StableMTL: Repurposing Latent Diffusion Models for Multi-Task Learning from Partially Annotated Synthetic Datasets
–Supplementary Material–

Anh-Quan Cao[1] Ivan Lopes[2] Raoul de Charette[2]
[1]Valeo.ai [2]Inria

https://github.com/astra-vision/StableMTL

In this supplementary material, we report additional ablations of StableMTL in Sec. B.1. For our method and baselines in Sec. B.2 we report more metrics for **semantic**, **depth**, **shading** and **albedo**, and detail in-domain performances in Sec. B.3. Additionally, we describe our task encoding scheme in Sec. B.4, provide visualization details for each task in Sec. B.5, and detail the semantic class mapping used for training in Sec. B.6. Image resolutions are reported in Sec. B.7. Finally, we present further analyses regarding scalability in Sec. B.9, task-attention mechanisms in Sec. B.10, and computational costs in Sec. B.11.

We direct the reader to the **supplementary video** at https://youtu.be/exsVFj34lrI and Sec. B.8 for a qualitative assessment of our performance.

B.1. Additional ablations

Effect of task attention masking. We ablate the task attention masking in Tab. 6. Masking a single task proportional to its attention weight (Sample(π_T)) improves performance as the masking probability ρ increases to an optimum (*i.e.*, ρ=4), as this fosters exploration of diverse task combinations. However, higher ρ values produces excessive exploration, lowering performance. Consequently, masking a random number of k tasks at once (Sample$_k$(π_T)) is less effective. Other approaches are also suboptimal: dropping the highest-attention task (argmax) prevents exploitation of the strongest signal, while randomly dropping tasks ($\mathcal{U}(\mathcal{T})$) offers a weaker exploration incentive.

B.2. Additional metrics

We present IoU scores for each mapped class on the Cityscapes dataset [5] in Tab. 7, with class mapping details provided in Tab. 11. Overall, StableMTL ranks either first or second on all classes.

For **depth** evaluation on KITTI [7] and DIODE [12], additional depth metrics are reported in Tab. 8; these include the following commonly used metrics [6]: absolute relative error (Abs Rel), squared relative error (Sq Rel), root mean squared error (RMSE), mean $\log_{10}$ error (RMSE log), and threshold accuracies (δ1, δ2, δ3). On most of these metrics, StableMTL surpasses all MTL baselines, only performing below the single-stream model StableMTL-$\mathcal{S}$ on one metric in one dataset.

We also report, in Tab. 9, metrics for **shading** and **albedo** on the MID dataset [10] following [3]: structural similarity index (SSIM) [13], local mean squared error (LMSE), and root mean squared error (RMSE). These results show our method outperforms all other MTL baselines.

Strategy	Prob. (ρ)	Semantic mIoU %↑ Cityscapes	Normal mAE °↓ DIODE	Depth AbsRel %↓ KITTI	Depth AbsRel %↓ DIODE	Opt. Flow EPE-2D px↓ KITTI	Scene Flow EPE-3D m↓ KITTI	Shading RMSE↓ MID	Albedo RMSE↓ MID	MTL Perf. Δ_m%↑ Avg
	0.0	54.90	22.88	14.90	32.57	11.45	0.2400	0.2351	0.2023	+3.41
	0.2	54.86	23.49	14.71	32.61	11.00	0.2378	0.2349	0.2047	+3.70
Sample(π_T)	**0.4**	55.08	23.36	14.03	32.97	11.22	0.2297	0.2350	0.2061	+4.14
	0.6	53.97	23.26	15.22	32.94	10.90	0.2284	0.2348	0.2029	+4.00
	0.8	52.95	23.02	14.73	32.89	11.21	0.2386	0.2349	0.2093	+2.72
Sample$_k$(π_T)	0.4	54.65	23.82	15.02	33.15	10.96	0.2311	0.2338	0.2064	+3.50
argmax	*idem*	54.45	23.09	14.98	32.87	11.22	0.2299	0.2339	0.2071	+3.65
$\mathcal{U}(\mathcal{T})$	*idem*	54.77	22.69	15.02	32.70	11.32	0.2398	0.2313	0.2057	+3.60

Table 6. **Ablation of attention masking.** Masking heads depending on their attention score using our distribution Sample(π_T) boosts performance by encouraging exploration. On the contrary, other methods, detailed in the text, perform worse. Best and Second-best are highlighted.

Method	road	building	pole	traffic light	traffic sign	vegetation	sky	vehicle	mIoU
Single-task baseline	89.36	61.82	17.13	0.42	3.21	72.75	67.86	72.82	48.17
JTR [11]	96.33	51.48	15.33	0.87	3.22	61.28	77.36	60.16	45.75
JTR* [11]	71.50	0.00	0.00	0.00	0.00	38.69	53.47	0.00	20.46
DiffusionMTL [14]	85.17	29.60	6.64	1.14	5.19	41.05	27.34	44.57	30.09
DiffusionMTL* [14]	95.36	68.26	11.01	4.27	6.40	65.61	48.35	68.15	45.92
StableMTL-$\mathcal{S}$	92.06	67.05	16.76	2.03	6.31	72.53	83.84	80.06	52.57
StableMTL	96.08	74.24	20.72	2.80	12.59	76.58	82.09	81.23	55.79

Table 7. **Per-class Semantic performance**. We report the IoU per mapped semantic class as well as the average IoU (mIoU), detailed in Tab. 11, on Cityscapes.

Method	KITTI							DIODE						
	Abs Rel↓	Sq Rel↓	RMSE↓	RMSE log↓	δ1↑	δ2↑	δ3↑	Abs Rel↓	Sq Rel↓	RMSE↓	RMSE log↓	δ1↑	δ2↑	δ3↑
Single-task baseline	14.21	0.7319	4.1734	0.2014	81.19	96.52	99.16	32.56	3.9233	3.9632	0.3163	73.72	87.72	93.11
JTR [11]	26.39	1.7532	6.1357	0.2852	64.60	88.30	95.86	66.39	8.1448	8.4119	0.5482	42.26	67.66	81.59
JTR* [11]	39.27	4.0532	9.1290	0.4301	42.99	73.19	88.00	73.14	9.2842	8.9962	0.5852	40.31	64.24	78.10
DiffusionMTL [14]	21.10	1.4998	5.8491	0.2715	68.11	89.87	96.80	45.19	4.6659	4.9657	0.4106	56.93	78.91	88.56
DiffusionMTL* [14]	24.83	1.8492	6.4705	0.3518	58.77	86.54	95.59	58.17	7.2017	7.6032	0.5166	49.57	73.62	84.55
StableMTL-$\mathcal{S}$	15.64	0.8268	4.3713	0.2499	77.62	95.52	98.72	33.36	3.9504	4.0281	0.3227	73.42	87.17	92.62
StableMTL	14.98	0.8224	4.3707	0.2170	79.43	95.94	98.87	33.03	3.9516	4.0073	0.3192	73.72	87.45	92.89

Table 8. **Additional depth metrics.** In all but one metric, StableMTL ranks first on both KITTI [7] and DIODE [12].

Method	Shading			Albedo		
	RMSE↓	SSIM↑	LMSE↓	RMSE↓	SSIM↑	LMSE↓
Single-task baseline	0.2551	0.4277	0.0426	0.2145	0.6067	0.0505
JTR* [11]	0.3030	0.1843	0.0530	0.3567	0.3102	0.0994
DiffusionMTL* [14]	0.3064	0.3142	0.0487	0.3660	0.3606	0.1620
StableMTL-$\mathcal{S}$	0.2311	0.4449	0.0363	0.2077	0.6151	0.0496
StableMTL	0.2346	0.4424	0.0366	0.2016	0.6199	0.0477

Table 9. **Additional shading and albedo metrics.** On MIDIntrinsics [10], StableMTL ranks either first or second; it is only surpassed by our single stream variant StableMTL-$\mathcal{S}$.

B.3. In-domain evaluation

Finally, we report the performances on test splits of in-domain data in Tab. 10. Our method beats all baselines by a large margin on all metrics.

Method	Semantic	Normal		Depth		Opt. Flow		Scene Flow		Shading	Albedo	MTL Perf.
	mIoU %↑	mAE °↓	mAE °↓	AbsRel %↓	AbsRel %↓	EPE-2D px↓	EPE-2D px↓	EPE-3D m↓	EPE-3D m↓	RMSE↓	RMSE↓	Δ_m%↑
	VKITT2	VKITT2	Hypersim	VKITT2	Hypersim	VKITT2	FlyingThings3D	VKITT2	Hypersim	Hypersim	Hypersim	Avg
Single-task baseline	64.43	24.96	17.97	19.62	17.76	9.30	12.87	0.1508	0.4201	0.2205	0.2478	0.00
DiffusionMTL* [14]	38.83	56.27	68.98	34.94	34.20	32.45	32.52	0.2630	0.7094	0.2866	0.5724	-109.02
JTR* [11]	36.36	51.68	54.73	66.23	59.83	32.29	35.04	0.3516	0.7454	0.2457	0.3865	-117.00
DiffusionMTL [14]	51.63	27.13	23.65	29.94	26.99	n/a	n/a	n/a	n/a	n/a	n/a	n/a
JTR [11]	42.21	29.56	30.47	35.38	37.05	n/a	n/a	n/a	n/a	n/a	n/a	n/a
StableMTL-$\mathcal{S}$	62.53	27.12	21.52	20.69	20.25	13.88	21.10	0.1611	0.5571	0.2021	0.2423	-13.24
StableMTL	67.98	25.39	19.30	19.11	18.93	9.44	14.53	0.1263	0.4382	0.2012	0.2366	+1.56

Table 10. **In-domain model performance.** We compare the different approaches by evaluating on in-domain test sets.

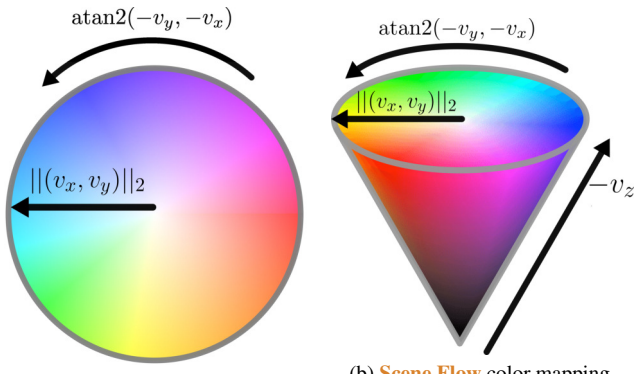
B.4. Task encoding

For unbounded quantities – **depth**, **optical flow**, and **scene flow** – we adopt affine-invariant representations as in [8]. Specifically, we apply independent linear scaling by mapping values from $[a, b]$ to $[-1, +1]$. Where (a, b) are the 2nd/98th percentiles for **depth** and min/max values for **optical flow** and **scene flow**. For categorical tasks such as **semantic** segmentation, the annotations are discrete, $y_{\text{seg}} \in [\![1, C]\!]^{H \times W}$, for C classes in total. We define a mapping $f_{\text{seg}} : [\![1, C]\!] \to [-1, +1]^3$ assigning a unique RGB vector to each class. Following [4], we tone-map **albedo** and **shading** with a scalar scale, then clamp and remap the values to $[-1, +1]$. After appropriate scaling, we standardize all shapes to match $\mathbb{M}^3$ by repeating channels. For **depth** and **shading** (both in $\mathbb{M}$), the grayscale maps are repeated three times. For **optical flow** ($y_{\text{o-flow}} \in \mathbb{M}^2$), we choose to repeat the horizontal flow once. Finally **scene flow**, **normal**, and **albedo** already belong to $\mathbb{M}^3$.

As a post-processing step for **semantic** segmentation, we obtain final prediction $\hat{y}_{\text{seg}}$ by first decoding the predicted latent with $y'_{\text{seg}} = \mathcal{D}(\hat{z}_{\text{seg}})$, then applying nearest neighbors search in the RGB space for each pixel (i, j): $\hat{y}_{\text{seg}}(i, j) = \arg\min_{c \in [\![1,C]\!]} \|\hat{y}'_{\text{seg}}(i, j) - f_{\text{seg}}(c)\|_2$. For **optical flow**, the prediction consists of the first two channels of the decoded latent. For **depth** and **shading**, we utilize the average of the three decoded channels. Other tasks do not require any post-processing.

B.5. Task Visualization

We detail here the visualization for each task. **Semantic** segmentation maps employ the Cityscapes color scheme (see Tab. 11). For **optical flow** $\mathbf{v}(u, v) = (v_x, v_y)$, we adopt the HSV color encoding from [2], where the lateral flow vector's (v_x, v_y) angle atan2$(-v_y, -v_x)$ determines hue and its magnitude $\|(v_x, v_y)\|_2$ defines saturation, as illustrated in Fig. 12a. Similarly, **Scene Flow** $\mathbf{v}(u, v) = (v_x, v_y, v_z)$ utilizes an HSV representation (cf. Fig. 12b): its lateral component (v_x, v_y) determines hue (angle: atan2$(-v_y, -v_x)$) and saturation (magnitude: $\|(v_x, v_y)\|_2$), while the depth flow component v_z is inversely mapped to value (brightness), with image-specific scaling applied to both the magnitude and v_z. For **normal** visualization, surface normal XYZ coordinates are directly mapped to RGB space. We predict outward-facing surface normals in OpenCV coordinate system following [1]. **Depth** visualization involves mapping scale-invariant depth values to the spectral color map before display. Finally, **shading** and **albedo** are directly visualized as grayscale and RGB images, respectively.



(a) **Optical Flow** color mapping. (b) **Scene Flow** color mapping.

Figure 12. **Flow color mappings.** We visualize the mapping used to visualize (a) **optical flow** and (b) **scene flow**.

B.6. Semantic class mapping

To allow evaluation of **semantic** on the Cityscapes dataset, our model is trained using only a subset of 8 classes, following the VKITTI 2↦Cityscapes class mapping from [9], detailed in Tab. 11.

VKITTI 2	ours	Cityscapes	ours	Color
terrain	*ignore*	road	road	
sky	sky	sidewalk	*ignore*	
tree	vegetation	building	building	
vegetation	vegetation	wall	vegetation	
building	building	fence	*ignore*	
road	road	pole	pole	
guardrail	*ignore*	light	light	
sign	sign	sign	sign	
light	light	vegetation	vegetation	
pole	pole	sky	sky	
misc	*ignore*	person	*ignore*	
truck	vehicle	rider	*ignore*	
car	vehicle	car	vehicle	
van	vehicle	bus	vehicle	
		motorbike	*ignore*	
		bike	*ignore*	

Table 11. **Classes used for training.** We use a common set of 8 classes (ours), mapping together semantically similar classes of VKITTI 2 for proper evaluation of our model on Cityscapes.

B.7. Image resolutions

We train the model on datasets using the following pixel resolutions, `height×width`: Hypersim (288×384), FlyingThings3D (268×480), VKITTI2 (187×621).
For evaluations, we use the following resolutions: Cityscapes (256×512), DIODE (384×512), KITTI (176×608), MIDIntrinsics (256×384).

B.8. Additional qualitative results

In Fig. 14 we report additional qualitative results of StableMTL, and the supplementary video further demonstrate the better of StableMTL compared to the existing baselines.

B.9. Scalability of StableMTL

On a H100 GPU, Stage 1 (single-stream) trains in 10hr and Stage 2 (multi-stream) in 20hr. During inference, all tasks are batch-predicted in parallel. As tasks increase $2 \rightarrow 7$, compute ($\sim 2.5 \rightarrow 7.7$ TFLOPs) and wall-clock ($\sim 0.095 \rightarrow 0.18$s) grow linearly though the latter has a much smaller slope,as shown in Fig. 13.

B.10. N-to-1 *vs*. N-to-N task-attention

We conduct a comparison on (**Semantic**, **Normal**, **Depth**), where independently encoded tasks are concatenated and jointly processed to enable full N-to-N interactions, with supervision applied on all available labels. N-to-N incurs 50% higher memory and degrades performance (KITTI AbsRel 13.92→15.72, Normal MAE 21.88→22.95), especially for semantics (mIoU 51.03→32.71) which is further dominated, likely due to its smaller gradients (Fig. 6b).

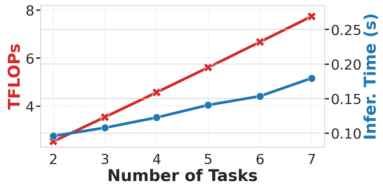


Figure 13. **Scalability of StableMTL.** Observing complexity (TFLOPs) and inference time (s) demonstrates that the latter increases slower than complexity when increasing the number of tasks, therefore showcasing scalability of our method.

Method	Time (ms)	TFLOPs	Memory (GB)	Δ_m (%)
JTR*	0.006	0.07	0.92	-106.87
DiffusionMTL*	0.110	2.89	25.69	-78.76
StableMTL-$\mathcal{S}$	0.086	6.08	6.93	-1.57
StableMTL	0.179	7.73	11.75	+4.78

Table 12. **Inference cost.**

B.11. Compute cost

Tab. 12 reports H100 inference latency, TFLOPs, and GPU memory, for a 256×256 image and 7 tasks. We adapt DiffusionMTL to scale to 7 tasks and upgrade the backbone (RN18→RN101) to boost performance. JTR is kept unchanged because of its unconventional network. StableMTL-$\mathcal{S}$ runs in 0.086s and StableMTL in 0.179s for 7 tasks. Latent decoding drives 56% of costs; the U-Net takes only 21% as it operates in 4-channel latent space (≈ 64 times smaller than image-space). StableMTL-$\mathcal{S}$ is 22% faster than DiffusionMTL, and StableMTL uses about half the memory (11.8 *vs*. 25.7GB) thanks to parallelized computation, despite higher TFLOPs.

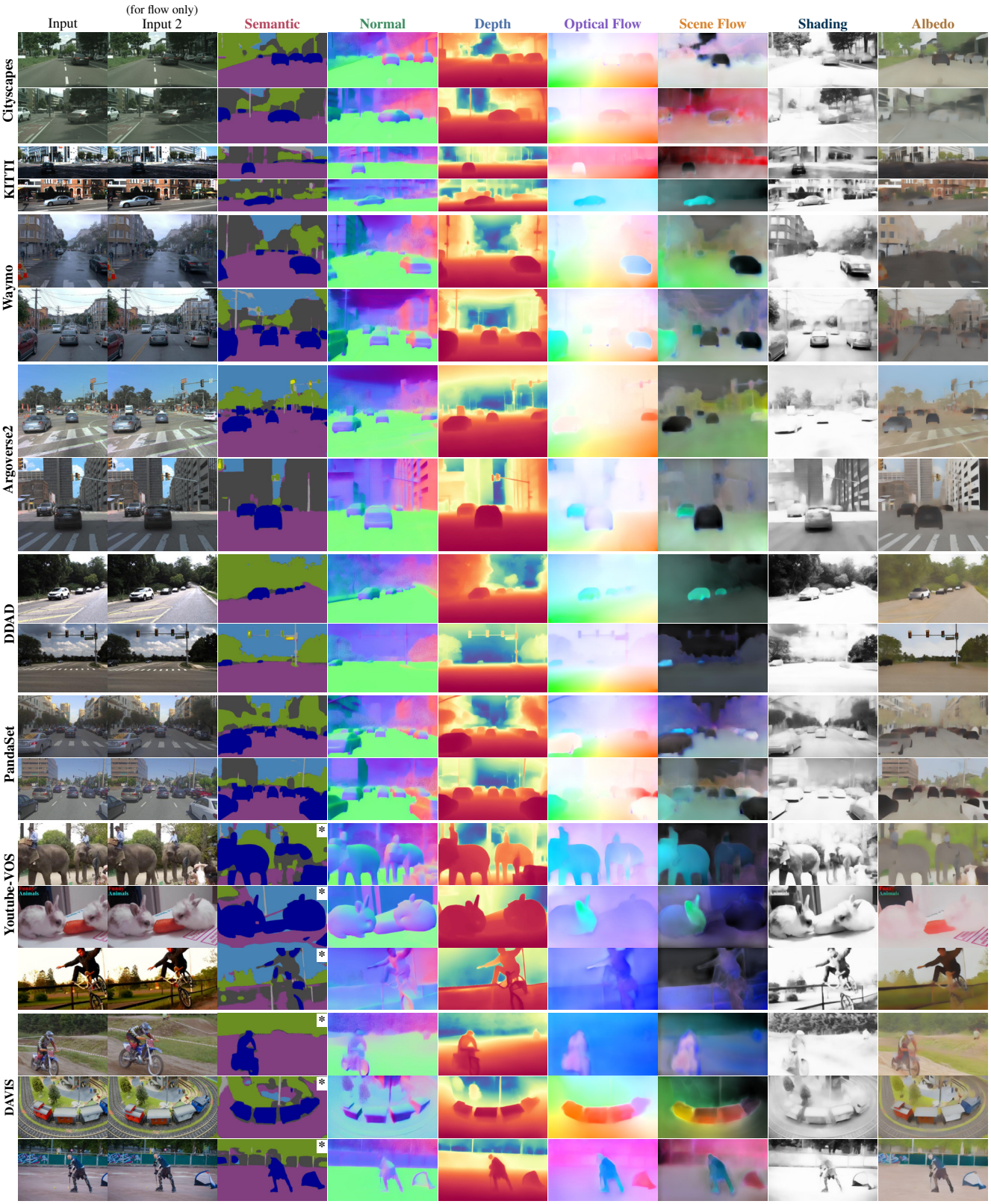


Figure 14. **Additional qualitative results on real-world data.** Despite being trained on partially annotated synthetic datasets, StableMTL demonstrates generalization to multi-task real-world scenarios. *Note that **semantic** is trained on closed-set driving classes although it generalizes to some extent to other domains.

References

[1] Bae, G., Davison, A.J.: Rethinking inductive biases for surface normal estimation. In: CVPR (2024) 2

[2] Baker, S., Roth, S., Scharstein, D., Black, M.J., Lewis, J., Szeliski, R.: A database and evaluation methodology for optical flow. In: ICCV (2007) 2

[3] Careaga, C., Aksoy, Y.: Intrinsic image decomposition via ordinal shading. In: ACM TOG (2023) 1

[4] Careaga, C., Aksoy, Y.: Colorful diffuse intrinsic image decomposition in the wild. In: ACM TOG (2024) 2

[5] Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., Schiele, B.: The cityscapes dataset for semantic urban scene understanding. In: CVPR (2016) 1

[6] Eigen, D., Puhrsch, C., Fergus, R.: Depth map prediction from a single image using a multi-scale deep network. In: NeurIPS (2014) 1

[7] Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? the kitti vision benchmark suite. In: CVPR (2012) 1, 2

[8] Ke, B., Obukhov, A., Huang, S., Metzger, N., Daudt, R.C., Schindler, K.: Repurposing diffusion-based image generators for monocular depth estimation. In: CVPR (2024) 2

[9] Lopes, I., Vu, T.H., de Charette, R.: DenseMTL: Cross-task attention mechanism for dense multi-task learning. In: WACV (2023) 2

[10] Murmann, L., Gharbi, M., Aittala, M., Durand, F.: A multi-illumination dataset of indoor object appearance. In: ICCV (2019) 1, 2

[11] Nishi, K., Kim, J., Li, W., Pfister, H.: Joint-task regularization for partially labeled multi-task learning. In: CVPR (2024) 1, 2

[12] Vasiljevic, I., Kolkin, N., Zhang, S., Luo, R., Wang, H., Dai, F.Z., Daniele, A.F., Mostajabi, M., Basart, S., Walter, M.R., Shakhnarovich, G.: DIODE: A Dense Indoor and Outdoor DEpth Dataset. In: CoRR (2019) 1, 2

[13] Wang, Z., Bovik, A., Sheikh, H., Simoncelli, E.: Image quality assessment: from error visibility to structural similarity. In: TIP (2004) 1

[14] Ye, H., Xu, D.: DiffusionMTL: Learning multi-task denoising diffusion model from partially annotated data. In: CVPR (2024) 1, 2