MTMamba++: Enhancing Multi-Task Dense Scene Understanding via Mamba-Based Decoders

Baijiong Lin, Weisen Jiang, Pengguang Chen, Shu Liu, *Member, IEEE*, and Ying-Cong Chen, *Member, IEEE*

***Abstract*—Multi-task dense scene understanding, which trains a model for multiple dense prediction tasks, has a wide range of application scenarios. Capturing long-range dependency and enhancing cross-task interactions are crucial to multi-task dense prediction. In this paper, we propose MTMamba++, a novel architecture for multi-task scene understanding featuring with a Mamba-based decoder. It contains two types of core blocks: self-task Mamba (STM) block and cross-task Mamba (CTM) block. STM handles long-range dependency by leveraging state-space models, while CTM explicitly models task interactions to facilitate information exchange across tasks. We design two types of CTM block, namely F-CTM and S-CTM, to enhance cross-task interaction from feature and semantic perspectives, respectively. Extensive experiments on NYUDv2, PASCAL-Context, and Cityscapes datasets demonstrate the superior performance of MTMamba++ over CNN-based, Transformer-based, and diffusion-based methods while maintaining high computational efficiency.**

***Index Terms*—Multi-task learning, dense scene understanding, Mamba.**

I. Introduction

MULTI-TASK dense scene understanding, which trains a single model to simultaneously handle multiple pixel-wise prediction tasks (e.g., semantic segmentation, depth estimation, surface normal estimation, and object boundary detection), has become increasingly important in many computer vision applications [1], such as autonomous driving [2], healthcare [3], and robotics [4]. The success of multi-task dense prediction hinges on addressing two fundamental challenges:

1) modeling long-range spatial relationships to capture global context information, which is essential for pixel-wise prediction tasks;
2) enhancing cross-task interactions to facilitate knowledge sharing among different tasks, which is crucial to multi-task learning.

Existing multi-task dense prediction approaches can be broadly categorized by their architectural design. CNN-based methods [5], [6] employ convolutional operations in decoders for task-specific predictions but primarily capture local features, struggling with modeling long-range dependencies and global context understanding [7], [8]. Transformer-based methods [9], [10], [11], [12], [13], [14] employ attention mechanisms [15] to better capture global context and demonstrate improved performance. However, they suffer from quadratic computational complexity with respect to sequence length [16], [17], making them computationally prohibitive for high-resolution dense prediction tasks.

To address these limitations, we propose MTMamba++, a novel Mamba-based architecture that achieves effective and efficient multi-task dense scene understanding. MTMamba++ introduces two key components based on state space models (SSMs) [18], [19] in the decoder:

1) The self-task Mamba (STM) block, inspired by [20], captures global context information for each task by leveraging the long-range modeling capabilities of SSMs with linear computational complexity;
2) The cross-task Mamba (CTM) block enables effective knowledge exchange across tasks through two variants: F-CTM for feature-level interaction and S-CTM for semantic-level interaction. The S-CTM introduces a novel cross SSM (CSSM) mechanism that models relationships between task-specific and shared feature sequences, providing more effective task interaction than simple feature fusion approaches used in F-CTM.

As the overall framework shown in Fig. 1, MTMamba++ features a three-stage Mamba-based decoder that progressively refines multi-task predictions. Each stage contains an ECR (expand, concatenate, and reduce) block that upscales features and fuses them with encoder features, followed by STM and CTM blocks for task-specific learning and cross-task interaction. This

Received 27 August 2024; revised 29 May 2025; accepted 26 July 2025. Date of publication 29 July 2025; date of current version 3 October 2025. This work was supported by Guangzhou-HKUST(GZ) Joint Funding Scheme under Grant 2024A03J0241, and in part by the Guangdong Provincial Key Lab of Integrated Communication, Sensing and Computation for Ubiquitous Internet of Things under Grant 2023B1212010007. Recommended for acceptance by S. J. Pan. *(Corresponding authors: Shu Liu; Ying-Cong Chen.)*

Baijiong Lin is with the Artificial Intelligence Thrust, The Hong Kong University of Science and Technology (Guangzhou), Guangzhou 511458, China, and also with HKUST(GZ) - SmartMore Joint Lab, Hong Kong (e-mail: bj.lin.email@gmail.com).

Weisen Jiang is with the Department of Computer Science and Engineering, The Hong Kong University of Science and Technology,, Hong Kong SAR, China (e-mail: waysonkong@gmail.com).

Pengguang Chen and Shu Liu are with SmartMore Corporation Limited, Shenzhen 518052, China (e-mail: akuxcw@gmail.com; liushuhust@gmail.com).

Ying-Cong Chen is with the Artificial Intelligence Thrust, The Hong Kong University of Science and Technology (Guangzhou), Guangzhou 511458, China, also with HKUST(GZ) - SmartMore Joint Lab, Hong Kong, and also with the Department of Computer Science and Engineering, Hong Kong University of Science and Technology,, Hong Kong SAR, China (e-mail: yingcongchen@ust.hk).

The code is available at https://github.com/EnVision-Research/MTMamba.

Digital Object Identifier 10.1109/TPAMI.2025.3593621

0162-8828 © 2025 IEEE. All rights reserved, including rights for text and data mining, and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: Chinese University of Hong Kong. Downloaded on October 13,2025 at 02:58:43 UTC from IEEE Xplore. Restrictions apply.

Fig. 1. Overview of the general architecture for MTMamba++ and its preliminary version MTMamba [24], presenting with semantic segmentation (abbreviated as "Semseg") and depth estimation (abbreviated as "Depth") tasks. The pretrained encoder (Swin-Large Transformer [25] is used here) is responsible for extracting multi-scale generic visual representations from the input RGB image. In the decoder, the ECR (expand, concatenate, and reduce) block is designed to upsample the feature maps and fuse them with high-level features derived from the encoder. Following this, the task-specific representations captured by the self-task Mamba (STM) blocks are further refined in the cross-task Mamba (CTM) block. This process ensures that each task benefits from the comprehensive feature set provided by the shared and task-specific components. Each task has its own head to generate the final predictions. We develop two types of CTM blocks and prediction heads, respectively. MTMamba++ and MTMamba utilize different CTM blocks and prediction heads as their default configurations. The details of each part are comprehensively introduced in Section III.

design effectively captures long-range dependencies and enhances cross-task interaction while maintaining computational efficiency.

We evaluate MTMamba++ on three standard multi-task dense prediction benchmark datasets, namely NYUDv2 [21], PASCAL-Context [22], and Cityscapes [23]. Quantitative results demonstrate that MTMamba++ significantly surpasses previous methods, including CNN-based, Transformer-based, and diffusion-based appoarch. Moreover, comprehensive efficiency analysis shows that MTMamba++ achieves state-of-the-art performance while maintaining high computational efficiency. Notably, our experiments demonstrate that SSM-based architectures are more effective and efficient than attention-based for multi-task dense prediction tasks. Additionally, qualitative studies show that MTMamba++ generates superior visual results with greater accuracy in detail, sharper boundaries, and more accurate detection in small objects compared to existing approaches.

In summary, the main contributions of this paper are four-fold:

- We propose MTMamba++, a novel multi-task architecture based on state space models (SSMs) for multi-task dense scene understanding. It contains a novel Mamba-based decoder, effectively modeling long-range spatial relationships and achieving cross-task correlation;
- In the decoder, we design two types of cross-task Mamba (CTM) blocks, namely F-CTM and S-CTM, to enhance cross-task interaction from feature and semantic perspectives, respectively;
- In the S-CTM block, we propose a novel cross SSM (CSSM) to model the relationship between two sequences based on the SSM mechanism;
- We evaluate MTMamba++ on three benchmark datasets, including NYUDv2, PASCAL-Context, and Cityscapes. Quantitative results demonstrate the superiority of MTMamba++ on multi-task dense prediction over previous methods while maintaining high computational efficiency. Qualitative evaluations show that MTMamba++ generates precise predictions.

A preliminary version of this work appeared in a conference paper [24]. Compared with the previous conference version, we propose a novel cross SSM (CSSM) mechanism that enables capturing the relationship between two sequences based on the SSM mechanism. By leveraging CSSM, we design a novel cross-task Mamba (CTM) block (i.e., S-CTM) to better achieve cross-task interaction. We also introduce a more effective and lightweight prediction head. Based on these innovations, MTMamba++ largely outperforms MTMamba [24]. Moreover, we extend our experiments to investigate the effectiveness of MTMamba++ on a new multi-task scene understanding benchmark dataset, i.e., Cityscapes [23]. We also provide more results and analysis to understand the proposed MTMamba++ model.

The rest of the paper is organized as follows. In Section II, we review some related works. In Section III, we present a detailed description of the various modules within our proposed MTMamba++ model. In Section IV, we quantitatively and qualitatively evaluate the proposed MTMamba++ model on three benchmark datasets (NYUDv2 [21], PASCAL-Context [22], and Cityscapes [23]). Finally, we make conclusions in Section V.

II. Related Works

A. Multi-Task Learning

Multi-task learning (MTL) is a learning paradigm that aims to jointly learn multiple related tasks using a single model [26], [27]. Current MTL research mainly focuses on multi-objective optimization [28], [29], [30], [31], [32], [33] and network architecture design [5], [6], [9], [10], [11], [12], [13], [14], [34], [35]. In multi-task visual scene understanding, most existing works focus on designing architecture [1], especially developing specific modules in the decoder to facilitate knowledge exchange among different tasks. For instance, based on CNN, Xu et al. [5] introduce PAD-Net, which integrates an effective multi-modal

Authorized licensed use limited to: Chinese University of Hong Kong. Downloaded on October 13,2025 at 02:58:43 UTC from IEEE Xplore. Restrictions apply.

distillation module aimed at enhancing information exchange among various tasks within the decoder. MTI-Net [6] is a complex multi-scale and multi-task CNN architecture that facilitates information distillation across various feature scales. As the convolution operation only captures local features [7], recent approaches [9], [10], [11], [12], [13], [14] develop Transformer-based decoders to grasp global context by attention mechanism [15]. For example, InvPT [9] is a Transformer-based multi-task architecture that employs an effective UP-Transformer block for multi-task feature interaction at different feature scales. MQTransformer [10] uses a cross-task query attention module in the decoder to enable effective task association and information communication.

These works demonstrate the significance of long-range dependency modeling and the enhancement of cross-task correlation for multi-task dense scene understanding. Different from existing methods, we propose a novel multi-task architecture derived from the SSM mechanism [36] to capture global information better and promote cross-task interaction.

B. State Space Models

State space models (SSMs) are a mathematical framework for characterizing dynamic systems, capturing the dynamics of input-output relationships via a hidden state. SSMs have found broad applications in various fields such as reinforcement learning [37], computational neuroscience [38], and linear dynamical systems [39]. Recently, SSMs have emerged as an alternative mechanism to model long-range dependencies in a manner that maintains linear complexity with respect to sequence length. Compared with the convolution operation, which excels at capturing local dependence, SSMs exhibit enhanced capabilities for modeling long sequences. Moreover, in contrast to attention mechanism [15], which incurs quadratic computational costs with respect to sequence length [16], [17], SSMs are more computation- and memory-efficient.

To improve the expressivity and efficiency of SSMs, many different structures have been proposed. Gu et al. [19] propose structured state space models (S4) to enhance computational efficiency by decomposing the state matrix into low-rank and normal matrices. Many follow-up works attempt to improve the effectiveness of S4. For instance, Fu et al. [40] propose a new SSM layer called H3 to reduce the performance gap between SSM-based networks and Transformers in language modeling. Mehta et al. [41] introduce a gated state space layer leveraging gated units to enhance the models' expressive capacity.

Recently, Gu and Dao [36] propose a new SSM-based architecture termed Mamba, which incorporates a new SSM called S6. This SSM is an input-dependent selection mechanism derived from S4. Mamba has demonstrated superior performance over Transformers on various benchmarks, such as language modeling [36], [42], [43], graph reasoning [44], [45], medical image analysis [46], [47], and image classification [20], [48]. Different from existing research efforts on Mamba, which mainly focus on single-task settings, in this paper, we consider a more challenging multi-task setting and propose a novel cross-task Mamba module to capture inter-task dependence.

III. Methodology

In this section, we begin with the foundational knowledge of state space models (Section III-A) and provide an overview of the proposed MTMamba++ in Section III-B. Next, we delve into a detailed exploration of each component in the decoder of MTMamba++, including the encoder in Section III-C, three types of block in the decoder (i.e., the ECR block in Section III-D, the STM block in Section III-E, and the CTM block in Section III-F), and the prediction head in Section III-G.

A. Preliminaries

SSMs [18], [19], [36], derived from the linear systems theory [39], map an input sequence $x(t) \in \mathbb{R}$ to an output sequence $y(t) \in \mathbb{R}$ though a hidden state $\mathbf{h} \in \mathbb{R}^N$ using a linear ordinary differential equation:

$$\mathbf{h}'(t) = \mathbf{A}\mathbf{h}(t) + \mathbf{B}x(t), \tag{1}$$

$$y(t) = \mathbf{C}^\top \mathbf{h}(t) + Dx(t), \tag{2}$$

where $\mathbf{A} \in \mathbb{R}^{N\times N}$ is the state transition matrix, $\mathbf{B} \in \mathbb{R}^N$ and $\mathbf{C} \in \mathbb{R}^N$ are projection matrices, and $D \in \mathbb{R}$ is the skip connection. Equation (1) defines the evolution of the hidden state $\mathbf{h}(t)$, while Equation (2) specifies that the output is derived from a linear transformation of the hidden state $\mathbf{h}(t)$ combined with a skip connection from the input $x(t)$.

Given that continuous-time systems are not compatible with digital computers and the discrete nature of real-world data, a discretization process is essential. This process approximates the continuous-time system with a discrete-time one. Let $\Delta \in \mathbb{R}$ be a discrete-time step size. Equations (1) and (2) are discretized as

$$\mathbf{h}_t = \bar{\mathbf{A}}\mathbf{h}_{t-1} + \bar{\mathbf{B}}x_t, \tag{3}$$

$$y_t = \bar{\mathbf{C}}^\top \mathbf{h}_t + Dx_t, \tag{4}$$

where $x_t = x(\Delta t)$, and

$$\begin{aligned} \bar{\mathbf{A}} &= \exp(\Delta\mathbf{A}), \\ \bar{\mathbf{B}} &= (\Delta\mathbf{A})^{-1}\left(\exp(\Delta\mathbf{A}) - \mathbf{I}\right)\cdot\Delta\mathbf{B} \approx \Delta\mathbf{B}, \\ \bar{\mathbf{C}} &= \mathbf{C}. \end{aligned} \tag{5}$$

S4 [19] treats $\mathbf{A}, \mathbf{B}, \mathbf{C}$, and Δ as trainable parameters and optimizes them by gradient descent. However, these parameters do not explicitly depend on the input sequence, which can lead to suboptimal extraction of contextual information. To address this limitation, Mamba [36] introduces a new SSM, namely S6. As illustrated in Fig. 4(a), it incorporates an input-dependent selection mechanism that enhances the system's ability to discern and select relevant information contingent upon the input sequence. Specifically, $\mathbf{B}, \mathbf{C}$, and Δ are defined as functions of the input $\mathbf{x} \in \mathbb{R}^{B\times L\times C}$. Following the computation of these parameters, $\bar{\mathbf{A}}, \bar{\mathbf{B}}$, and $\bar{\mathbf{C}}$ are calculated via Equation (5). Subsequently, the output sequence $\mathbf{y} \in \mathbb{R}^{B\times L\times C}$ is computed by Equations (3) and (4), thereby improving the contextual information extraction. Without specific instructions, in this paper, S6 [36] is used in the SSM mechanism.

Authorized licensed use limited to: Chinese University of Hong Kong. Downloaded on October 13,2025 at 02:58:43 UTC from IEEE Xplore. Restrictions apply.

Fig. 2. (a) Illustration of the ECR (expand, concatenate, and reduce) block. It is responsible for upsampling the task feature and fusing it with the multi-scale feature from the encoder. More details are provided in Section III-D. (b) Overview of the self-task Mamba (STM) block, which is responsible for learning discriminant features for each task. Its core module SS2D is derived from [20]. As shown in Fig. 4(b), SS2D extends 1D SSM operation (introduced in Section III-A) to process 2D images. More details about STM are put in Section III-E.

B. Overall Architecture

An overview of MTMamba++ is illustrated in Fig. 1. It contains three components: an off-the-shelf encoder, a Mamba-based decoder, and task-specific prediction heads. Specifically, the encoder is shared across all tasks and plays a pivotal role in extracting multi-scale generic visual representations from the input image. The decoder consists of three stages, each of which progressively expands the spatial dimensions of the feature maps. This expansion is crucial for dense prediction tasks, as the resolution of the feature maps directly impacts the accuracy of the pixel-level predictions [9]. Each decoder stage is equipped with the ECR block designed to upsample the feature and integrate it with high-level features derived from the encoder. Following this, the STM block is employed to capture the long-range spatial relationship for each task. Additionally, the CTM block facilitates feature enhancement for each task by promoting knowledge exchange across different tasks. We design two types of CTM block, namely F-CTM and S-CTM, as introduced in Section III-F. In the end, a prediction head is used to generate the final prediction for each task. We introduce two types of head, called DenseHead and LiteHead, as described in Section III-G.

MTMamba++ and our preliminary version MTMamba [24] have a similar architecture. The default configuration for MTMamba++ utilizes the S-CTM block and LiteHead, while the default configuration for MTMamba employs the F-CTM block and DenseHead.

C. Encoder

The encoder in MTMamba++ is shared across different tasks and is designed to learn generic multi-scale visual features from the input RGB image. As an example, we consider the Swin Transformer [25], which segments the input image into non-overlapping patches. Each patch is treated as a token, and its feature representation is a concatenation of the raw RGB pixel values. After patch segmentation, a linear layer is applied to project the raw token into a C-dimensional feature embedding. The projected tokens then sequentially pass through four stages of the encoder. Each stage comprises multiple Swin Transformer blocks and a patch merging layer. The patch merging layer is specifically utilized to downsample the spatial dimensions by a factor of $2\times$ and expand the channel numbers by a factor of $2\times$, while the Swin Transformer blocks are dedicated to learning and refining the feature representations. Finally, for an input image with dimensions $H \times W \times 3$, where H and W denote the height and width, the encoder generates hierarchical feature representations at four different scales, i.e., $\frac{H}{4} \times \frac{W}{4} \times C$, $\frac{H}{8} \times \frac{W}{8} \times 2C$, $\frac{H}{16} \times \frac{W}{16} \times 4C$, and $\frac{H}{32} \times \frac{W}{32} \times 8C$.

D. ECR Block

The ECR (expand, concatenate, and reduce) block is responsible for upsampling the feature and aggregating it with the encoder's feature. As illustrated in Fig. 2(a), it contains three steps. For an input feature, ECR block first $2\times$ upsamples the feature resolution and $2\times$ reduces the channel number by a linear layer and the rearrange operation. Then, the feature is fused with the high-level feature from the encoder through skip connections. Fusing these features is crucial for compensating the loss of spatial information that occurs due to downsampling in the encoder. Finally, a 1×1 convolutional layer is used to reduce the channel number. Consequently, the ECR block facilitates the efficient recovery of high-resolution details, which is essential for dense prediction tasks that require precise spatial information.

E. STM Block

The self-task Mamba (STM) block is responsible for learning task-specific features. As illustrated in Fig. 2(b), its core module is the 2D-selective-scan (SS2D) module, which is derived from [20]. The SS2D module is designed to address the limitations of applying 1D SSMs (as discussed in Section III-A) to process 2D image data. As depicted in Fig. 4(b), it unfolds

Authorized licensed use limited to: Chinese University of Hong Kong. Downloaded on October 13,2025 at 02:58:43 UTC from IEEE Xplore. Restrictions apply.

Fig. 3. Illustration of two types of cross-task Mamba (CTM) block. (a) F-CTM contains a task-shared fusion block for generating a global representation $\mathbf{z}^{\text{sh}}$ and T task-specific feature blocks (only one is illustrated) for obtaining each task's feature $\mathbf{z}^t$. Each task's output is the aggregation of task-specific feature $\mathbf{z}^t$ and global feature $\mathbf{z}^{\text{sh}}$ weighted by a task-specific gate $\mathbf{g}^t$. More details about F-CTM are provided in Section III-F1. (b) Similar to F-CTM, S-CTM generates a global feature by a fusion block and processes each task's feature with a task-specific block (only one is illustrated). Differently, S-CTM achieves semantic-aware cross-task interaction in the cross SS2D (CSS2D) module, which is shown in Fig. 4(d). More details about S-CTM and CSS2D are provided in Section III-F2.

the feature map along four distinct directions, creating four unique feature sequences, each of which is then processed by an SSM. The outputs from four SSMs are subsequently added and reshaped to form a comprehensive 2D feature map.

For an input feature, the STM block operates through several stages: it first employs a linear layer to expand the channel number by a controllable expansion factor α. A convolutional layer with a SiLU activation function is used to extract local features. The SS2D operation models the long-range dependencies within the feature map. An input-dependent gating mechanism is integrated to adaptively select the most salient representations derived from the SS2D process. Finally, another linear layer is applied to reduce the expanded channel number, yielding the output feature. Therefore, the STM block effectively captures both local and global spatial information, which is essential for the accurate learning of task-specific features in dense scene understanding tasks.

F. CTM Block

While the STM block excels at learning distinctive representations for individual tasks, it fails to establish inter-task connections, which are essential for enhancing the performance of MTL. To address this limitation, we propose the novel cross-task Mamba (CTM) block, depicted in Fig. 3, which facilitates information exchange across various tasks. We develop two types of CTM blocks, called F-CTM and S-CTM, from different perspectives to achieve cross-task interaction.

1) F-CTM: Feature-Level Interaction: As shown in Fig. 3(a), F-CTM comprises a task-shared fusion block and T task-specific feature blocks, where T is the number of tasks. It inputs T features and outputs T features. For each task, the input features have a channel dimension of C.

The task-shared fusion block first concatenates all task features, resulting in a concatenated feature with a channel dimension of TC. This concatenated feature is then fed into a linear layer to transform the channel dimension from TC to αC, aligning it with the dimensions of the task-specific features from the task-specific feature blocks, where α is the expansion factor introduced in Section III-E. The transformed feature is subsequently processed through a sequence of operations "Conv - SiLU - SS2D" to learn a global representation $\mathbf{z}^{\text{sh}}$, which contains information from all tasks.

In the task-specific feature block, each task independently processes its own feature representation $\mathbf{z}^t$ through its own sequence of operations "Linear - Conv - SiLU - SS2D". Then, we use a task-specific and input-dependent gate $\mathbf{g}^t$ to aggregate the task-specific representation $\mathbf{z}^t$ and the global representation $\mathbf{z}^{\text{sh}}$ as $\mathbf{g}^t \times \mathbf{z}^t + (1 - \mathbf{g}^t) \times \mathbf{z}^{\text{sh}}$.

Hence, F-CTM allows each task to adaptively integrate the cross-task representation with its own feature, promoting information sharing and interaction among tasks. The use of input-dependent gates ensures that each task can selectively emphasize either its own feature or the shared global representation based on the input data, thereby enhancing the model's ability to learn discriminative features in a multi-task learning context.

2) S-CTM: Semantic-Aware Interaction: While feature fusion in F-CTM is an effective way to interact with information, it may not be sufficient to capture all the complex relationships across different tasks, especially in multi-task scene understanding where the interactions between multiple pixel-level dense prediction tasks are highly dynamic and context-dependent. Thus, we propose S-CTM to achieve semantic-aware interaction.

As shown in Fig. 3(b), S-CTM contains a task-shared fusion block and T task-specific feature blocks. The fusion block first concatenates all task features and then passes the concatenated

Authorized licensed use limited to: Chinese University of Hong Kong. Downloaded on October 13,2025 at 02:58:43 UTC from IEEE Xplore. Restrictions apply.

Fig. 4. (a) Illustration of SSM. Given an input sequence, SSM first computes the input-dependent parameters (i.e., $\mathbf{B}$, $\mathbf{C}$, and Δ) and then calculates the output by querying the input through Equations (3) and (4). More details about SSM are provided in Section III-A. (b) Overview of SS2D from [20], which extends 1D SSMs to process 2D images. It unfolds the 2D feature map along four directions, generating four different feature sequences, each of which is then fed into an SSM. The four outputs are aggregated and folded to the 2D feature. (c) Illustration of the proposed cross SSM (CSSM), which enables modeling the relationships between two input sequences based on the SSM mechanism. In CSSM, one input sequence is used to compute (i.e., $\mathbf{B}$, $\mathbf{C}$, and Δ) and the other input is considered as the query. The output of CSSM is computed via Equations (3) and (4). More details about CSSM are provided in Section III-F. (d) Overview of the proposed cross SS2D (CSS2D). It inputs two 2D feature maps, scans them along four directions to generate four pairs of feature sequences, and then passes each pair through a CSSM. The outputs of CSSMs are subsequently added and reshaped to form a final 2D output feature. The details of CSS2D are put in Section III-F2.

feature through two convolution layers to generate the global representation, which contains knowledge across all tasks. The task-specific feature block in S-CTM is adapted from the STM block by replacing the SS2D with a novel cross SS2D (CSS2D). The additional input of CSS2D is from the task-shared fusion block.

As discussed in Section III-A, SSM only models the internal relationship within a single input sequence, but it does not capture the interactions between two different sequences. To address this limitation, we propose the cross SSM (CSSM) to model the relationship between the task-specific feature sequence (blue) and the task-shared feature sequence (red), as illustrated in Fig. 4(c). CSSM receives two sequences as input and outputs one sequence. The task-shared feature sequence is used to generate the SSMs parameters (i.e., $\mathbf{B}$, $\mathbf{C}$, and Δ), and the task-specific feature sequence is considered as the query input $\mathbf{x}$. The output is computed via Equations (3) and (4). Consequently, by leveraging the SSM mechanism, CSSM can capture the interactions between two input sequences at the semantic level. Furthermore, we extend SS2D as CSS2D, as shown in Fig. 4(d). This module takes two 2D input features, expands them along four directions to generate four pairs of feature sequences, and feeds each pair into a CSSM. The outputs from these sequences are subsequently aggregated and reshaped to form a 2D output feature.

Therefore, compared with F-CTM, S-CTM can better learn context-aware relationships because of the CSSM mechanism. CSSM can explicitly and effectively model long-range spatial relationships within two sequences, allowing S-CTM to understand the interactions between task-specific features and the global representation, which is critical for multi-task learning scenarios. In contrast, the feature fusion in F-CTM makes it difficult to capture the complex dependencies inherent across tasks.

G. Prediction Head

As shown in Fig. 1, after the decoder, the size of task-specific feature is $\frac{H}{4} \times \frac{W}{4} \times C$. Each task has its own prediction head to generate its final prediction. We introduce two types of prediction heads as follows.

1) Densehead: DenseHead is inspired by [49] and is used in our preliminary version MTMamba [24]. Specifically, each head contains a patch expand operation and a final linear layer. The patch expanding operation, similar to the one in the ECR block (as shown in Fig. 2(a)), performs $4\times$ upsampling to restore the resolution of the feature maps to the original input resolution $H \times W$. The final linear layer is used to project the feature channels to the task's output dimensions and output the final pixel-wise prediction.

2) Litehead: In DenseHead, upsampling is performed first, which can lead to a significant computational cost. Hence, we introduce a more simple, lightweight, and effective head architecture, called LiteHead. Specifically, it consists of a 3×3 convolutional layer, followed by a batch normalization layer, a ReLU activation function, and a final linear layer that projects the feature channels onto the task's output dimensions. Subsequently, the feature is simply interpolated to align with the input resolution and then used as the output. Thus, LiteHead

Authorized licensed use limited to: Chinese University of Hong Kong. Downloaded on October 13,2025 at 02:58:43 UTC from IEEE Xplore. Restrictions apply.

is much more computationally efficient than DenseHead. Note that since each task has its own head, the overall computational cost reduction is linearly related to the number of tasks.

IV. Experiments

In this section, we conduct extensive experiments to evaluate the proposed MTMamba++ in multi-task dense scene understanding.

A. Experimental Setups

1) Datasets: Following [9], [11], [12], we conduct experiments on three multi-task dense prediction benchmark datasets:

1) NYUDv2 [21] contains a number of indoor scenes, including 795 training images and 654 testing images. It consists of four tasks: 40-class semantic segmentation (Semseg), monocular depth estimation (Depth), surface normal estimation (Normal), and object boundary detection (Boundary).
2) PASCAL-Context [22], originated from the PASCAL dataset [50], includes both indoor and outdoor scenes and provides pixel-wise labels for tasks like semantic segmentation, human parsing (Parsing), and object boundary detection, with additional labels for surface normal estimation and saliency detection tasks generated by [51]. It contains 4,998 training images and 5,105 testing images.
3) Cityscapes [23] is an urban scene understanding dataset. It has two tasks (19-class semantic segmentation and disparity estimation) with 2,975 training and 500 testing images.

2) Implementation Details: We use the Swin-Large Transformer [25] pretrained on the ImageNet-22K dataset [52] as the encoder. The expansion factor α is set to 2 in both STM and CTM blocks. Following [9], [11], [12], we resize the input images of NYUDv2, PASCAL-Context, and Cityscapes datasets as 448×576, 512×512, and 512×1024, respectively, and use the same data augmentations including random color jittering, random cropping, random scaling, and random horizontal flipping. The ℓ_1 loss is used for depth estimation and surface normal estimation tasks, while the cross-entropy loss is for other tasks. The proposed model is trained with a batch size of 4 for 40,000 iterations. The AdamW optimizer [53] with a weight decay of 1×10^{-6} and the polynomial learning rate scheduler are used for all three datasets. The learning rate is set to 2×10^{-5}, 8×10^{-5}, and 1×10^{-4} for NYUDv2, PASCAL-Context, and Cityscapes datasets, respectively.

3) Evaluation Metrics: Following [9], [11], [12], we adopt mean intersection over union (mIoU) as the evaluation metric for semantic segmentation and human parsing tasks, root mean square error (RMSE) for monocular depth estimation and disparity estimation tasks, mean error (mErr) for surface normal estimation task, maximal F-measure (maxF) for saliency detection task, and optimal-dataset-scale F-measure (odsF) for object boundary detection task. Moreover, we report the average relative performance improvement of an MTL model $\mathcal{A}$ over single-task (STL) models as the overall metric, which is defined as follows,

$$\Delta_m(\mathcal{A}) = 100\% \times \frac{1}{T}\sum_{t=1}^{T}(-1)^{s_t}\frac{M_t^{\mathcal{A}} - M_t^{\text{STL}}}{M_t^{\text{STL}}}, \quad (6)$$

where T is the number of tasks, $M_t^{\mathcal{A}}$ is the metric value of method $\mathcal{A}$ on task t, and s_t is 0 if a larger value indicates better performance for task t, and 1 otherwise.

B. Comparison With State-of-The-Art Methods

We compare the proposed MTMamba++ method with three types of MTL methods:

1) CNN-based methods, including Cross-Stitch [54], PAP [55], PSD [56], PAD-Net [5], MTI-Net [6], ATRC [57], and ASTMT [51];
2) Transformer-based methods, including InvPT [9], TaskPrompter [12], InvPT++ [11], MQTransformer [10], TSP-Transformer [13], and MLoRE [14]; and
3) Diffusion-based method TaskDiffusion [34].

Table I provides the results on NYUDv2 and PASCAL-Context datasets. As can be seen, MTMamba++ largely outperforms CNN-based, Transformer-based, and Diffusion-based methods, especially achieving the best performance in all four tasks of NYUDv2. Notably, MTMamba++ shows significant improvements over MLoRE [14] by +1.05 (mIoU) and +0.97 (odsF) in semantic segmentation and object boundary detection tasks, which demonstrates the superiority of MTMamba++. Moreover, MTMamba++ performs better than MTMamba, showing the effectiveness of S-CTM and LiteHead.

On the PASCAL-Context dataset, MTMamba++ continues to demonstrate superior performance on all tasks except the normal prediction task, which is also comparable. Compared with MLoRE [14], MTMamba++ achieves notable improvements of +0.53 (mIoU), +2.35 (mIoU), +0.66 (maxF), and +3.18 (odsF) in semantic segmentation, human parsing, saliency detection, and object boundary detection tasks, respectively. When compared to the diffusion-based method TaskDiffusion [34], MTMamba++ shows advantages of +0.73 (mIoU), +3.25 (mIoU), +0.62 (maxF), and +3.71 (odsF) in four tasks. These results clearly demonstrate the effectiveness of MTMamba++ for multi-task dense prediction. Furthermore, MTMamba++ outperforms our preliminary work MTMamba on three of five tasks while maintaining comparable performance on the remaining two, further validating the effectiveness of our proposed components.

Table II shows the results on the Cityscapes dataset. We can see that Mamba-based methods perform largely better than the previous CNN-based and Transformer-based approaches on both two tasks. Moreover, MTMamba++ archives the best performance. Notably, MTMamba++ outperforms TaksPrompter [12] by +6.72 (mIoU) in the semantic segmentation task, demonstrating that MTMamba++ is more effective. Besides, MTMamba++ performs better than MTMamba, which shows the effectiveness of S-CTM and LiteHead.

The qualitative comparisons with baselines (i.e., InvPT [9], TaskPrompter [12], and MTMamba [24]) on NYUDv2, PASCAL-Context, and Cityscapes datasets are shown in

Authorized licensed use limited to: Chinese University of Hong Kong. Downloaded on October 13,2025 at 02:58:43 UTC from IEEE Xplore. Restrictions apply.

TABLE I
COMPARISON WITH STATE-OF-THE-ART METHODS ON NYUDV2 (LEFT) AND PASCAL-CONTEXT (RIGHT) DATASETS

Method	Semseg mIoU↑	Depth RMSE↓	Normal mErr↓	Boundary odsF↑
CNN-based decoder				
Cross-Stitch [54]	36.34	0.6290	20.88	76.38
PAP [55]	36.72	0.6178	20.82	76.42
PSD [56]	36.69	0.6246	20.87	76.42
PAD-Net [5]	36.61	0.6270	20.85	76.38
MTI-Net [6]	45.97	0.5365	20.27	77.86
ATRC [57]	46.33	0.5363	20.18	77.94
Transformer-based decoder				
InvPT [9]	53.56	0.5183	19.04	78.10
InvPT++ [11]	53.85	0.5096	18.67	78.10
TaskPrompter [12]	55.30	0.5152	18.47	78.20
MQTransformer [10]	54.84	0.5325	19.67	78.20
TSP-Transformer [13]	55.39	0.4961	18.44	77.50
MLoRE [14]	55.96	0.5076	18.33	78.43
Diffusion-based decoder				
TaskDiffusion [34]	55.65	0.5020	18.43	78.64
Mamba-based decoder				
MTMamba [24]	55.82	0.5066	18.63	78.70
MTMamba++	**57.01**	**0.4818**	**18.27**	**79.40**

Method	Semseg mIoU↑	Parsing mIoU↑	Saliency maxF↑	Normal mErr↓	Boundary odsF↑
CNN-based decoder					
ASTMT [51]	68.00	61.10	65.70	14.70	72.40
PAD-Net [5]	53.60	59.60	65.80	15.30	72.50
MTI-Net [6]	61.70	60.18	84.78	14.23	70.80
ATRC [57]	62.69	59.42	84.70	14.20	70.96
ATRC-ASPP [57]	63.60	60.23	83.91	14.30	70.86
ATRC-BMTAS [57]	67.67	62.93	82.29	14.24	72.42
Transformer-based decoder					
InvPT [9]	79.03	67.61	84.81	14.15	73.00
InvPT++ [11]	80.22	69.12	84.74	13.73	74.20
TaskPrompter [12]	80.89	68.89	84.83	13.72	73.50
MQTransformer [10]	78.93	67.41	83.58	14.21	73.90
TSP-Transformer [13]	81.48	70.64	84.86	13.69	74.80
MLoRE [14]	81.41	70.52	84.90	**13.51**	75.42
Diffusion-based decoder					
TaskDiffusion [34]	81.21	69.62	84.94	13.55	74.89
Mamba-based decoder					
MTMamba [24]	81.11	72.62	84.14	14.14	**78.80**
MTMamba++	**81.94**	**72.87**	**85.56**	14.29	78.60

↑ (↓) indicates that a higher (lower) result corresponds to better performance. The best and second best results are highlighted in **bold** and underline, respectively.

TABLE II
COMPARISON WITH STATE-OF-THE-ART METHODS ON THE CITYSCAPES DATASET

Method	Semseg mIoU↑	Disparity RMSE↓
CNN-based decoder		
PAD-Net [5]	53.19	5.05
MTI-Net [6]	59.85	5.06
Transformer-based decoder		
InvPT [9]	71.78	4.67
TaskPrompter [12]	72.41	5.49
Mamba-based decoder		
MTMamba [24]	78.00	4.66
MTMamba++	**79.13**	**4.63**

↑ (↓) indicates that a higher (lower) result corresponds to better performance. The best and second best results are highlighted in **bold** and underline, respectively.

Fig. 5. A qualitative comparison of each decoder stage in MTMamba++ on NYUDv2. Zoom in for more details.

Figs. 6, 7, and 8, demonstrating that MTMmaba++ provides more precise predictions and details.

C. Model Analysis

In this section, we provide a comprehensive analysis of the proposed MTMamba++. Without specific instructions, the encoder in this section is the Swin-Large Transformer.

1) Effectiveness of Each Component: The decoders of MTMamba++ contain two types of core blocks: STM and CTM blocks. Compared to the preliminary version MTMamba [24], MTMamba++ replaces the F-CTM block and DenseHead of MTMamba with the S-CTM block and LiteHead, respectively.

In this experiment, we study the effectiveness of each component on the NYUDv2 dataset. We first introduce two groups of baselines:

1) "Multi-task" represents an MTL model using only standard Swin Transformer blocks [25] after the ECR block in each decoder stage for each task; and
2) "Single-task" means that each task has a task-specific encoder-decoder.

The results are shown in Table III, where #9 and #11 are the default configurations of MTMamba and MTMamba++, respectively.

Firstly, the STM block outperforms the Swin Transformer block [25] in terms of efficiency and effectiveness for multi-task dense prediction, as indicated by the superior results in Table III (#3 vs. #7 and #5 vs. #8). Secondly, merely increasing the number of STM blocks from two to three does not enhance performance significantly. When the F-CTM block is incorporated, the performance largely improves in terms of Δ_m (#9 vs. #7/#8), demonstrating the effectiveness of F-CTM. Thirdly, comparisons between #3 and #4, #5 and #6, as well as #9 and #10 show that LiteHead is more effective and efficient than DenseHead. Fourthly, compared #10 with #11, we can find that replacing F-CTM with S-CTM leads to a significant performance improvement in all tasks with a tiny additional cost, demonstrating that the semantic-aware interaction in S-CTM is

Authorized licensed use limited to: Chinese University of Hong Kong. Downloaded on October 13,2025 at 02:58:43 UTC from IEEE Xplore. Restrictions apply.

TABLE III
EFFECTIVENESS OF EACH CORE COMPONENT ON NYUDV2

#	Method	Each Decoder Stage	Head	Semseg mIoU↑	Depth RMSE↓	Normal mErr↓	Boundary odsF↑	Δ_m[%] ↑	#Param MB↓	FLOPs GB↓
1	Single-task	2*Swin	DenseHead	54.32	0.5166	19.21	77.30	0.00	889	1075
2		2*STM	DenseHead	54.94	0.5100	18.85	78.00	+1.29	864	1040
3	Multi-task	2*Swin	DenseHead	53.72	0.5239	19.97	76.50	-1.87	303	466
4		2*Swin	LiteHead	53.37	0.5201	19.62	78.40	-0.78	302	436
5		3*Swin	DenseHead	54.22	0.5225	19.84	77.40	-1.11	341	563
6		3*Swin	LiteHead	54.44	0.5117	19.65	78.60	+0.14	339	533
7	MTMamba++	2*STM	DenseHead	54.66	0.4984	18.81	78.20	+1.84	276	435
8		3*STM	DenseHead	54.75	0.5054	18.81	78.20	+1.55	300	517
9		2*STM+1*F-CTM	DenseHead	55.82	0.5066	18.63	78.70	+2.38	308	541
10		2*STM+1*F-CTM	LiteHead	56.53	0.5054	18.71	79.20	+2.82	306	510
11		2*STM+1*S-CTM	LiteHead	**57.01**	**0.4818**	**18.27**	**79.40**	**+4.82**	315	524

"Multi-task" denotes an MTL model where each task uses standard Swin Transformer blocks [25] after the ECR block in each decoder stage. "Single-task" is the single-task counterpart of "Multi-task". #11 is the default configuration of MTMamba++.

TABLE IV
COMPARISON BETWEEN SSM AND ATTENTION ON NYUDV2

	Semseg mIoU↑	Depth RMSE↓	Normal mErr↓	Boundary odsF↑	Δ_m[%] ↑	#Param MB↓	FLOPs GB↓
attention-based	55.15	0.4945	18.72	79.00	+2.63	448	796
SSM-based	**57.01**	**0.4818**	**18.27**	**79.40**	**+4.82**	315	524

We replace the SSM-related modules in MTMamba++ (i.e., the SS2D and CSS2D modules) with attention-based mechanisms (i.e., self-attention and cross-attention mechanisms).

TABLE V
EFFECTIVENESS OF EACH DECODER STAGE IN MTMAMBA++ ON NYUDV2

Stage1	Stage2	Stage3	Semseg mIoU↑	Depth RMSE↓	Normal mErr↓	Boundary odsF↑	Δ_m[%] ↑	#Param MB↓	FLOPs GB↓
✓	✗	✗	55.50	0.4960	18.98	67.90	-1.20	287	291
✓	✓	✗	55.54	0.4872	18.46	77.70	+3.08	309	393
✓	✓	✓	**57.01**	**0.4818**	**18.27**	**79.40**	**+4.82**	315	524

TABLE VI
EFFECT OF EACH SCAN IN CSS2D MODULE ON NYUDV2

	Semseg mIoU↑	Depth RMSE↓	Normal mErr↓	Boundary odsF↑	Δ_m[%] ↑
MTMamba++	**57.01**	**0.4818**	18.27	**79.40**	**+4.82**
w/o scan1	56.02	0.4962	18.30	79.30	+3.60
w/o scan2	56.50	0.4967	**18.22**	79.30	+3.90
w/o scan3	56.09	0.4874	18.41	79.30	+3.91
w/o scan4	56.27	0.4942	18.36	79.30	+3.73

more effective than F-CTM. Finally, the default configuration of MTMamba++ significantly surpasses the "Single-task" baselines across all tasks (#11 vs. #1/#2), thereby demonstrating the effectiveness of MTMamba++.

2) Comparison Between SSM and Attention: To demonstrate the superiority of the SSM-based architecture in multi-task dense prediction, we conduct an experiment on NYUDv2 by replacing the SSM-related components in MTMamba++ with attention-based counterparts. Specifically, we substitute the SS2D module in the STM block with window-based multi-head self-attention [25] and replace the CSS2D module in the S-CTM block with window-based multi-head cross-attention. The comparative results in Table IV show that MTMamba++ significantly outperforms the attention-based variant across all tasks while requiring approximately 29.7% fewer parameters and 34.2% lower FLOPs. This efficiency advantage is primarily from SSM's linear computational complexity with respect to sequence length, in contrast to the quadratic complexity of attention mechanisms. These results demonstrate that SSM-based architectures are more effective and efficient for multi-task dense prediction tasks, where we need to process high-resolution feature maps in pixel-level prediction.

3) Effectiveness of Each Decoder Stage: As shown in Fig. 1, the decoder of MTMamba++ consists of three stages. In this experiment, we study the effectiveness of these three stages on the NYUDv2 dataset. Table V presents the ablation results, clearly demonstrating that each decoder stage contributes positively to the performance of MTMamba++. The progressive performance gains achieved by successively incorporating each stage validate the effectiveness of our multi-stage decoder design in capturing and integrating multi-scale contextual features. As visualized in Fig. 5, this hierarchical feature aggregation enables progressively refined predictions with sharper boundaries, particularly benefiting the boundary detection task.

4) Effect of Each Scan in CSS2D Module: As mentioned in Section III-F2, the CSS2D module scans the 2D feature map from four different directions. We conduct an experiment on NYUDv2 to study the effect of each scan. The results are presented in Table VI. As can be seen, dropping any direction leads to a performance drop compared with the default configuration that uses all directions, showing that all directions are beneficial to MTMamba++.

5) Analysis of α: As mentioned in Sections III-E and III-F2, in MTMamba++, both STM and S-CTM blocks expand the feature channel to improve the model capacity by a hyperparameter α. We conduct an experiment on NYUDv2 to explore the relationship between α and task conflicts. Increasing the expansion factor α enhances the model's representational capacity for capturing both task-specific features and cross-task interactions. However, excessively large values can lead to increased computational complexity and over-parameterization. The redundancy in the representation space dilutes effective information and makes model optimization more challenging, resulting in worse performance.

Authorized licensed use limited to: Chinese University of Hong Kong. Downloaded on October 13,2025 at 02:58:43 UTC from IEEE Xplore. Restrictions apply.

TABLE VII
EFFECT OF EXPAND FACTOR α IN MTMAMBA++ ON NYUDV2 WITH DIFFERENT NUMBERS OF TASKS

	α	Semseg mIoU↑	Depth RMSE↓	Normal mErr↓	Boundary odsF↑	Δ_m[%] ↑
S-D	1	58.10	**0.4768**	-	-	**+7.33**
	2	**58.25**	0.4808	-	-	+7.08
	3	58.23	0.4859	-	-	+6.57
S-D-N	1	54.85	0.4956	18.57	-	+2.79
	2	**55.78**	**0.4888**	18.43	-	**+4.04**
	3	55.30	0.4932	**18.39**	-	+3.53
S-D-N-B	1	56.74	0.4927	18.45	**79.40**	+3.93
	2	**57.01**	**0.4818**	**18.27**	**79.40**	**+4.82**
	3	55.85	0.4882	18.40	79.10	+3.71

"S", "D", "N", and "B" denote the semantic segmentation, depth estimation, surface normal estimation, and boundary detection tasks, respectively.

TABLE VIII
PERFORMANCE OF MTMAMBA++ WITH DIFFERENT SCALES OF SWIN TRANSFORMER ENCODER ON NYUDV2

Method	Encoder	Semseg mIoU↑	Depth RMSE↓	Normal mErr↓	Boundary odsF↑
MTMamba [24]	Swin-Small	51.93	0.5246	**19.45**	77.80
MTMamba++		**52.44**	**0.5210**	19.51	**78.10**
MTMamba [24]	Swin-Base	53.62	0.5126	19.28	77.70
MTMamba++		**55.08**	**0.5006**	**18.78**	**78.60**
MTMamba [24]	Swin-Large	55.82	0.5066	18.63	78.70
MTMamba++		**57.01**	**0.4818**	**18.27**	**79.40**

The results in Table VII demonstrate that the optimal α value is correlated with the severity of task conflicts. For the 2-task setting (S-D), $\alpha = 1$ achieves the best Δ_m, because the semantic segmentation and depth estimation tasks have relatively low conflict, requiring minimal additional capacity for cross-task interaction modeling. However, when the normal estimation task is added in the 3-task setting (S-D-N), task conflicts become more severe as evidenced by the significant performance drop of both semantic segmentation and depth estimation tasks. In this case, $\alpha = 2$ becomes optimal in terms of Δ_m, indicating that increased representational capacity is needed to handle the heightened task conflicts. In the 4-task setting (S-D-N-B), while the boundary detection task is added, the conflicts appear to be somewhat alleviated as the boundary detection task can provide complementary information to other tasks. Thus, $\alpha = 2$ continues to perform best in terms of Δ_m, maintaining the balance between adequate capacity for conflict resolution and avoiding over-parameterization. Notably, $\alpha = 3$ consistently underperforms across all configurations, demonstrating that excessively large expansion factors lead to over-parameterization.

These results demonstrate that smaller α suffices for low-conflict scenarios, while moderately larger α is beneficial when severe conflicts exist, but excessively large α always degrades performance. Thus, $\alpha = 2$ is adopted as the default configuration in MTMamba++ as it provides robust performance across various multi-task scenarios.

6) Performance on Different Encoders: We perform an experiment on NYUDv2 to investigate the performance of the proposed MTMamba++ with different scales of Swin Transformer encoder. The results are shown in Table VIII. As can be seen, as the model capacity increases, MTMamba++ performs better on all tasks accordingly. Moreover, MTMamba++ consistently outperforms MTMamba on different encoders, confirming the effectiveness of the proposed S-CTM and LiteHead.

TABLE IX
COMPARISON WITH STATE-OF-THE-ART METHODS IN MODEL SIZE AND COST ON THE PASCAL-CONTEXT DATASET

Method	#Param MB↓	FLOPs GB↓	Semseg mIoU↑	Parsing mIoU↑	Saliency maxF↑	Normal mErr↓	Boundary odsF↑
PAD-Net† [5]	330	773	78.01	67.12	79.21	14.37	72.60
MTI-Net† [6]	851	774	78.31	67.40	84.75	14.67	73.00
ATRC† [57]	340	871	77.11	66.84	81.20	14.23	72.10
InvPT† [9]	423	669	79.03	67.61	84.81	14.15	73.00
TaskPrompter† [12]	401	497	80.89	68.89	84.83	13.72	73.50
InvPT++ [11]	421	667	80.22	69.12	84.74	13.73	74.20
TSP-Transformer [13]	422	1991	81.48	70.64	84.86	13.69	74.80
MLoRE [14]	407	571	81.41	70.52	84.90	**13.51**	75.42
TaskDiffusion [34]	416	610	81.21	69.62	84.94	13.55	74.89
MTMamba [24]	336	632	81.11	72.62	84.14	14.14	**78.80**
MTMamba++	343	609	**81.94**	**72.87**	**85.56**	14.29	78.60

† denotes that the results are from [12].

7) Analysis of Model Size and Cost: Table IX compares model size and FLOPs between the proposed MTMamba++ and baselines on the PASCAL-Context dataset. We can see that MTMamba++ achieves state-of-the-art performance while maintaining high computational efficiency. Specifically, with only 343 MB parameters (14.3%, 18.7%, 15.7%, and 17.5% fewer than InvPT, TSP-Transformer, MLoRE, and TaskDiffusion, respectively), our MTMamba++ achieves superior performance across most tasks. In terms of computational cost, MTMamba++ requires only 609 GB FLOPs, which is merely 30.6% of the resources needed by TSP-Transformer (1991 GB) while still outperforming it. Compared to MLoRE and TaskDiffusion, MTMamba++ achieves better results with comparable computational demands. These results confirm that MTMamba++ offers not only performance advantages but also practical benefits for real-world applications through its efficient use of computational resources.

D. Visualization of Predictions

In this section, we compare the output predictions from MTMamba++ against baselines, including InvPT [9], TaskPrompter [12], and MTMamba [24]. Figs. 6, 7, and 8 show the qualitative results on NYUDv2, PASCAL-Context, and Cityscapes datasets, respectively. As can be seen, MTMamba++ has better visual results than baselines in all datasets. For example, as highlighted with yellow circles in Fig. 6, MTMamba++ demonstrates fewer misclassification errors in semantic segmentation and produces sharper predicted boundaries in the boundary detection task. Fig. 7 illustrates that MTMamba++ achieves more accurate detection of small objects in both semantic segmentation and human parsing tasks, particularly evident in the highlighted regions where our method can effectively detect distant pedestrians. MTMamba++ also generates sharper predicted boundaries for the object boundary detection task. Similarly, as highlighted in Fig. 8, MTMamba++ achieves higher precision in detecting small objects (e.g., street lamps and tree trunks) in semantic segmentation, which are missed by Transformer-based methods. Hence, both qualitative study (Figs. 6, 7, and 8) and quantitative study (Tables I and II) show the superior performance of MTMamba++.

Authorized licensed use limited to: Chinese University of Hong Kong. Downloaded on October 13,2025 at 02:58:43 UTC from IEEE Xplore. Restrictions apply.

Fig. 6. Qualitative comparison with baselines (i.e., InvPT [9], TaskPrompter [12], and MTMamba [24]) on the NYUDv2 dataset. As highlighted, MTMamba++ generates better predictions with more accurate details and sharper boundaries. In the semantic segmentation task, the black regions in GT denote the background and are excluded from the computation of loss and evaluation metric (i.e., mIoU). Zoom in for more details.

Fig. 7. Qualitative comparison with baselines (i.e., InvPT [9], TaskPrompter [12], and MTMamba [24]) on the PASCAL-Context dataset. As highlighted, MTMamba++ generates better predictions with sharper boundaries and greater precision in small objects. Zoom in for more details.

Fig. 8. Qualitative comparison with baselines (i.e., InvPT [9], TaskPrompter [12], and MTMamba [24]) on the Cityscapes dataset. As highlighted, MTMamba++ produces more precise predictions in small objects. Zoom in for more details.

Authorized licensed use limited to: Chinese University of Hong Kong. Downloaded on October 13,2025 at 02:58:43 UTC from IEEE Xplore. Restrictions apply.

V. Conclusion

In this paper, we propose MTMamba++, a novel multi-task architecture with a Mamba-based decoder for multi-task dense scene understanding. With two types of core blocks (i.e., STM and CTM blocks), MTMamba++ can effectively model long-range dependency and achieve cross-task interaction. We design two variants of the CTM block to promote knowledge exchange across tasks from the feature and semantic perspectives, respectively. Experiments on three benchmark datasets demonstrate that MTMamba++ achieves better performance than previous methods while maintaining high computational efficiency.

References

[1] S. Vandenhende, S. Georgoulis, W. Van Gansbeke, M. Proesmans, D. Dai, and L. V. Gool, "Multi-task learning for dense prediction tasks: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 7, pp. 3614–3633, Jul. 2022.

[2] X. Liang, X. Liang, and H. Xu, "Multi-task perception for autonomous driving," in *Autonomous Driving Perception: Fundamentals and Applications*. Berlin, Germany: Springer, 2023, pp. 281–321.

[3] K. Hur et al., "GenhHPF: General healthcare predictive framework for multi-task multi-source learning," *IEEE J. Biomed. Health Informat.*, vol. 28, no. 1, pp. 502–513, Jan. 2024.

[4] Y. Ze et al., "GFFactor: Multi-task real robot learning with generalizable neural feature fields," in *Proc. Conf. Robot Learn.*, 2023, pp. 284–301.

[5] D. Xu, W. Ouyang, X. Wang, and N. Sebe, "PAD-Net: Multi-tasks guided prediction-and-distillation network for simultaneous depth estimation and scene parsing," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 675–684.

[6] S. Vandenhende, S. Georgoulis, and L. Van Gool, "MTI-Net: Multi-scale task interaction networks for multi-task learning," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 527–543.

[7] I. Bello, B. Zoph, A. Vaswani, J. Shlens, and Q. V. Le, "Attention augmented convolutional networks," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 3285–3294.

[8] K. Li et al., "UniFormer: Unifying convolution and self-attention for visual recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 10, pp. 12581–12600, Oct. 2023.

[9] H. Ye and D. Xu, "Inverted pyramid multi-task transformer for dense scene understanding," in *Proc. Eur. Conf. Comput. Vis.*, 2022, pp. 514–530.

[10] Y. Xu, X. Li, H. Yuan, Y. Yang, and L. Zhang, "Multi-task learning with multi-query transformer for dense prediction," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 34, no. 2, pp. 1228–1240, Feb. 2024.

[11] H. Ye and D. Xu, "InvPT++: Inverted pyramid multi-task transformer for visual scene understanding," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 46, no. 12, pp. 7493–7508, Dec. 2024.

[12] H. Ye and D. Xu, "TaskPrompter: Spatial-channel multi-task prompting for dense scene understanding," in *Proc. Int. Conf. Learn. Representations*, 2023.

[13] S. Wang et al., "TSP-Transformer: Task-specific prompts boosted transformer for holistic scene understanding," in *Proc. IEEE/CVF Winter Conf. Appl. Comput. Vis.*, 2024, pp. 914–923.

[14] Y. Yang, P.-T. Jiang, Q. Hou, H. Zhang, J. Chen, and B. Li, "Multi-task dense prediction via mixture of low-rank experts," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2024, pp. 27927–27937.

[15] A. Vaswani et al., "Attention is all you need," in *Proc. Conf. Neural Inf. Process. Syst.*, 2017, pp. 6000–6010.

[16] Y.-H. Wu, Y. Liu, X. Zhan, and M.-M. Cheng, "P2T: Pyramid pooling transformer for scene understanding," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 11, pp. 12760–12771, Nov. 2023.

[17] W. Wang et al., "CrossFormer : A versatile vision transformer hinging on cross-scale attention," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 46, no. 5, pp. 3123–3136, May 2024.

[18] A. Gu et al., "Combining recurrent, convolutional, and continuous-time models with linear state space layers," in *Proc. Conf. Neural Inf. Process. Syst.*, 2021, pp. 572–585.

[19] A. Gu, K. Goel, and C. Re, "Efficiently modeling long sequences with structured state spaces," in *Proc. Int. Conf. Learn. Representations*, 2022.

[20] Y. Liu et al., "VMamba: Visual state space model," in *Proc. Conf. Neural Inf. Process. Syst.*, 2024, pp. 103031–103063.

[21] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, "Indoor segmentation and support inference from RGBD images," in *Proc. Eur. Conf. Comput. Vis.*, 2012, pp. 746–760.

[22] X. Chen, R. Mottaghi, X. Liu, S. Fidler, R. Urtasun, and A. Yuille, "Detect what you can: Detecting and representing objects using holistic models and body parts," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2014, pp. 1979–1986.

[23] M. Cordts et al., "The cityscapes dataset for semantic urban scene understanding," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 3213–3223.

[24] B. Lin, W. Jiang, P. Chen, Y. Zhang, S. Liu, and Y.-C. Chen, "MTMamba: Enhancing multi-task dense scene understanding by mamba-based decoders," in *Proc. Eur. Conf. Comput. Vis.*, 2024, pp. 314–330.

[25] Z. Liu et al., "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 9992–10002.

[26] Y. Zhang and Q. Yang, "A survey on multi-task learning," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 12, pp. 5586–5609, Dec. 2022.

[27] W. Chen et al., "Gradient-based multi-objective deep learning: Algorithms, theories, applications, and beyond," 2025, *arXiv:2501.10945*.

[28] B. Lin, F. Ye, Y. Zhang, and I. W. Tsang, "Reasonable effectiveness of random weighting: A litmus test for multi-task learning," *Trans. Mach. Learn. Res.*, 2022. [Online]. Available: https://openreview.net/forum?id=jjtFD8A1Wx

[29] B. Lin et al., "Dual-balancing for multi-task learning," 2023, *arXiv:2308.12029*.

[30] F. Ye, B. Lin, Z. Yue, P. Guo, Q. Xiao, and Y. Zhang, "Multi-objective meta learning," in *Proc. Conf. Neural Inf. Process. Syst.*, 2021, pp. 21338–21351.

[31] F. Ye, B. Lin, X. Cao, Y. Zhang, and I. Tsang, "A first-order multi-gradient algorithm for multi-objective bi-level optimization," in *Proc. Eur. Conf. Artif. Intell.*, 2024, pp. 2621–2628.

[32] B. Liu, X. Liu, X. Jin, P. Stone, and Q. Liu, "Conflict-averse gradient descent for multi-task learning," in *Proc. Conf. Neural Inf. Process. Syst.*, 2021, pp. 18878–18890.

[33] F. Ye, B. Lin, Z. Yue, Y. Zhang, and I. Tsang, "Multi-objective meta-learning," *Artif. Intell.*, vol. 335, 2024, Art. no. 104184.

[34] Y. Yang, P.-T. Jiang, Q. Hou, H. Zhang, J. Chen, and B. Li, "Multi-task dense predictions via unleashing the power of diffusion," in *Proc. Int. Conf. Learn. Representations*, 2025.

[35] H. Shi, S. Ren, T. Zhang, and S. J. Pan, "Deep multitask learning with progressive parameter sharing," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2023, pp. 19867–19878.

[36] A. Gu and T. Dao, "Mamba: Linear-time sequence modeling with selective state spaces," in *Proc. Conf. Lang. Model.*, 2024.

[37] D. Hafner, T. Lillicrap, J. Ba, and M. Norouzi, "Dream to control: Learning behaviors by latent imagination," in *Proc. Int. Conf. Learn. Representations*, 2020.

[38] K. J. Friston, L. Harrison, and W. Penny, "Dynamic causal modelling," *Neuroimage*, vol. 19, pp. 1273–1302, 2003.

[39] J. P. Hespanha, *Linear Systems Theory*. Princeton, NJ, USA: Princeton Univ. Press, 2018.

[40] D. Y. Fu, T. Dao, K. K. Saab, A. W. Thomas, A. Rudra, and C. Re, "Hungry hungry hippos: Towards language modeling with state space models," in *Proc. Int. Conf. Learn. Representations*, 2023.

[41] H. Mehta, A. Gupta, A. Cutkosky, and B. Neyshabur, "Long range language modeling via gated state spaces," in *Proc. Int. Conf. Learn. Representations*, 2023.

[42] R. Grazzi, J. Siems, S. Schrodi, T. Brox, and F. Hutter, "Is mamba capable of in-context learning?," in *Proc. Int. Conf. Automated Mach. Learn.*, 2024, pp. 1–26.

[43] J. Wang, T. Gangavarapu, J. N. Yan, and A. M. Rush, "MambaByte: Token-free selective state space model," in *Proc. Conf. Lang. Model.*, 2024.

[44] C. Wang, O. Tsepa, J. Ma, and B. Wang, "Graph-mamba: Towards long-range graph sequence modeling with selective state spaces," 2024, *arXiv:2402.00789*.

[45] A. Behrouz and F. Hashemi, "Graph mamba: Towards learning on graphs with state space models," in *Proc. ACM SIGKDD Conf. Knowl. Discov. Data Mining*, 2024, pp. 119–130.

[46] J. Ma, F. Li, and B. Wang, "U-Mamba: Enhancing long-range dependency for biomedical image segmentation," 2024, *arXiv:2401.04722*.

[47] Z. Xing, T. Ye, Y. Yang, G. Liu, and L. Zhu, "SegMamba: Long-range sequential modeling mamba for 3D medical image segmentation," in *Proc. Int. Conf. Med. Image Comput. Comput.- Assist. Intervention*, 2024, pp. 578–588.

Authorized licensed use limited to: Chinese University of Hong Kong. Downloaded on October 13,2025 at 02:58:43 UTC from IEEE Xplore. Restrictions apply.

[48] L. Zhu, B. Liao, Q. Zhang, X. Wang, W. Liu, and X. Wang, "Vision mamba: Efficient visual representation learning with bidirectional state space model," in *Proc. Int. Conf. Mach. Learn.*, 2024, pp. 62429–62442.
[49] H. Cao et al., "Swin-Unet: Unet-like pure transformer for medical image segmentation," in *Proc. Eur. Conf. Comput. Vis.*, 2022, pp. 205–218.
[50] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (VOC) challenge," *Int. J. Comput. Vis.*, vol. 88, pp. 303–338, 2010.
[51] K.-K. Maninis, I. Radosavovic, and I. Kokkinos, "Attentive single-tasking of multiple tasks," in *Proc. IEEE/CVF Comput. Vis. Pattern Recognit.*, 2019, pp. 1851–1860.
[52] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2009, pp. 248–255.
[53] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *Proc. Int. Conf. Learn. Representations*, 2019.
[54] I. Misra, A. Shrivastava, A. Gupta, and M. Hebert, "Cross-stitch networks for multi-task learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 3994–4003.
[55] Z. Zhang, Z. Cui, C. Xu, Y. Yan, N. Sebe, and J. Yang, "Pattern-affinitive propagation across depth, surface normal and semantic segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 4101–4110.
[56] L. Zhou et al., "Pattern-structure diffusion for multi-task learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 4513–4522.
[57] D. Brüggemann, M. Kanakis, A. Obukhov, S. Georgoulis, and L. V. Gool, "Exploring relational context for multi-task dense prediction," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 15849–15858.

Baijiong Lin received the bachelor's degree from South China Normal University, in 2020. He is currently working toward the PhD degree with the Hong Kong University of Science and Technology (Guangzhou). His research interests include multi-task learning and large language models. He has published several papers on top-tier conferences and journals such as NeurIPS, ICML, ECCV, and JMLR.

Weisen Jiang received the bachelor's degree from the South China University of Technology, in 2012, and the master's degree from the University of Chinese Academy of Sciences, in 2015. He is currently working toward the PhD degree with the Hong Kong University of Science and Technology. His research interests include meta-learning and large language models. He has published several papers on top-tier conferences such as ICML, NeurIPS, and ICLR.

Pengguang Chen received the BEng degree in computer science from Nanjing University, and the PhD degree from the Chinese University of Hong Kong (CUHK), under the supervision of Prof. Jiaya Jia. He is currently a researcher in SmartMore. He serves as a reviewer for CVPR, ICCV, ECCV, *IEEE Transactions on Pattern Analysis and Machine Intelligence*. His research interests include neural architecture search, self-supervised learning, knowledge distillation, and semantic segmentation.

Shu Liu (Member, IEEE) received the BS degree from the Huazhong University of Science and Technology, and the PhD degree from the Chinese University of Hong Kong. He is now serves as co-founder and technical head in SmartMore. He was the winner of 2017 COCO Instance Segmentation Competition and received the Outstanding Reviewer of ICCV in 2019. He continuously served as a reviewer for *IEEE Transactions on Pattern Analysis and Machine Intelligence*, CVPR, ICCV, NIPS, ICLR and etc. His research interests include deep learning and computer vision.

Ying-Cong Chen (Member, IEEE) is an assistant professor with AI Thrust of Hong Kong University of Science and Technology (Guangzhou), and also jointly appointed by the Department of Computer Science & Engineering. Prior to that, he was a postdoctoral associate with the Computer Science & Artificial Intelligence Lab, Massachusetts Institute of Technology. He has been dedicated to research in computer vision, particularly in deep generative models. His works are published in top conferences and journals like *IEEE Transactions on Pattern Analysis and Machine Intelligence*, CVPR, ICCV, ECCV, and he has been invited to give oral presentations at these conferences. His research achievements include being selected for ESI highly cited papers, ICCV best paper nomination, and winning the first prize in the Natural Science Award of the China Society for Image and Graphics.

Authorized licensed use limited to: Chinese University of Hong Kong. Downloaded on October 13,2025 at 02:58:43 UTC from IEEE Xplore. Restrictions apply.