arXiv:2506.08013v1 [cs.CV] 9 Jun 2025

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

Anh-Quan Cao[1] **Ivan Lopes**[2] **Raoul de Charette**[2]

[1]Valeo.ai [2]Inria

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

Abstract

Multi-task learning for dense prediction is limited by the need for extensive annotation for every task, though recent works have explored training with partial task labels. Leveraging the generalization power of diffusion models, we extend the partial learning setup to a zero-shot setting, training a multi-task model on multiple synthetic datasets, each labeled for only a subset of tasks. Our method, StableMTL, repurposes image generators for latent regression. Adapting a denoising framework with task encoding, task conditioning and a tailored training scheme. Instead of per-task losses requiring careful balancing, a unified latent loss is adopted, enabling seamless scaling to more tasks. To encourage inter-task synergy, we introduce a multi-stream model with a task-attention mechanism that converts N-to-N task interactions into efficient 1-to-N attention, promoting effective cross-task sharing. StableMTL outperforms baselines on 7 tasks across 8 benchmarks.

1 Introduction

For many real-world applications, multi-task learning (MTL) offers a vital alternative to classical single-task models since MTL allows solving multiple tasks at once. This is of utmost importance for computer vision systems interacting with our world such as robotics or virtual reality which typically require estimating various scene cues (*e.g.*, semantic segmentation, depth, optical flow, etc.) in a time-sensitive manner. Beyond practical aspects, a vast body of literature shows that different tasks learn at least partially distinct representations that can benefit other tasks [74, 54]. Because it learns multiple tasks at once, MTL offers an ideal setting to enforce information exchange across tasks [57]. However, the benefits of MTL is impeded by the small number of datasets with multiple overlapping task labels, especially for dense predictions given the cost of pixel-level annotation [46].

An alternative is to train in a partially labeled setup [32] – that is, using images with annotations for a subset of all tasks. This differs from prior semi-supervised setting [47, 21] which typically overlooks the crucial inter-task relationships. Yet, existing partially annotated MTL methods [32, 71, 46] are confined to training on a single domain, therefore offering poor generalization capability, and struggling to balance conflicting task objectives [57].

To reduce reliance on extensive annotations, we propose a new setting, leveraging *multiple synthetic datasets* with partial labels. This, however, introduces a synthetic-to-real domain gap, necessitating models with strong generalization capability to real-world data. We address this by building upon recent findings that pre-trained Latent Diffusion Models (LDMs), originally designed for image generation, can be repurposed for dense prediction tasks [27, 25, 40, 66] by preserving the LDM's rich latent space while fine-tuning only the UNet. In fact, such models exhibit remarkable real-world generalization, even when trained on relatively small synthetic datasets.

Preprint. Under review.

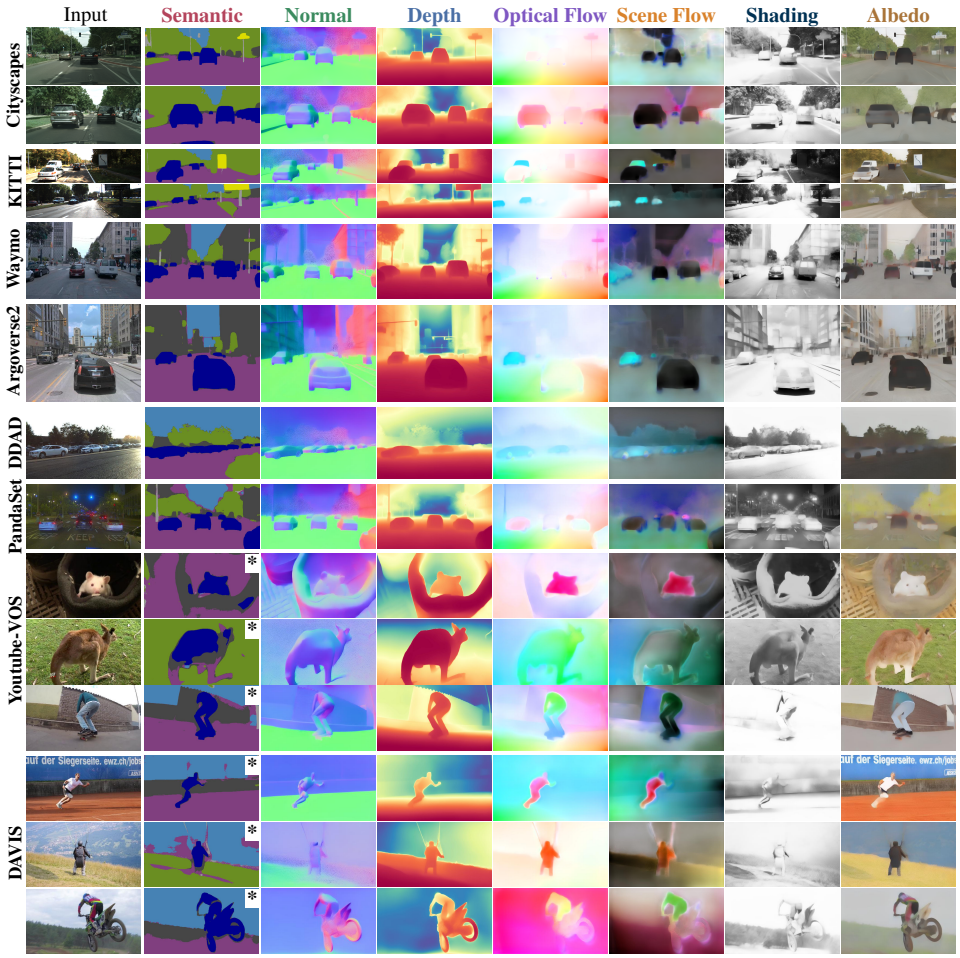


Figure 1: **StableMTL output on unseen real-world data.** StableMTL demonstrates robust generalization to real-world data, despite being trained on partially labeled synthetic datasets. * Note that **semantic** is trained on driving classes and is not expected to generalize to unseen classes.

In practice, our method StableMTL, extends deterministic single-step LDMs [25] to the partially labeled multi-task setting with task-token conditioning. Additionally we make use of a custom training scheme that decouples task-specific gradients. To benefit from cross-task interactions, rather than N-to-N attentions [37, 7], which scales poorly to higher number of tasks, our multi-stream architecture enables a more efficient N-to-1 task inter-task attention mechanism. While adding tasks typically necessitates specific losses, our approach eliminates this requirement by unifying them into a single and simple latent Mean Squared Error (MSE) loss, thereby significantly enhancing scalability. In summary, our methodological contributions are twofold:

- We introduce a more realistic partially labeled MTL setting, training on multiple synthetic datasets, each with partial task annotations.
- Our method, StableMTL, repurposes pre-trained image generation Latent Diffusion Models for MTL, using a multi-stream architecture favoring task exchanges via N-to-1 attention. By employing a single, unified latent loss across all tasks alongside a tailored training scheme, our method eliminates the need for task-specific losses and complex inter-task balancing.

As shown in Fig. 1, while training on only three relatively small-scale synthetic datasets, StableMTL generalizes effectively to a vast range of real-world scenarios, including unseen structures and domains significantly out-of-distribution (*e.g.*, DAVIS and Youtube-VOS), in a zero-shot fashion. Further, it outperforms existing partially labeled MTL methods on 7 tasks, across 8 benchmarks, with a remarkable +83.54 Δ_{m} overall improvement.

2 Related works

Multi-Task Learning (MTL). There is a large body of literature on MTL for dense tasks, surveyed in [57], which rely on various architectural choices like task-specialized modules [51, 11, 43], dynamically adapting architectures [39, 14, 24, 1], integrating attention mechanisms for feature sharing [77, 55, 7], or use of transformer-based models [70, 68, 17]. Closer to us, the rise of large-scale pre-trained models spurred adapter-based fine-tuning methods [36, 33]. To cope with multitask objectives people uses adaptive loss weighting [29], gradient manipulations [12, 13, 72, 53], or seek Pareto-optimal solutions[44, 34]. A key aspect of MTL is how to enhance task exchange [74], which has been enforced for out-of-distribution performance [73, 52], utilizing consistency losses [38], or learning joint task spaces [46, 32]. These relation-based approaches typically rely on consistency losses or pairwise constraints, which hinder scalability and stability as task counts increases.

MTL with partial annotation. Early efforts addressed partially labeled shallow models [35, 76, 60], subsequent deep learning approaches like MTPSL [32], Dejavu [6], DiffusionMTL [71], and JTR [46] (see [18] for a comprehensive review) have predominantly relied on simulating missing labels within single, fully-annotated datasets. This setup not only fails to fully capture real-world complexities where annotations are naturally sparse across different data sources but also hinges on the use of real-world datasets which are expensive to annotate densely for many tasks. SemiMTL [61], which explored multi-dataset MTL, was restricted to two tasks and real-world data. Motivated by these gaps and the prohibitive cost of annotating numerous tasks on real-world data, we propose a more realistic, scalable, and challenging scenario: learning a greater number of tasks from multiple synthetic datasets, each with naturally partially overlapping sets of task annotations.

Repurposing Latent Diffusion Models (LDMs) for dense prediction. LDMs for image generation, known for robust representations, were first repurposed by Marigold [27, 28] for dense prediction tasks like depth estimation, demonstrating strong generalization from small synthetic datasets. Subsequent efforts extended this to other individual tasks such as geometry [19], normal estimation [69], motion [63] and also improved efficiency with enhanced sampling [22] or single-step latent regression [25, 40, 66]. in In the LDM context, holistic MTL remains largely unexplored. Recently, large-scale LDM-based visual foundation models like Diception [78] and OneDiffusion [31] emerged, handling diverse tasks by training on large-scale real-world data. However, their primary focus is not on holistic MTL capabilities when restricted to synthetic-only training or small-scale, partially labeled data. Our approach is complementary, offering a method to fine-tune these models for multi-task using only small-scale, partially labeled synthetic datasets.

3 StableMTL

We address the problem of multi-task partially supervised learning [32] by extending it to a more realistic setting where *partial* labels are obtained from *multiple synthetic* datasets. In summary, our aim is to learn K tasks from a mixture of N synthetic datasets, where each dataset has only access to labels for fewer than K tasks.

Our method, coined StableMTL, builds on recent advances in diffusion models repurposed for dense task estimation [25] allowing us to generalize to the real domain, in a zero-shot fashion. Importantly, we frame multi-task learning as a latent regression problem thus *alleviating the need for task-specific losses and weighting*. To train our complete architecture, depicted in Fig. 2, we follow a two-stage process. In the first stage, detailed in Sec. 3.1, we fine-tune a latent diffusion model repurposed for dense prediction [25], which we adapt to multi-task learning by incorporating task tokens, a unified latent loss and a custom training scheme. For clarity, we illustrate the first stage in Fig. 8 (Appendix A.2). In the second stage, to encourage task exchange, we learn a separate task-conditioning model where interactions among tasks are modeled through a task-attention mechanism. A byproduct of our unified latent loss and our careful design choices is that StableMTL can scale to various spatial and temporal tasks without the usual need for cumbersome task balancing [18].

Our training set, detailed in Tab. 1, comprises three synthetic datasets and seven tasks. These tasks include: high-level **semantic** segmentation; two geometrical tasks, namely **normal** and **depth**; two dynamic flow-based estimations, **optical flow** (pixel displacement in screen-space) and **scene flow** (3D displacement in camera-space); and two low-level intrinsic component estimations, **shading** (grayscale irradiance for lighting) and **albedo** (diffuse reflectance for material).

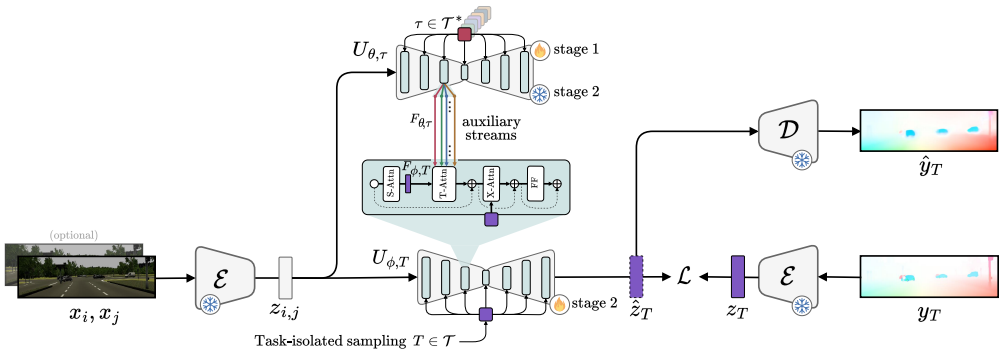


Figure 2: **Overview of StableMTL.** Our pipeline comprises two training stages. In the first stage (Sec. 3.1), we fine-tune a UNet ($U_{\theta,\tau}$) to predict target annotation latents from input image latents, conditioned on multi-task tokens sampled via our training scheme to isolate task gradient (see Fig. 8 in Appendix A.2). To encourage task exchanges, the second stage (Sec. 3.2) uses a multi-stream architecture combining the frozen single-stream UNet trained during Stage 1 and a trainable *main* UNet ($U_{\phi,T}$). It further benefits from a novel task attention mechanism which enriches the main UNet's features with task-specific information from its auxiliary counterparts. Both stages are trained with a single MSE latent loss, instead of task-specific losses.

Dataset	Scene	#samples	Semantic	Normal	Depth	Optical Flow	Scene Flow	Shading	Albedo
Hypersim [49]	Indoor	39K		✓	✓			✓	✓
Virtual KITTI 2 [8]	Urban scenes	20K	✓	✓	✓	✓	✓		
Flying Things 3D [41]	Objects	20K				✓	✓		

Table 1: **Training set.** We train on an ensemble of *synthetic only* datasets, spanning over 7 tasks.

3.1 Single-stream architecture

Since we focus on pixel-wise tasks, we denote $\mathbb{M} = \mathbb{R}^{H \times W}$ for brevity. Let $x_i \in \mathbb{M}^3$ be an input image and $y_\tau \in \mathcal{Y}_\tau$ be the corresponding annotation for a task τ from a set of tasks $\mathcal{T}$. Our pipeline revolves around a latent space $\mathcal{Z} \subset \mathbb{M}^4$ in which we perform multi-task learning. We employ the pretrained StableDiffusion [50] VAE consisting of a frozen encoder $\mathcal{E} : \mathbb{M}^3 \to \mathcal{Z}$ and a decoder $\mathcal{D} : \mathcal{Z} \to \mathbb{M}^3$ for the inverse operation. An image latent is computed directly as $z_i = \mathcal{E}(x_i)$. Considering annotations y_τ are of different nature and encoding (*e.g.*, **semantic** encodes discrete class indices while **depth** is continuous), tasks must first be processed by a task-specific function $f_\tau : \mathcal{Y}_\tau \to \mathbb{M}^3$. As a result, task latents are obtained as: $z_\tau = \mathcal{E}(f_\tau(y_\tau))$. A UNet model, denoted by U_θ, parametrized by θ, is fine-tuned to predict in a single step the task latent $\hat{z}_\tau$ from the input latent z_i, *i.e.*, $\hat{z}_\tau = U_\theta(z_i)$. For tasks requiring multi-frame inputs – such as flow estimation tasks – we extend our pipeline to receive multiple input frames. At the decoding stage, a network $\mathcal{D} : \mathcal{Z} \to \mathbb{M}^3$ maps latent representations back to the RGB space. Similarly, predicted maps $\hat{y}_\tau$ require a post-processing step $p_\tau : \mathbb{M}^3 \to \mathcal{Y}_\tau$ after being decoded, resulting in $p_\tau(\mathcal{D}(\hat{z}_\tau))$.

Unlike traditional multi-step diffusion methods [27] with iterative denoising, we employ a single-step, deterministic formulation which leads to faster training and inference times while maintaining generalization ability [25]. We extend such models for multi-task learning by adopting task tokens, initially introduced for dense prediction by diffusion models in RGB↔X [75]. Such conditioning is achieved by providing the UNet with fixed token embeddings for each task c_τ and computing cross-attention between these embeddings and the layer activations. We detail these task tokens in Appendix A.2. The UNet is predicting the task latent as $\hat{z}_\tau = U_\theta(z_i, c_\tau)$, written $U_{\theta,\tau}(z_i)$ for short. Although RGB↔X makes use of the same mechanism, it focuses on image decomposition while we address the broader scope of multi-task learning. Our goal is to benefit from a joint learning of such tasks, but in a multi-dataset and partially-annotated setting.

Multi-frame adaptation. Some tasks such as flow estimation can be solved from a monocular input [2, 3] but are better conditioned using a pair of temporal frames (i, j). To accommodate such spatio-temporal tasks, StableMTL takes as input a channel-wise concatenated pair of latents, being

$z_{i,j} = \texttt{concat}(z_i, z_j)$. The model then predicts the task output as $\hat{z}_\tau = U_{\theta,\tau}(z_{i,j})$. For single-frame tasks (*e.g.*, **semantic**, **depth**) we simply set $j = i$, thereby concatenating z_i with itself. This simple scheme provides a basis for handling both spatial and spatio-temporal tasks in a unified framework.

Task-gradient isolation scheme. While we use a single unified latent loss, we also observe as in classical MTL that some tasks may exhibit larger gradient magnitudes and conflicting directions [18], resulting in task with weaker gradients being dominated by the stronger ones. To address this, we adopt a task-isolation training scheme, where at each training step, each mini-batch contains only a single task. Further, during gradient accumulation – used to simulate larger batch sizes – we accumulate gradients only from mini-batches of the same task and subsequently perform an optimizer step followed by gradient resetting, before proceeding to the next task. For each task available, we minimize the Mean Squared Error (MSE) loss in the latent space between the predicted latent $\hat{z}_\tau$ and the ground-truth latent z_τ:

$$\mathcal{L}(\theta) = \|\hat{z}_\tau - z_\tau\|_2^2 = \|U_{\theta,\tau}(z_{i,j}) - \mathcal{E}(f_\tau(y_\tau))\|_2^2 \,. \tag{1}$$

3.2 Multi-stream architecture

Our above single-stream architecture can handle multiple tasks simultaneously with task tokens, but does not incorporate any explicit tasks exchange mechanisms. Prior works have addressed this by training N tasks jointly with an N-to-N multi-task attention mechanism, where each task attends to all others to learn task interactions [7, 37]. Such strategy, however, scales poorly with the number of tasks. Therefore, we propose a multi-stream architecture, where the main stream attends tasks, processed via the single-stream (Sec. 3.1), making use of a more scalable 1-to-N attention.

Architecture. Our multi-stream architecture, depicted in Fig. 2, extends the single-stream architecture described above by adding an additional UNet. This *main* trainable UNet, denoted $U_{\phi,T}$, is responsible of the final MTL output, conditioned on a *main* task token T (*e.g.*, `[token]`). The single-stream UNet trained in Sec. 3.1, denoted $U_{\theta,\tau}$, is kept frozen and referred as *auxiliary* since it generates task-specific feature streams, for auxiliary tasks, beneficial for solving the main task. During inference, $U_{\theta,\tau}$ produces auxiliary streams for all tasks τ, other than the main task T. The main UNet $U_{\phi,T}$ then predicts the output for T, while attending to these auxiliary features. The training of $U_{\phi,T}$ is similar to Sec. 3.1, with its weights initialized from $U_{\theta,\tau}$.

Task attention. The UNet is comprised of stacked 2D convolutional residual blocks and transformer blocks. Each transformer block contains a spatial attention layer (S-Attn), a cross-attention layer (X-Attn), and a feed-forward network (FF). Spatial attention aggregates information from correlated 2D spatial locations, while cross-attention conditions the features on the task token. In the main UNet, $U_{\phi,T}$, we introduce an additional task-attention layer within each transformer block, placed immediately after the spatial (self-)attention layer. This new layer enriches the main stream features by attending to relevant task-specific features from the auxiliary streams, as depicted in Fig. 3.

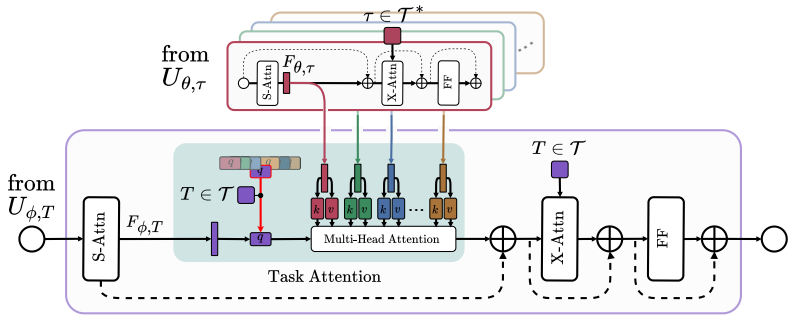


Figure 3: **Proposed task attention.** In addition to standard spatial and cross-attention mechanisms, our transformer blocks in the main UNet incorporate multi-stream information from auxiliary tasks. This is achieved by connecting the dedicated frozen single-stream UNet ($U_{\theta,\tau}$) to the main UNet ($U_{\phi,T}$), providing the latter with auxiliary features. $U_{\theta,\tau}$ is kept frozen.

We define $F_{\phi,T}$ as the feature map from the spatial attention layer in the main UNet $U_{\phi,T}$, conditioned on task T. Let $\mathcal{T}^*{=}\mathcal{T} \setminus \{T\}$ and $\{F_{\theta,\tau} \mid \tau \in \mathcal{T}^*\}$ be the set of corresponding feature maps from the frozen auxiliary UNet $U_{\theta,\tau}$ (the auxiliary streams), each conditioned on an auxiliary task token τ. The cross-attention from the main stream to the auxiliary streams is computed as $\mathrm{Attention}(Q,K,V) = \mathrm{softmax}(QK^\top/\sqrt{d})V$, where

$$Q = \big[q_T\big(F_{\phi,T}\big)\big], \quad K = \big[k_\tau\big(F_{\theta,\tau}\big) \mid \tau \in \mathcal{T}^*\big], \quad V = \big[v_\tau\big(F_{\theta,\tau}\big) \mid \tau \in \mathcal{T}^*\big], \tag{2}$$

and d is the feature dimension. Each task t has its own projection layers (q_t, k_t, v_t), enabling task-specific adaptation. The brackets $[\cdot]$ denote concatenation along a newly introduced *task* dimension, conceptually achieved via an `unsqueeze` operation followed by concatenation.

Attention-guided task masking. As previously mentioned, both the main UNet $U_{\phi,T}$ and the auxiliary UNet $U_{\theta,\tau}$ are initialized with identical pretrained weights from our single-stream training. This shared initialization can cause the task attention mechanism to concentrate heavily (become "spiky") on a few auxiliary tasks early in training. To promote broader exploration and prevent attention saturation, we implement attention-guided task masking within each task-attention layer.

For each spatial location (*i.e.*, "image token") in every task-attention layer, $(\mathbf{s}_{T\to\tau})_{\tau\in\mathcal{T}^*}$ denotes the vector of attention scores from the main stream (task T) to the auxiliary streams (tasks $\tau \in \mathcal{T}^*$). These scores are computed as components of $\mathrm{softmax}(QK^\top/\sqrt{d})$ and represent the attention assigned by task T to auxiliary task τ. Intuitively, tasks with higher scores should have a higher chance of being masked. We model this with the distribution π_T over auxiliary tasks $\mathcal{T}^*$: $\pi_T(\tau) = \frac{\mathbf{s}_{T\to\tau}}{\sum_{\tau\in\mathcal{T}^*}\mathbf{s}_{T\to\tau}}$.

To select a task for masking, we proceed by sampling $m_T \sim \mathrm{Sample}(\pi_T)$ which results in per-task masks: $M(t) = \mathbb{1}[t \neq m_T]$, where $\mathbb{1}$ is the indicator function. This mask is then applied when computing the attention score $\{M(\tau) * \mathbf{s}_{T\to\tau} | \, \tau \in \mathcal{T}^*\}$. To push exploration, we stochastically apply the masking procedure with probability ρ. We later ablate the impact of ρ and alternative sampling strategies. This encourages diverse task combinations and prevents over-reliance on dominant tasks.

4 Experiments

Datasets. As listed in Tab. 1, we train our model for seven tasks using three synthetic datasets: Hypersim [49], VKITTI 2 [8], and FlyingThings3D [41]. At each training step, we first uniformly sample a task. If the selected task is labeled in multiple datasets, we sample from a specific dataset according to the following probabilities: for **depth** and **normal**, we use the sampling as Marigold [27]; for **optical flow** and **scene flow**, we sample FlyingThings3D and VKITTI 2 equally.

For evaluation, we leverage real image datasets. **Normal** is evaluated on DIODE [58]; **depth** on KITTI [20] and DIODE [58]; **semantic** on Cityscapes [15] with the class mapping from [37]; **optical flow** and **scene flow** on KITTI flow [42]; lastly we use MIDIntrinsics [45, 9] for **shading** and **albedo**. Additional real-world datasets are leveraged for qualitative results. Details are in Appendix A.4.

Baselines. We highlight that most MTL methods are designed to be trained on fully annotated datasets, thus not directly comparable. We therefore compare to [71, 46], which also handle partial annotations. Since both DiffusionMTL [71] and JTR [46] are trained with **semantic**, **normal**, and **depth** at most, we report these numbers by training on our ensemble of datasets using the baselines losses and loss weights. For a more direct comparison with our method, we also put significant efforts to adapt baselines to our full multi-dataset and full task set $\mathcal{T}$ setting, including support of temporal tasks which required modification of the architectures. Adaptations are reported as DiffusionMTL* and JTR*. Furthermore, since our single-stream architecture (Sec. 3.1) is also capable of performing multitask, we include its results as StableMTL-$\mathcal{S}$.

Metrics. We rely on mean intersection over union (mIoU) for **semantic**; absolute relative error (AbsRel) for **depth**; mean angular error in degrees (mAE) for **normal**; average endpoint error for **optical flow** (EPE-2D) and **scene flow** (EPE-3D); and root mean squared error (RMSE) for **shading** and **albedo**. For tasks with unbounded quantities, we first perform a per-channel least squares fitting against the ground truth [27, 9]. The delta metric (Δ_m) is used to quantify multi-task performance [57]. It measures the average relative performance over all tasks for a given model compared to its single-task counterparts. For **depth**, the Δ_m is averaged over two datasets.

Implementation. We provide details on how each task is encoded into the latent space in Appendix A.1. Our architecture follows Lotus-D [25] (discriminative variant) without its detail preserver.

Method	Semantic	Normal	Depth		Opt. Flow	Sc. Flow	Shading	Albedo	MTL Perf.
	mIoU %↑ Cityscapes	mAE °↓ DIODE	AbsRel %↓ KITTI	AbsRel %↓ DIODE	EPE-2D px↓ KITTI	EPE-3D m↓ KITTI	RMSE↓ MID	RMSE↓ MID	Δ_m% ↑ Average
Single-task baseline	48.17	22.27	14.21	32.56	10.36	0.2735	0.2145	0.2551	0.00
JTR [46]	45.75	40.23	26.39	66.39	n/a	n/a	n/a	n/a	–
JTR* [46]	20.46	50.91	39.27	73.14	34.92	0.5176	0.3030	0.3565	–106.87
DiffusionMTL [71]	30.09	35.64	21.10	45.19	n/a	n/a	n/a	n/a	–
DiffusionMTL* [71]	45.92	44.56	24.83	58.17	36.60	0.3502	0.3004	0.3660	–78.76
StableMTL-$\mathcal{S}$	52.57	23.94	15.64	33.36	12.76	0.2618	0.2310	0.2077	–1.57
StableMTL	55.79	23.27	14.98	33.03	10.76	0.2313	0.2346	0.2016	+4.78

Table 2: **Multi-task performance**. StableMTL significantly expands task coverage, addressing over twice the tasks of original baselines, while consistently outperforming them on individual tasks and multi-task metric. best and second-best are highlighted. * baselines adapted to full setting

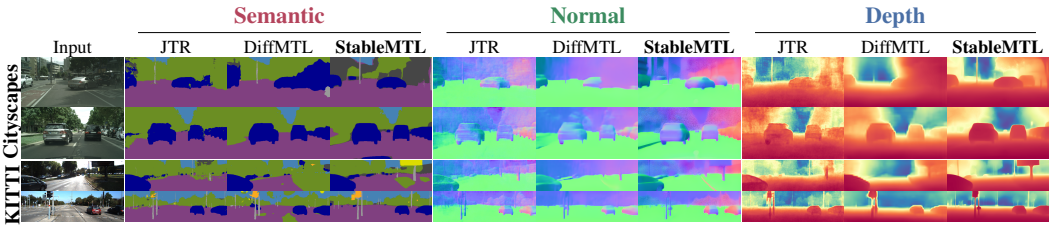


Figure 4: **Qualitative comparison.** We compare against original baseline versions as these are better performing (*cf*. Tab. 2) than the adapted full setting variants on the three tasks displayed. StableMTL demonstrates superior qualitative results.

To accommodate two-frame inputs, we triple the number of input channels in the first convolution of $U_{\phi,T}$. We initialize the expanded weights and dividing them by three as in [27]. We insert our task-attention layers at each of the 16 transformer blocks of $U_{\phi,T}$. We employ multi-head attention [59] to foster learning diversity. Our optimal model uses four attention heads, discussed in Sec. 4.2.

4.1 Main results

Tab. 2 compares our method with 4 baselines on 8 benchmarks across 7 tasks. When comparing StableMTL with the best baseline in each task, we observe a significant boost across all tasks, exemplified by improvements such as +9.87 mIoU (**semantic**), a -12.37 mAE (**normal**), and a -0.1189 EPE-3D (**scene flow**). Notably, our method improves the overall multi-task performance by +83.54 Δ_m. Furthermore, our single-stream version, StableMTL-$\mathcal{S}$, underperforms single-task baseline with a –1.57 Δ_m in overall MTL performance, our complete model, StableMTL, improves the overall performance by +4.78 Δ_m, showing the benefits of our multi-stream architecture and inter-task attention. We highlight that because DiffusionMTL and JTR use task-specific losses, they are difficult to scale architecturally and their training is unstable with a large number of tasks. Note how **normal** and **depth** degrade in the full setting (e.g., comparing JTR* and JTR). We further assess qualitative results in Fig. 4, demonstrating that StableMTL consistently outperforms the baselines. Additional results on real-world datasets [15, 42, 56, 64, 23, 65, 67, 48] are in Figs. 1 and 11, showing our method generalizes, in a zero-shot fashion, to diverse out-of-distribution domains.

4.2 Discussion

Impact of training datasets. Tab. 3 shows that removing individual datasets degrades performance, depending on domain similarity and annotation quality. For example, removing VKITTI 2 for flow tasks (row #2) leads to bigger performance degradation, especially on **sc. flow** and **opt. flow**, compared to removing FlyingThings3D (row #3), which shows VKITTI 2 better aligns with our evaluation domain. Removing Hypersim for geometrical tasks (row #5) drastically impacts **depth** and **normal**, especially on the DIODE dataset, which may result from its higher annotation quality and domain match. Interestingly, KITTI **depth** exhibits similar drops regardless of the dataset removed,

	Seg. & Sha. & Alb.	Opt.Flow & Sc.Flow		Normal & Depth		Semantic	Normal	Depth		Opt. Flow	Sc. Flow	Shading	Albedo	MTL Perf.
	Hypersim & Virtual KITTI 2	Virtual KITTI 2	Flying Things 3D	Virtual KITTI 2	Hypersim	mIoU %↑ Cityscapes	mAE °↓ DIODE	AbsRel %↓ KITTI	AbsRel %↓ DIODE	EPE-2D px↓ KITTI	EPE-3D m↓ KITTI	RMSE↓ MID	RMSE↓ MID	Δ_m % ↑ Avg
#1	✓	✓	✓	✓	✓	55.08	23.36	14.03	32.97	11.22	0.2297	0.2350	0.2061	+4.14
#2	✓	-	✓	✓	✓	53.47	23.69	13.74	33.54	24.15	0.4126	0.2367	0.2103	–24.30
#3	✓	✓	-	✓	✓	55.28	22.29	14.82	33.21	13.60	0.2541	0.2383	0.2012	+0.01
#4	✓	✓	✓	-	✓	44.55	23.79	17.98	34.13	11.99	0.2456	0.2236	0.2110	–3.10
#5	✓	✓	✓	✓	-	50.12	52.65	18.26	45.27	12.40	0.2484	0.2279	0.2107	–23.46

Table 3: **Ablation study of training datasets.** Removing task supervision from any dataset degrades performance on the corresponding tasks (*cf*. text for details). We highlight best and second-best.

Method	Semantic	Normal	Depth		Opt. Flow	Scene Flow	Shading	Albedo	MTL Perf.
	mIoU %↑ Cityscapes	mAE °↓ DIODE	AbsRel %↓ KITTI	AbsRel %↓ DIODE	EPE-2D px↓ KITTI	EPE-3D m↓ KITTI	RMSE↓ MID	RMSE↓ MID	Δ_m % ↑ Avg
StableMTL(ρ=0)	54.90	22.88	14.90	32.57	11.45	0.2400	0.2351	0.2023	+3.41
w/o separate (q_t, k_t, v_t)	48.38	24.15	16.17	33.24	11.06	0.2327	0.2295	0.2066	+0.85
w/o single-stream init.	44.69	23.35	15.60	33.21	12.17	0.2647	0.2325	0.2110	–3.11
w/o multi-stream	52.57	23.94	15.64	33.36	12.76	0.2618	0.2310	0.2077	–1.57

Table 4: **Multi-stream ablation.** Each design choice contributes to the best performance.

suggesting that VKITTI 2's domain proximity offsets its lower annotation quality, while Hypersim's quality alone cannot overcome the larger domain gap.

Multi-stream ablation. In Tab. 4, we conduct an ablation study on various design choices of StableMTL. Removing separate projection layer per task *i.e.*, "w/o separate (q_t, k_t, v_t)" improves performance on certain tasks like optical flow, scene flow, and shading, but degrades others and leads to an overall multi-task drop of 2.56 in Δ_m. This might be attributed to the highly similar attention patterns across tasks illustrated in Fig. 10, which limits the learned inter-task interaction. Initializing U_ϕ with Stable Diffusion weights instead of U_θ *i.e.*, "w/o single-stream init.", significantly degrades performance by 6.52 in Δ_m. We also report using only single-stream *i.e.*, "w/o multi-stream" also degrading performance on all tasks (drop of 4.98 in Δ_m), since the single stream does not benefit from cross-task exchanges. Additionally, we ablate our multi-frame adaptation on the single-stream only for single-input tasks, with ours being $z_{i,j}$=`concat`(z_i, z_j), compared to $z_{i,j}$=`avg`(z_i, z_j), and $z_{i,j}$=`concat`$(z_i, \emptyset)$; they perform -1.57, -4.11, and -2.72 in Δ_m, respectively.

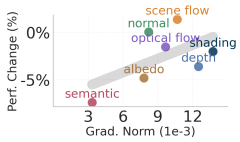
Benefit of task-gradient isolation. Figure 5a demonstrates the efficacy of our isolation scheme, which improves overall performance by 2.54 Δ_m. In particular, it drastically improves semantic and albedo, which have lower gradient norm as seen in Fig. 5b. The latter further reveals an inverse relationship between gradient norms and the performance drop when removing our isolation scheme. Without this scheme, tasks with smaller gradient exhibit large performance drops.

Effects of task attention heads. Varying task attention to use 1/2/4/8 heads leads to an overall performance of $+2.45$/$+4.14$/$+\mathbf{4.78}$/$+4.41$ in Δ_m, respectively. As is evident, although more heads promote greater attention, a plateau is quickly reached, likely due to the reduced capacity of each head. Notably, all outperform the single-stream StableMTL-$\mathcal{S}$, which achieved -1.57 Δ_m.

Impact of training tasks. Figure 6 shows performance on semantic and normal when training on different set tasks w.r.t. single-task performance. Adding new tasks to the pipeline affects single-task

Method	Semantic	Normal	Depth		Opt. Flow	Scene Flow	Shading	Albedo	MTL Perf.
	mIoU %↑ Cityscapes	mAE °↓ DIODE	AbsRel %↓ KITTI	AbsRel %↓ DIODE	EPE-2D px↓ KITTI	EPE-3D m↓ KITTI	RMSE↓ MID	RMSE↓ MID	Δ_m % ↑ Avg
StableMTL-$\mathcal{S}$	52.57	23.94	15.64	33.36	12.76	0.2618	0.2310	0.2077	–1.57
w/o Task-gradient isolation	48.68	23.93	15.52	34.82	12.96	0.2583	0.2379	0.2178	–4.11

(a) Ablation of task-grad. isolation training scheme.



(b) Observed correlation

Figure 5: **Task-gradient isolation strategy.** In (a), we report the performance w/ and w/o our isolation strategy, showing that it drastically benefits some tasks (*e.g.*, semantic) and improves the overall Δ_m metric. (b) shows that when removing gradient isolation, tasks with smaller gradient magnitudes are overwhelmed by those with larger ones, leading to a significant performance drop.

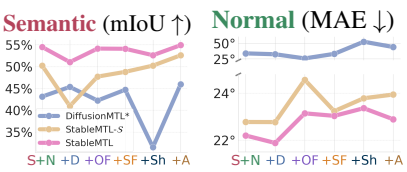


Figure 6: **Tasks ablation.** Separate trainings with increasing task counts.

Figure 7: **Task attention scores.** Strong interactions are observed not only among tasks with known mutual benefits but also one-way interactions, as detailed in the text.

Strategy	Prob. (ρ)	Semantic	Normal	Depth		Opt. Flow	Scene Flow	Shading	Albedo	MTL Perf.
		mIoU %↑	mAE °↓	AbsRel %↓	AbsRel %↓	EPE-2D px↓	EPE-3D m↓	RMSE↓	RMSE↓	Δ_m% ↑
		Cityscapes	DIODE	KITTI	DIODE	KITTI	KITTI	MID	MID	Avg
Sample(π_T)	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
	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 5: **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.

performance, which is a clear issue for StableMTL-$\mathcal{S}$ but somehow is mitigated by the multi-stream architecture of StableMTL thanks to its task-attention mechanism leveraging inter-task information.

Visualization of task attentions. Figure 7 shows the task attention scores from main stream tasks (columns) to auxiliary stream tasks (represented by colored bars) at selected layers: here, layer 12 and 15 – more layers shown in Appendix A.5). The observed patterns reveal strong mutually interactions like **normal**/**depth**, **optical flow**/**scene flow**, and **shading**/**albedo**, but also that some tasks are one-way beneficiaries of others such as **optical flow** using **semantic** and **semantic** using **shading**. Notably, **shading** and **albedo**, capturing lighting and material properties, respectively, can assist other tasks. These interactions support the findings of a prior work [74].

Effect of task attention masking. We ablate the task attention masking in Tab. 5. 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.

Limitations. We observe that despite enhancing multi-task capabilities, StableMTL performance on some individual tasks lags behind single-task models. Furthermore, adding some new tasks (*e.g.*, shading) affects the individual performances on other tasks as evidenced in Fig. 6, showing that cross-task interaction is far from optimal yet. In terms of inference, multi-task prediction is currently sequential. We believe this is a constraint that future work could address with models enabling simultaneous multi-output prediction via improved multi-token conditioning. Additionally, the uniform task sampling strategy could be refined through adaptive approaches.

5 Conclusion

This work extends the task of partially supervised MTL to a more practical setting where the model is trained on multiple synthetic datasets with partial labels. To bridge the synthetic-to-real gap, we adapt a single-step latent diffusion model with multi-task encoding and task-gradient isolation. While this model is capable of multi-tasking, it suffers from degraded overall performance. To overcome this limitation, we introduce a multi-stream architecture with task attention, allowing the model to leverage inter-task relationships and improve multi-task performance. Our approach not only scales to more tasks, it also achieves significant results on unseen data.

References

[1] Aich, A., Schulter, S., Roy-Chowdhury, A.K., Chandraker, M., Suh, Y.: Efficient controllable multi-task architectures. In: ICCV (2023)

[2] Aleotti, F., Poggi, M., Mattoccia, S.: Learning optical flow from still images. In: CVPR (2021)

[3] Argaw, D.M., Kim, J., Rameau, F., Cho, J.W., Kweon, I.S.: Optical flow estimation from a single motion-blurred image. In: AAAI (2021)

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

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

[6] Borse, S., Das, D., Park, H., Cai, H., Garrepalli, R., Porikli, F.: Dejavu: Conditional regenerative learning to enhance dense prediction. In: CVPR (2023)

[7] Brüggemann, D., Kanakis, M., Obukhov, A., Georgoulis, S., Van Gool, L.: Exploring relational context for multi-task dense prediction. In: ICCV (2021)

[8] Cabon, Y., Murray, N., Humenberger, M.: Virtual kitti 2. In: arXiv (2020)

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

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

[11] Chen, T., Chen, X., Du, X., Rashwan, A., Yang, F., Chen, H., Wang, Z., Li, Y.: AdaMV-MoE: Adaptive multi-task vision mixture-of-experts. In: ICCV (2023)

[12] Chen, Z., Badrinarayanan, V., Lee, C.Y., Rabinovich, A.: Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In: ICML (2018)

[13] Chen, Z., Ngiam, J., Huang, Y., Luong, T., Kretzschmar, H., Chai, Y., Anguelov, D.: Just pick a sign: Optimizing deep multitask models with gradient sign dropout. In: NeurIPS (2020)

[14] Choi, W., Im, S.: Dynamic neural network for multi-task learning searching across diverse network topologies. In: CVPR (2023)

[15] 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)

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

[17] Fan, Z., Sarkar, R., Jiang, Z., Chen, T., Zou, K., Cheng, Y., Hao, C., Wang, Z., et al.: M^3VIT: Mixture-of-experts vision transformer for efficient multi-task learning with model-accelerator co-design. In: NeurIPS (2022)

[18] Fontana, M., Spratling, M., Shi, M.: When multitask learning meets partial supervision: A computer vision review. Proceedings of the IEEE (2024)

[19] Fu, X., Yin, W., Hu, M., Wang, K., Ma, Y., Tan, P., Shen, S., Lin, D., Long, X.: Geowizard: Unleashing the diffusion priors for 3d geometry estimation from a single image. In: ECCV (2024)

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

[21] Ghiasi, G., Zoph, B., Cubuk, E.D., Le, Q.V., Lin, T.Y.: Multi-task self-training for learning general representations. In: ICCV (2021)

[22] Gui, M., Schusterbauer, J., Prestel, U., Ma, P., Kotovenko, D., Grebenkova, O., Baumann, S.A., Hu, V.T., Ommer, B.: DepthFM: Fast monocular depth estimation with flow matching. In: AAAI (2025)

[23] Guizilini, V., Ambrus, R., Pillai, S., Raventos, A., Gaidon, A.: 3d packing for self-supervised monocular depth estimation. In: CVPR (2020)

[24] Guo, P., Lee, C.Y., Ulbricht, D.: Learning to branch for multi-task learning. In: ICML (2020)

[25] He, J., Li, H., Yin, W., Liang, Y., Li, L., Zhou, K., Liu, H., Liu, B., Chen, Y.C.: Lotus: Diffusion-based visual foundation model for high-quality dense prediction. In: ICLR (2025)

[26] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016)

[27] 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)

[28] Ke, B., Qu, K., Wang, T., Metzger, N., Huang, S., Li, B., Obukhov, A., Schindler, K.: Marigold: Affordable adaptation of diffusion-based image generators for image analysis. T-PAMI (2025)

[29] Kendall, A., Gal, Y., Cipolla, R.: Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In: CVPR (2018)

[30] Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: ICLR (2015)

[31] Le, D.H., Pham, T., Lee, S., Clark, C., Kembhavi, A., Mandt, S., Krishna, R., Lu, J.: One diffusion to generate them all. In: CVPR (2025)

[32] Li, W.H., Liu, X., Bilen, H.: Learning multiple dense prediction tasks from partially annotated data. In: CVPR (2022)

[33] Liang, X., Wu, Y., Han, J., Xu, H., Xu, C., Liang, X.: Effective adaptation in multi-task co-training for unified autonomous driving. In: NeurIPS (2022)

[34] Lin, X., Zhen, H.L., Li, Z., Zhang, Q.F., Kwong, S.: Pareto multi-task learning. In: NeurIPS (2019)

[35] Liu, Q., Liao, X., Carin, L.: Semi-supervised multitask learning. In: NeurIPS (2007)

[36] Liu, Y.C., Ma, C.Y., Tian, J., He, Z., Kira, Z.: Polyhistor: Parameter-efficient multi-task adaptation for dense vision tasks. In: NeurIPS (2022)

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

[38] Lu, Y., Pirk, S., Dlabal, J., Brohan, A., Pasad, A., Chen, Z., Casser, V., Angelova, A., Gordon, A.: Taskology: Utilizing task relations at scale. In: CVPR (2021)

[39] Lu, Y., Kumar, A., Zhai, S., Cheng, Y., Javidi, T., Feris, R.: Fully-adaptive feature sharing in multi-task networks with applications in person attribute classification. In: CVPR (2017)

[40] Martin Garcia, G., Abou Zeid, K., Schmidt, C., de Geus, D., Hermans, A., Leibe, B.: Fine-tuning image-conditional diffusion models is easier than you think. In: WACV (2025)

[41] Mayer, N., Ilg, E., Häusser, P., Fischer, P., Cremers, D., Dosovitskiy, A., Brox, T.: A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In: CVPR (2016)

[42] Menze, M., Geiger, A.: Object scene flow for autonomous vehicles. In: CVPR (2015)

[43] Misra, I., Shrivastava, A., Gupta, A., Hebert, M.: Cross-stitch networks for multi-task learning. In: CVPR (2016)

[44] Momma, M., Dong, C., Liu, J.: A multi-objective/multi-task learning framework induced by pareto stationarity. In: ICML (2022)

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

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

[47] Ouali, Y., Hudelot, C., Tami, M.: Semi-supervised semantic segmentation with cross-consistency training. In: CVPR (2020)

[48] Perazzi, F., Pont-Tuset, J., McWilliams, B., Gool, L.V., Gross, M., Sorkine-Hornung, A.: A benchmark dataset and evaluation methodology for video object segmentation. In: CVPR (2016)

[49] Roberts, M., Ramapuram, J., Ranjan, A., Kumar, A., Bautista, M.A., Paczan, N., Webb, R., Susskind, J.M.: Hypersim: A photorealistic synthetic dataset for holistic indoor scene understanding. In: ICCV (2021)

[50] Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. In: CVPR (2022)

[51] Ruder, S., Bingel, J., Augenstein, I., Søgaard, A.: Latent multi-task architecture learning. In: AAAI (2019)

[52] Saha, S., Obukhov, A., Paudel, D.P., Kanakis, M., Chen, Y., Georgoulis, S., Van Gool, L.: Learning to relate depth and semantics for unsupervised domain adaptation. In: CVPR (2021)

[53] Senushkin, D., Patakin, N., Kuznetsov, A., Konushin, A.: Independent component alignment for multi-task learning. In: CVPR (2023)

[54] Standley, T., Zamir, A.R., Chen, D., Guibas, L., Malik, J., Savarese, S.: Which tasks should be learned together in multi-task learning? In: ICML (2020)

[55] Sun, K., Xiao, B., Liu, D., Wang, J.: Deep high-resolution representation learning for human pose estimation. In: CVPR (2019)

[56] Sun, P., Kretzschmar, H., Dotiwalla, X., Chouard, A., Patnaik, V., Tsui, P., Guo, J., Zhou, Y., Chai, Y., Caine, B., Vasudevan, V., Han, W., Ngiam, J., Zhao, H., Timofeev, A., Ettinger, S., Krivokon, M., Gao, A., Joshi, A., Zhang, Y., Shlens, J., Chen, Z., Anguelov, D.: Scalability in perception for autonomous driving: Waymo open dataset. In: CVPR (2020)

[57] Vandenhende, S., Georgoulis, S., Van Gansbeke, W., Proesmans, M., Dai, D., Van Gool, L.: Multi-task learning for dense prediction tasks: A survey. T-PAMI (2022)

[58] 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. CoRR (2019)

[59] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: NeurIPS (2017)

[60] Wang, F., Wang, X., Li, T.: Semi-supervised multi-task learning with task regularizations. In: ICDM (2009)

[61] Wang, Y., Tsai, Y.H., Hung, W.C., Ding, W., Liu, S., Yang, M.H.: Semi-supervised multi-task learning for semantics and depth. In: WACV (2022)

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

[63] Wang, Z., Li, H., Sui, L., Zhou, T., Jiang, H., Nie, L., Liu, S.: StableMotion: Repurposing diffusion-based image priors for motion estimation. In: arXiv (2025)

[64] Wilson, B., Qi, W., Agarwal, T., Lambert, J., Singh, J., Khandelwal, S., Pan, B., Kumar, R., Hartnett, A., Pontes, J.K., Ramanan, D., Carr, P., Hays, J.: Argoverse 2: Next generation datasets for self-driving perception and forecasting. In: NeurIPS (2021)

[65] Xiao, P., Shao, Z., Hao, S., Zhang, Z., Chai, X., Jiao, J., Li, Z., Wu, J., Sun, K., Jiang, K., Wang, Y., Yang, D.: PandaSet: Advanced sensor suite dataset for autonomous driving. In: ITSC (2021)

[66] Xu, G., Ge, Y., Liu, M., Fan, C., Xie, K., Zhao, Z., Chen, H., Shen, C.: What matters when repurposing diffusion models for general dense perception tasks? In: ICLR (2025)

[67] Xu, N., Yang, L., Fan, Y., Yang, J., Yue, D., Liang, Y., Price, B., Cohen, S., Huang, T.: Youtube-vos: Sequence-to-sequence video object segmentation. In: ECCV (2018)

[68] Xu, X., Zhao, H., Vineet, V., Lim, S.N., Torralba, A.: Mtformer: Multi-task learning via transformer and cross-task reasoning. In: ECCV (2022)

[69] Ye, C., Qiu, L., Gu, X., Zuo, Q., Wu, Y., Dong, Z., Bo, L., Xiu, Y., Han, X.: StableNormal: Reducing diffusion variance for stable and sharp normal. ACM TOG (2024)

[70] Ye, H., Xu, D.: Inverted pyramid multi-task transformer for dense scene understanding. In: ECCV (2022)

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

[72] Yu, T., Kumar, S., Gupta, A., Levine, S., Hausman, K., Finn, C.: Gradient surgery for multi-task learning. In: NeurIPS (2020)

[73] Zamir, A.R., Sax, A., Cheerla, N., Suri, R., Cao, Z., Malik, J., Guibas, L.J.: Robust learning through cross-task consistency. In: CVPR (2020)

[74] Zamir, A.R., Sax, A., Shen, W.B., Guibas, L., Malik, J., Savarese, S.: Taskonomy: Disentangling task transfer learning. In: CVPR (2018)

[75] Zeng, Z., Deschaintre, V., Georgiev, I., Hold-Geoffroy, Y., Hu, Y., Luan, F., Yan, L.Q., Hašan, M.: RGB$\leftrightarrow$X: Image decomposition and synthesis using material- and lighting-aware diffusion models. In: SIGGRAPH (2024)

[76] Zhang, Y., Yeung, D.Y.: Semi-supervised multi-task regression. In: Buntine, W., Grobelnik, M., Mladenić, D., Shawe-Taylor, J. (eds.) ECML PKDD (2009)

[77] Zhang, Z., Cui, Z., Xu, C., Jie, Z., Li, X., Yang, J.: Joint task-recursive learning for semantic segmentation and depth estimation. In: ECCV (2018)

[78] Zhao, C., Liu, M., Zheng, H., Zhu, M., Zhao, Z., Chen, H., He, T., Shen, C.: DICEPTION: A generalist diffusion model for visual perceptual tasks. In: arXiv (2025)

Acknowledgments. This work was funded by the French Agence Nationale de la Recherche (ANR) with project SIGHT (ANR-20-CE23-0016) and performed with HPC resources from GENCI-IDRIS (Grants AD011014102R2, AD011014102R1, AD011014389R1, and AD011012808R3). The authors are also grateful for the resources provided by the CLEPS infrastructure from Inria Paris.

Broader impact and ethics. Training models for joint multi-task prediction with partially labeled synthetic data substantially lowers annotation costs, benefiting numerous applications and advancing research through its comprehensive outputs. Despite these benefits, prediction errors raise ethical concerns, particularly in high-stakes areas like autonomous driving where failures can be fatal, mandating robust safety measures and redundancy. Multi-task learning's nature, however, might provide resilience by allowing tasks to offset each other's errors.

A Appendix

A.1 Task encoding

For unbounded quantities – **depth**, **optical flow**, and **scene flow** – we adopt affine-invariant representations as in [27]. 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 [10], 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.

A.2 Single stream architecture

For clarity, our single-stream architecture (*cf*. Sec. 3.1) is depicted in Fig. 8. We fine-tune the UNet ($U_{\theta,\tau}$) to predict annotation latents from input image latents. This prediction is conditioned on task tokens, which are sampled using our custom training scheme to isolate task gradients.

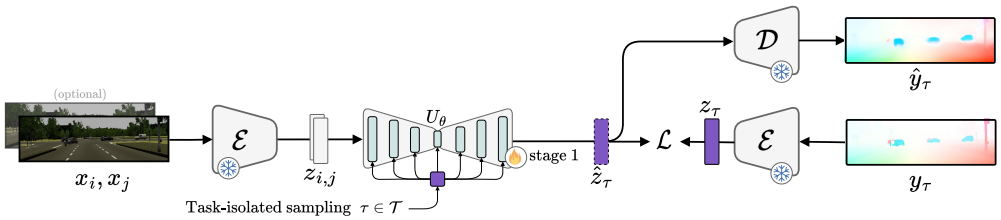


Figure 8: **Single-stream architecture.** During stage 1 (Sec. 3.1), we fine-tune a UNet ($U_{\theta,\tau}$) to perform latent regression. It is then used during stage 2 (Sec. 3.2) as an auxiliary stream to provide task features.

We use arbitrary text prompts to identify each task, $\texttt{[prompt]}_\tau \in \{\texttt{"normal"}, \texttt{"depth"}, \dots\}$. Task prompts are passed through a CLIP text encoder to retrieve their corresponding task tokens: $c_\tau = \text{CLIP}(\texttt{[prompt]}_\tau)$. We inject these embeddings to condition the models in Sec. 3.1 and Sec. 3.2.

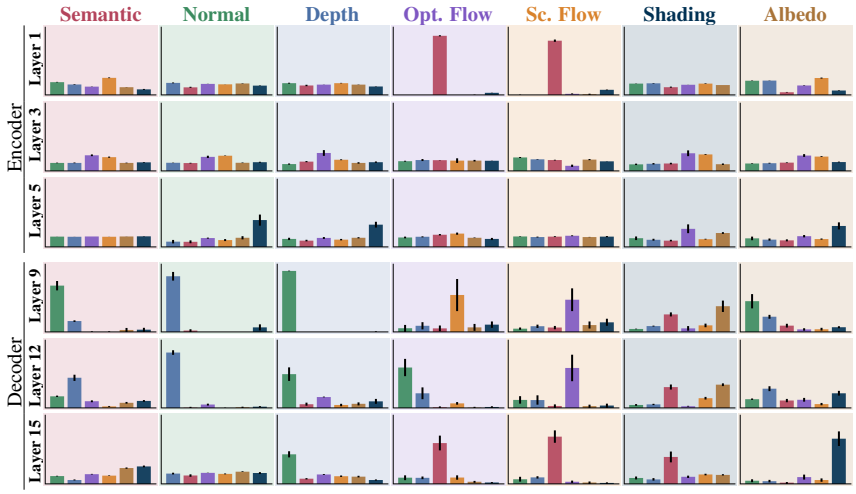


Figure 9: **Task attention scores in the U-Net ($U_{\phi,T}$)** (last layer of each encoder/decoder block shown). Attention becomes more peaky in deeper layers and highlights beneficial cross-task relationships.

A.3 Training details

For our method, the single stream UNet U_θ is initialized with weights from Stable Diffusion v2 [50] and trained for 20,000 steps (8 hours). The main stream UNet U_ϕ trains for another 10,000 steps (12 hours). For both models, we use Adam optimizer [30] with a learning rate of $1.0\text{e}{-4}$ and employ an effective batch size of 32 with two gradient accumulation steps on a single H100 GPU.

For adapted baselines, we use the appropriate losses on the added tasks. Considering tasks are mixed inside a mini-batch, we sample them as uniformly as possible across datasets. We use the same augmentations and image resolutions as for training StableMTL. For DiffusionMTL*, we change the ResNet-50 backbone to the stronger ResNet-101 [26]. In JTR*, the joint task space and alignment losses are expanded to account for the additional tasks.

A.4 Evaluation datasets

For **depth** estimation, we evaluate on the KITTI Eigen test split [20, 16] (652 urban images with sparse LiDAR depth) and the DIODE validation split [58] (325 indoor and 446 outdoor LiDAR depth maps); this DIODE split is also utilized for evaluating surface **normal** estimation. For **semantic** segmentation, evaluation is on the Cityscapes validation set [15] (500 images, as test labels are withheld). **Optical Flow** and **scene flow** performance is reported on 200 training images from the KITTI Flow 2025 benchmark [42] (test split is hidden) with semi-automatically established ground truth. Finally, for **shading** and **albedo**, we use the MIT Multi-Illumination dataset's [45] test set of 750 samples, with ground-truth provided by MIDIntrinsics [9]. For qualitative evaluation, we also employ six additional real-world datasets spanning diverse domains [65, 64, 56, 23, 67, 48].

A.5 Visualization of task attention on more layers

Figure 9 illustrates the task attention scores across multiple attention layers of the U-Net ($U_{\phi,T}$), showing scores from the last attention layer in each encoder and decoder block. We observe that attention becomes more "peaky" in deeper layers of the network, while it is more homogeneous in the initial layers. This trend is likely because deeper layers encode richer, higher-level features, containing more task-specific information.

StableMTL demonstrates effective attention allocation between known mutually beneficial task pairs, such as **depth**/**normal**, **optical flow**/**scene flow**, and **shading**/**albedo**. Furthermore, it leverages synergies with other tasks. For instance, **semantic** segmentation appears to benefit from **scene flow**/**optical flow**, potentially helping in differing dynamic and static classes. Similarly, **normal** and **depth** estimation benefit from lighting information provided by **shading**, as lighting conditions can significantly influence the perception of **depth** and surface **normal**.

A.6 Task attention without separate projection layers (q_t, k_t, v_t)

Fig. 10 highlights that sharing projection layers across tasks results in highly repetitive attention score patterns. Such patterns may contribute to a decline in multi-task performance, a shown by the row "w/o separate (q_t, k_t, v_t)" in Tab. 4.

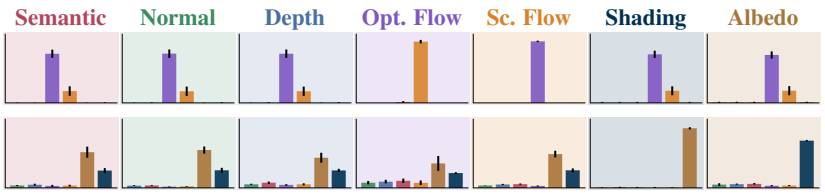


Figure 10: **StableMTL w/o separate** (q_t, k_t, v_t). Without separate projection layers (q_t, k_t, v_t), the attention pattern is highly repetitive across tasks.

A.7 More qualitative results

Fig. 11 presents additional qualitative results on real-world data. Notably, StableMTL demonstrates generalization to diverse, multi-task real-world scenarios, even when trained on partially labeled synthetic datasets.

B.1 Additional metrics

We present IoU scores for each mapped class on the Cityscapes dataset [15] in Tab. 6, with class mapping details provided in Tab. 9. Overall, StableMTL ranks either first or second on all classes.

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 [46]	96.33	51.48	15.33	0.87	3.22	61.28	77.36	60.16	45.75
JTR* [46]	71.50	0.00	0.00	0.00	0.00	38.69	53.47	0.00	20.46
DiffusionMTL [71]	85.17	29.60	6.64	1.14	5.19	41.05	27.34	44.57	30.09
DiffusionMTL* [71]	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 6: **Per-class Semantic performance**. We report the IoU per mapped semantic class as well as the average IoU (mIoU), detailed in Tab. 9, on Cityscapes.

For **depth** evaluation on KITTI [20] and DIODE [58], additional depth metrics are reported in Tab. 7; these include the following commonly used metrics [16]: absolute relative error (Abs Rel), squared relative error (Sq Rel), root mean squared error (RMSE), mean $\log_{10}$ error (RMSE log), and threshold accuracies ($\delta 1$, $\delta 2$, $\delta 3$). 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. 8, metrics for **shading** and **albedo** on the MID dataset [45] following [9]: structural similarity index (SSIM) [62], local mean squared error (LMSE), and root mean squared error (RMSE). These results show our method outperforms all other MTL baselines.

B.2 Visualization of task-specific outputs

We detail here the visualization for each task. **Semantic** segmentation maps employ the Cityscapes color scheme (see Tab. 9). For **optical flow** $\mathbf{v}(u, v) = (v_x, v_y)$, we adopt the HSV color encoding from [5], where the lateral flow vector's (v_x, v_y) angle $\text{atan2}(-v_y, -v_x)$ 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: $\text{atan2}(-v_y, -v_x)$) 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

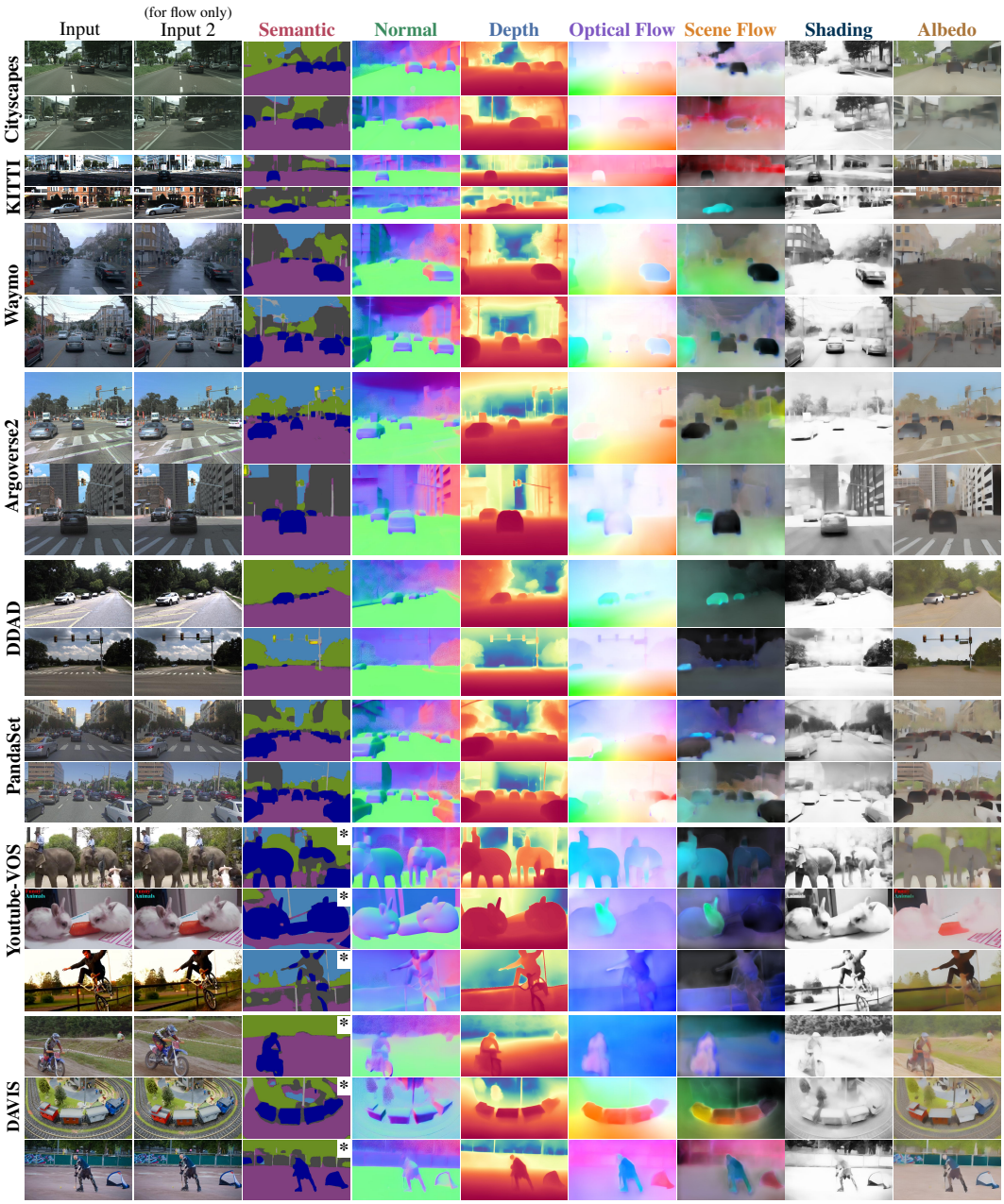


Figure 11: **Additional qualitative results on real-world data.** Despite being trained on partially annotated synthetic datasets, StableMTL demonstrates generalization to multi-task real-world scenarios. * Note that **semantic** is trained on driving classes and is not expected to generalize to unseen classes.

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 [4]. **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.

B.3 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 [37], detailed in Tab. 9.

Method	KITTI							DIODE						
	Abs Rel↓	Sq Rel↓	RMSE↓	RMSE log↓	$\delta 1$↑	$\delta 2$↑	$\delta 3$↑	Abs Rel↓	Sq Rel↓	RMSE↓	RMSE log↓	$\delta 1$↑	$\delta 2$↑	$\delta 3$↑
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 [46]	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* [46]	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 [71]	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* [71]	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 7: **Additional depth metrics.** In all but one metric, StableMTL ranks first on both KITTI [20] and DIODE [58].

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* [46]	0.3030	0.1843	0.0530	0.3567	0.3102	0.0994
DiffusionMTL* [71]	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 8: **Additional shading and albedo metrics.** On MIDIntrinsics [45], StableMTL ranks either first or second; it is only surpassed by our single stream variant StableMTL-$\mathcal{S}$.

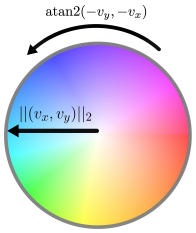
B.4 Image resolutions for training and evaluation

We train the model on datasets using the following pixel resolutions, `height`×`width`:

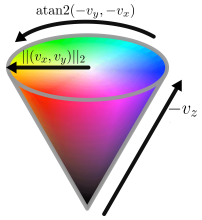
- Hypersim (288×384),
- FlyingThings3D (268×480),
- VKITTI2 (187×621).

For evaluation, we use the following resolutions:

- Cityscapes (256×512),
- DIODE (384×512),
- KITTI (176×608),
- MIDIntrinsics (256×384).



(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**.

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