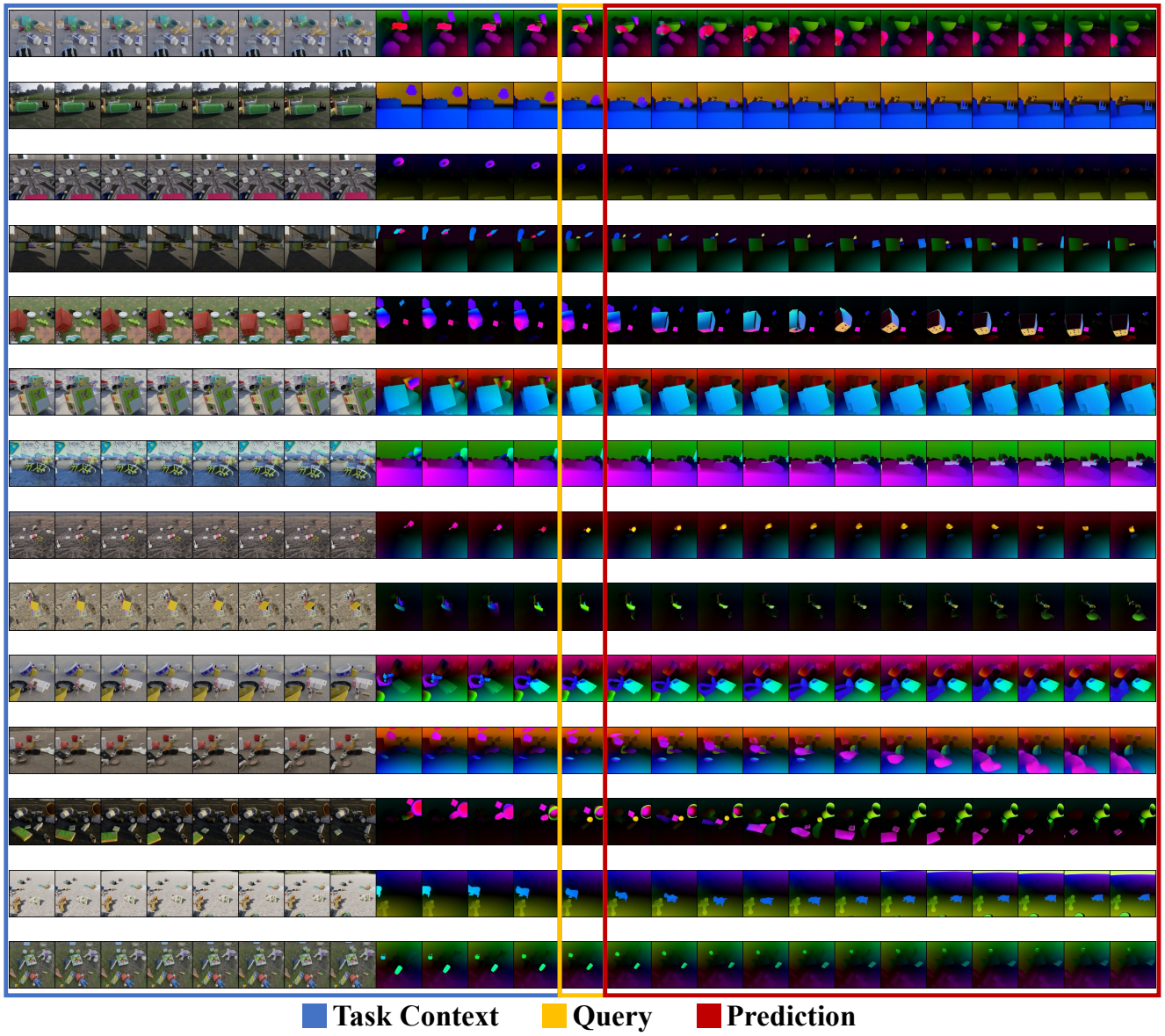

Figure 21. **Qualitative results on optical flow estimation.** Each row includes a video sequence with a series of target frames as task context (marked in blue), followed by a query frame (marked in yellow). A set of frames in the red box indicates the model's predictions. Due to the length of the sequence, a portion of the task context is hidden. *Best viewed in color.*

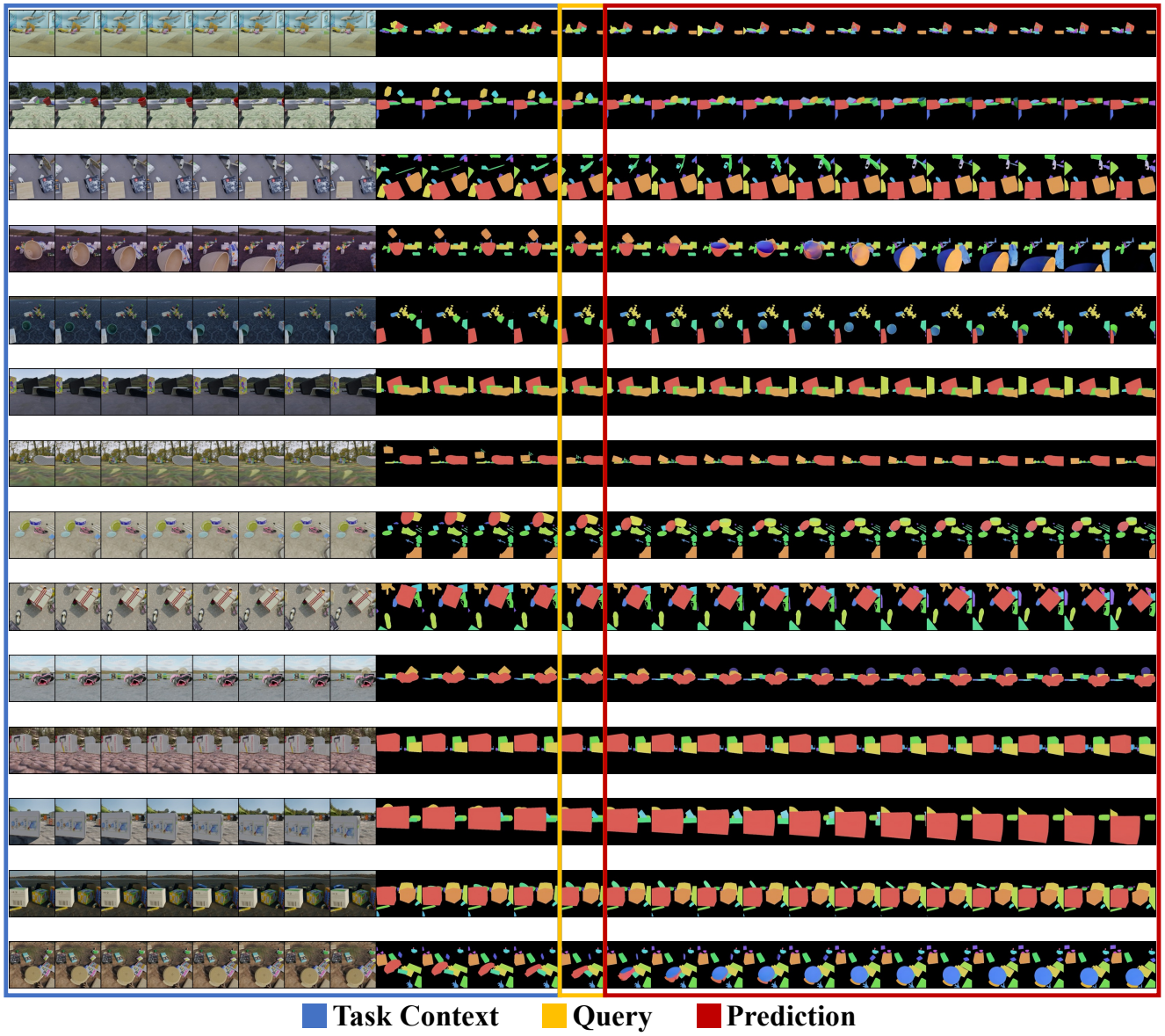

Figure 22. **Qualitative results on video instance segmentation.** Each row includes a video sequence with a series of target frames as task context (marked in blue), followed by a query frame (marked in yellow). A set of frames in the red box indicates the model's predictions. Due to the length of the sequence, a portion of the task context is hidden. *Best viewed in color.*

Figure 23. **Potential application of single-view scene reconstruction.** Given an RGB image and predicted depth map, we lift this image into a 3D space. We illustrate three views of this scene. *Best viewed in color.*

References

[1] Gwangbin Bae and Andrew J Davison. Rethinking inductive biases for surface normal estimation. In *CVPR*, pages 9535–9545, 2024. 2

[2] Amir Bar, Yossi Gandelsman, Trevor Darrell, Amir Globerson, and Alexei Efros. Visual prompting via image inpainting. In *NeurIPS*, pages 25005–25017, 2022. 2

[3] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, pages 3213–3223, 2016. 3

[4] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, pages 248–255, 2009. 2, 3

[5] Klaus Greff, Francois Belletti, Lucas Beyer, Carl Doersch, Yilun Du, Daniel Duckworth, David J Fleet, Dan Gnanapragasam, Florian Golemo, Charles Herrmann, et al. Kubric: A scalable dataset generator. In *CVPR*, pages 3749–3761, 2022. 3

[6] Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, et al. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017. 3

[7] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, pages 740–755, 2014. 3

[8] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *ICLR*, 2023. 1

[9] René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *IEEE transactions on pattern analysis and machine intelligence*, 44(3):1623–1637, 2020. 2

[10] Amirreza Shaban, Shray Bansal, Zhen Liu, Irfan Essa, and Byron Boots. One-shot learning for semantic segmentation. *arXiv preprint arXiv:1709.03410*, 2017. 2

[11] Shuai Shao, Zeming Li, Tianyuan Zhang, Chao Peng, Gang Yu, Xiangyu Zhang, Jing Li, and Jian Sun. Objects365: A large-scale, high-quality dataset for object detection. In *ICCV*, pages 8430–8439, 2019. 3

[12] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, pages 746–760, 2012. 2

[13] K Soomro. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012. 3

[14] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018. 2

[15] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *CVPR*, pages 633–641, 2017. 3