Supplementary Material: VSCode: General Visual Salient and Camouflaged Object Detection with 2D Prompt Learning

Ziyang Luo[1] Nian Liu[2,*] Wangbo Zhao[3] Xuguang Yang[1] Dingwen Zhang[1]
Deng-Ping Fan[5] Fahad Khan[2,4] Junwei Han[1]
[1]Northwestern Polytechnical University [2]Mohamed bin Zayed University of Artificial Intelligence
[3]National University of Singapore [4] CVL, Linköping University
[5] Nankai International Advanced Research Institute (SHENZHEN FUTIAN) & CS, Nankai University

1. Additional Implementation Details

For VSOD and VCOD tasks, we follow the common practice of utilizing Flownet2.0 [19] as the optical flow extractor due to its consistently strong performance. It is worth noting that our results for the VSOD task may differ significantly from previous studies. This discrepancy is due to our adoption of a PyTorch-based toolbox for evaluating all tasks, whereas previous methods relied on a MATLAB-based toolbox which has different implementation details*.

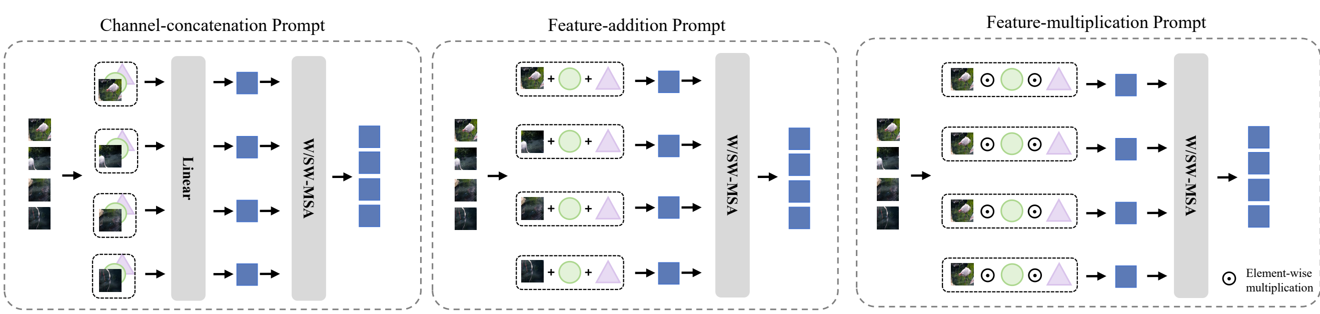

Figure 1. **Proposed channel-concatenation prompt, feature-addition prompt, and feature-multiplication prompt.**

2. Prompt Design Variants

We further investigate different types of prompts and conduct an analysis of their parameter counts. For computational efficiency, we limit our design to simple learnable prompts based on attention mechanism in the encoder, as shown in Figure 1.

2.1. Channel-concatenation Prompt

To maintain consistency in the fusion technique across RGB and other modalities, we suggest using learnable channel-concatenation prompt $\boldsymbol{p}_i^c \in \mathbb{R}^{1\times c_i}$. We concatenate the $\boldsymbol{p}_i^c$ with the image feature $\boldsymbol{f}_i^E$ along the channel dimension and utilize a linear projection to project them back to the original channel number. The entire process is expressed as follows:

$$\boldsymbol{f}_i^E = \text{Linear}([\boldsymbol{f}_i^E; \boldsymbol{p}_i^c]), \tag{1}$$

*Corresponding author: Nian Liu (liunian228@gmail.com)

*The reference link of the PyTorch-based toolbox is https://github.com/zzhanghub/eval-co-sod, and the link of MATLAB-based toolbox is https://github.com/DengPingFan/DAVSOD.

Settings	Params	RGB SOD DUTS[65]			RGBD SOD NJUD[25]			RGBT SOD VT5000[61]			VSOD SegV2[29]		
	(M)	$S_m \uparrow$	$F_m \uparrow$	$E_m \uparrow$	$S_m \uparrow$	$F_m \uparrow$	$E_m \uparrow$	$S_m \uparrow$	$F_m \uparrow$	$E_m \uparrow$	$S_m \uparrow$	$F_m \uparrow$	$E_m \uparrow$
channel-concatenation prompt	57.22	.798	.744	.852	.881	.870	.917	.846	.799	.898	.822	.744	.911
feature-addition prompt	54.09	.892	.875	.935	.924	.922	.956	.890	.854	.930	.899	.860	.945
feature-multiplication prompt	54.09	.732	.657	.793	.870	.858	.910	.792	.720	.847	.811	.706	.915
token-concatenation prompt	54.09	**.902**	**.890**	**.945**	**.931**	**.932**	**.962**	**.909**	**.877**	**.947**	**.931**	**.917**	**.975**

Table 1. **Ablation studies of different prompt design variants on the Swin-T backbone [42] with** 224×224 **image size.** We conduct evaluations on one representative dataset for each task.

Settings	Params	RGB SOD DUTS[65]			RGBD SOD NJUD[25]			RGBT SOD VT5000[61]			VSOD SegV2[29]			RGB COD CAMO[26]			VCOD CAD[1]		
	(M)	$S_m \uparrow$	$F_m \uparrow$	$E_m \uparrow$	$S_m \uparrow$	$F_m \uparrow$	$E_m \uparrow$	$S_m \uparrow$	$F_m \uparrow$	$E_m \uparrow$	$S_m \uparrow$	$F_m \uparrow$	$E_m \uparrow$	$S_m \uparrow$	$F_m \uparrow$	$E_m \uparrow$	$S_m \uparrow$	$F_m \uparrow$	$E_m \uparrow$
margin = 0.2	54.09	.908	.897	**.948**	.933	.934	.961	.911	.880	.949	.937	.927	.979	**.811**	**.786**	**.889**	.734	**.616**	**.817**
margin = 0.1	54.09	.907	.897	**.948**	.931	.933	.960	.911	.881	.949	.935	.920	.976	.810	.781	.884	.725	.588	.800
margin = 0	54.09	**.909**	**.899**	**.948**	**.935**	**.938**	**.965**	**.912**	**.882**	**.950**	**.943**	**.930**	**.984**	**.811**	.782	.884	**.736**	.614	.797

Table 2. **Ablation studies of the prompt discrimination loss settings with different margins using** 224×224 **image size.**

where $[;]$ indicates the concatenation operation and Linear means linear projection. Since the parameters of the linear operation can be expressed as $2c_i * c_i + c_i = 2c_i^2 + c_i$, and the parameters for the channel-concatenation prompt are $1 * c_i$. Therefore, the total number of parameters for the channel-concatenation prompt becomes $\sum_i 2c_i^2 + 2c_i$.

2.2. Feature-addition Prompt

In ViPT [40], prompts are introduced by incorporating carefully designed layers and added to the inputs. To emphasize simple and learnable prompt, we refer to these addition computational forms and introduce feature-addition prompt $\boldsymbol{p}_i^a \in \mathbb{R}^{1\times c_i}$ for the image features $\boldsymbol{f}_i^E$:

$$\boldsymbol{f}_i^E = \boldsymbol{f}_i^E + \boldsymbol{p}_i^a. \tag{2}$$

The feature-addition prompt can capture domain- or task-specific details for pixels. The total parameters for the feature-addition prompt amount to c_i.

2.3. Feature-multiplication Prompt

In segmentation tasks, masks are commonly employed to consolidate object information and extract distinct features [3]. Expanding on this concept, we utilize feature-multiplication prompt $\boldsymbol{p}_i^m \in \mathbb{R}^{1\times c_i}$ as the mask query and apply them to the image features. This operation is computed as follows:

$$\boldsymbol{f}_i^E = \boldsymbol{f}_i^E \odot \boldsymbol{p}_i^m. \tag{3}$$

The feature-multiplication prompt selectively extracts domain-specific or task-specific information from image features. In the case of the feature-multiplication prompt, the total number of parameters is also c_i.

We conduct experiments on domain-specific prompts using the aforementioned prompt forms, as shown in Table 1. Our token-concatenation prompt uses fewer parameters while delivering superior results, demonstrating the efficiency of our design. This also underscores that the concatenation design maximizes feature variations for different tasks and domains compared to addition and multiplication.

3. Further Ablation Study

3.1. Effectiveness of the Prompt Discrimination Loss

In fact, our prompt discrimination loss can be viewed as a specific instance of the hinge loss with a margin set to 0. To further evaluate its effectiveness, we present a more general expression

$$\mathcal{L}_{dis} = \sum_m \ln(1 + \max\{|\mathcal{CS}_m|, margin\}). \tag{4}$$

When the margin is greater than 0, it indicates that the loss doesn't impose any constraints on prompts when the correlation is below the margin. In other words, it signifies a higher degree of shared knowledge among different domains and tasks.

Summary	Task	NJUD [25]			NLPR[52]			DUTLF-Depth[54]			ReDWeb-S[38]			STERE[46]			SIP[7]		
		S_m	F_m	E_m	S_m	F_m	E_m	S_m	F_m	E_m	S_m	F_m	E_m	S_m	F_m	E_m	S_m	F_m	E_m
CMINet[75]	RGB-D	.929	.934	.957	.932	.922	.963	.912	.913	.938	.725	.726	.800	.918	.916	.951	.899	.910	.939
VST[39]	RGB-D	.922	.920	.951	.932	.920	.962	.943	.948	.969	.759	.763	.826	.913	.907	.951	.904	.915	.944
VST-T++ [36]	RGB-D	.928	.929	.958	.933	.921	.964	.944	.948	.969	.756	.757	.819	.916	.911	.950	.903	.914	.944
SPSN[27]	RGB-D	-	-	-	.923	.912	.960	-	-	-	-	-	-	.907	.902	.945	.892	.900	.936
CAVER[51]	RGB-D	.920	.924	.953	.929	.921	.964	.931	.939	.962	.730	.724	.802	.914	.911	.951	.893	.906	.934
VSCode-T	ZS RGB-D	.910	.912	.941	.912	.887	.940	.931	.936	.956	.746	.733	.809	.908	.904	.937	.925	.939	.963
VSCode-T	RGB-D	**.941**	**.945**	**.967**	**.938**	**.930**	**.966**	**.952**	**.959**	**.974**	**.766**	**.771**	**.831**	**.928**	**.926**	**.957**	**.917**	**.936**	**.955**

Table 3. **Quantitative comparison of our proposed VSCode with other 5 out-performing RGB-D SOD methods on six benchmark datasets.**"ZS" indicates zero-shot.

Summary	Task	COD10K[6]			NC4K[44]			CAMO[26]		
		S_m	F_m	E_m	S_m	F_m	E_m	$S_m \uparrow$	F_m	E_m
UJSC[28]	RGB	.817	.750	.902	.856	.835	.920	.803	.775	.867
SegMar[24]	RGB	.833	.755	.907	.841	.827	.907	.816	.803	.884
FEDER[15]	RGB	.822	.768	.905	.847	.833	.915	.802	.789	.873
VSCode-T	ZS RGB	**.836**	**.778**	**.916**	**.870**	**.850**	**.926**	**.830**	**.805**	**.904**
VSCode-T	RGB	**.847**	**.795**	**.925**	**.874**	**.853**	**.930**	**.838**	**.821**	**.909**

Table 4. **Quantitative comparison of our proposed VSCode with other SOTA RGB COD methods on three benchmark datasets.**"ZS" indicates zero-shot.

Since the correlation between different domains and tasks ranges from 0 to 0.26 when prompt discrimination loss is not applied, as shown in Figure 5 in the main text, we set margins of 0.1 and 0.2 to illustrate the effectiveness of our design. The results can be found in Table 2. Overall, the results obtained with a margin of 0 outperform other designs, providing evidence of the effectiveness of fully disentangling different domain and task knowledge in our joint learning approach.

However, setting the margin as 0 shows a decrease in COD tasks. This can be attributed to the significantly fewer training data available for COD tasks (7915 training images) compared to SOD tasks (30450 training images), which means COD tasks need more shared knowledge learned from the SOD data. Hence, attempting to completely separate SOD and COD knowledge might further compromise performance in COD tasks. In our future work, we will explore methods to balance the relationship between tasks and ensure comprehensive training for all tasks.

4. Deeper Analysis of Generalization Capacity

To further investigate our model's zero-shot generalization, we reserved separate tasks for zero-shot evaluation and retrained our model on other tasks. Recognizing the risk of overfitting when training with limited data, we evaluated the zero-shot capability using the RGB COD task and the RGB-D SOD task for single-modality task and SOD task, respectively (10553 training images for RGB SOD *v.s* 4040 for RGB COD, 4040 training images for RGB-D COD *v.s* 2985 for RGB-D SOD).

As depicted in Table 4, even without training with the RGB COD task, our VSCode model still outperforms state-of-the-art task-specific RGB COD models, although it works in a zero-shot way. This indicates that our model relies not only on the effectiveness of domain-specific prompts in segregating domain knowledge, but also on the accuracy of our task-specific prompts in integrating task-related knowledge. However, for the RGB-D SOD task, the performance of our zero-shot VSCode lags behind that of state-of-the-art task-specific training methods, as shown in Table 3. We hypothesize that this is because the depth maps of RGB-D COD datasets are generated using an off-the-shelf depth estimation model [55], which is different from most RGB-D SOD datasets that use real depth maps captured by Microsoft Kinect (*e.g.* NLPR [52]), light field cameras (*e.g.* DUTLF-Depth [54]), and smartphones (*e.g.* SIP [7]). The estimated depth maps of RGB-D COD might lack certain high-quality geometric cues for real scenarios, leading to incomplete depth knowledge for depth prompts. Despite our zero-shot performance in the RGB-D SOD task not outperforming state-of-the-art methods, the comparable performance still highlights the generalization ability of our VSCode when encountering unseen tasks.

5. More Comparison Results

To conserve space, we focus on presenting state-of-the-art methods from 2021 in the main text. In this section, we provide a more comprehensive comparison of state-of-the-art methods dating back to 2018, as shown in Table 5, Table 6, Table 7, Table 8, Table 9, and Table 10. We also introduce an additional evaluation metric, the mean absolute error (M), to assess model performance. Furthermore, we include various versions of our VSCode with different backbones for comparison with other models. For instance, VSCode-B utilizes the Swin-B backbone [42] and demonstrates exceptional performance compared to

Summary	Backbone	Params (M)	DUTS[65]				ECSSD[71]				HKU-IS[32]				PASCAL-S[34]				DUT-O[73]				SOD[45]			
			S_m	F_m	E_m	M	S_m	F_m	E_m	M	S_m	F_m	E_m	M	S_m	F_m	E_m	M	S_m	F_m	E_m	M	S_m	F_m	E_m	M
PiCANet[35]	ResNet50	47.22	.863	.840	.915	.040	.916	.929	.953	.035	.905	.913	.951	.031	.846	.824	.882	.071	.826	.767	.865	.054	.813	.824	.871	.073
AFNet[10]	VGG16	35.95	.867	.838	.910	.045	.914	.924	.947	.042	.905	.910	.949	.036	.849	.824	.877	.076	.826	.759	.861	.057	.811	.819	.867	.085
TSPOANet[41]	VGG16	-	.860	.828	.907	.049	.907	.919	.942	.047	.902	.909	.950	.039	.841	.817	.871	.082	.818	.750	.858	.062	.802	.809	.852	.094
EGNet-R[79]	ResNet50	111.64	.887	.866	.926	.039	.925	.936	.955	.037	.918	.923	.956	.031	.852	.825	.874	.080	.841	.778	.878	.053	.824	.831	.875	.080
ITSD-R[83]	ResNet50	26.47	.885	.867	.929	.041	.925	.939	.959	.035	.917	.926	.960	.031	.861	.839	.889	.771	.840	.792	.880	.061	.835	.849	.889	.075
MINet-R[50]	ResNet50	162.38	.884	.864	.926	.037	.925	.938	.957	.034	.919	.926	.960	.029	.856	.831	.883	.071	.883	.769	.869	.056	.830	.835	.878	.074
LDF-R[68]	ResNet50	25.15	.892	.877	.930	.034	.925	.938	.954	.034	.920	.929	.958	.028	.861	.839	.888	.067	.839	.782	.870	.052	.831	.841	.878	.071
CSF-R2[13]	Res2Net50	36.53	.890	.869	.929	.037	.931	.942	.960	.033	-	-	-	-	.863	.839	.885	.073	.838	.775	.869	.055	.826	.832	.883	.079
GateNet-R[80]	ResNet50	128.63	.891	.874	.932	.038	.924	.935	.955	.038	.921	.926	.959	.031	.863	.836	.886	.071	.840	.782	.878	.055	.827	.835	.877	.079
VST[39]	T2T-ViT$_t$-14	44.48	.896	.877	.939	.037	.932	.944	.964	.034	.928	.937	.968	.030	.873	.850	.900	.067	.850	.800	.888	.058	.854	.866	.902	.065
ICON-R[86]	ResNet50	33.09	.890	.876	.931	.037	.928	.943	.960	.032	.920	.931	.960	.029	.862	.844	.888	.071	.845	.799	.884	.057	.848	.861	.899	.067
VST-T++ [36]	Swin-T	53.60	.901	.887	.943	.033	.937	.949	.968	.029	.930	.939	.968	.026	**.878**	**.855**	**.901**	.063	.853	.804	.892	.053	.853	.866	.899	.065
MENet[67]	ResNet50	27.83	.905	.895	.943	.028	.927	.938	.956	.031	.927	.939	.965	**.023**	.871	.848	.892	**.062**	.850	.792	.879	.045	.841	.847	.884	.065
VSCode-T	Swin-T	54.09	**.917**	**.910**	**.954**	**.027**	**.945**	**.957**	**.971**	**.024**	**.935**	**.946**	**.970**	.024	**.878**	.852	.900	**.062**	**.869**	**.830**	**.910**	**.045**	**.863**	**.879**	**.908**	**.056**
EVP[40]	SegFormer-B4	64.52	.917	.910	.956	.027	.936	.949	.965	.029	.935	.945	.971	.024	.880	.859	.902	.061	.864	.822	.902	.047	.854	.873	.901	.065
VSCode-S	Swin-S	74.72	**.926**	**.922**	**.960**	**.024**	**.949**	**.959**	**.974**	**.022**	**.940**	**.951**	**.974**	**.021**	**.887**	**.864**	**.904**	**.058**	**.877**	**.840**	**.912**	**.043**	**.870**	**.882**	**.910**	**.054**
VSCode-B	Swin-B	117.41	**.932**	**.930**	**.965**	**.022**	**.949**	**.961**	**.974**	**.022**	**.941**	**.951**	**.974**	**.021**	**.890**	**.866**	**.906**	**.056**	**.880**	**.846**	**.913**	**.043**	**.871**	**.882**	**.910**	.056

Table 5. **Quantitative comparison of our proposed VSCode with other 14 SOTA RGB SOD methods on six benchmark datasets.** "-R","-R2", "-T", "-S", and "-B" mean the ResNet50 [16], Res2Net [12], SwinT-1k, SwinS-22k, and SwinB-22k[42] backbones, respectively. '-' indicates the code is not available. The best performance under all settings is **bolded**, and the best results under each setting are labeled in **bold**.

Summary	Backbone	Params (M)	NJUD [25]				NLPR[52]				DUTLF-Depth[54]				ReDWeb-S[38]				STERE[46]				SIP[7]			
			S_m	F_m	E_m	M	S_m	F_m	E_m	M	S_m	F_m	E_m	M	S_m	F_m	E_m	M	S_m	F_m	E_m	M	S_m	F_m	E_m	M
ATST[76]	VGG19	32.17	.885	.893	.930	.047	.909	.898	.951	.027	.916	.928	.953	.033	.679	.673	.758	.155	.896	.901	.942	.038	.849	.861	.901	.063
CMW[30]	VGG16	85.56	.870	.871	.927	.061	.917	.903	.951	.029	.797	.779	.864	.098	.634	.607	.714	.195	.852	.837	.907	.067	.705	.677	.804	.141
Cas-Gnn[43]	VGG16	-	.911	.916	.948	.036	.919	.906	.955	.025	.920	.926	.953	.030	-	-	-	-	.899	.901	.944	.039	-	-	-	-
HDFNet[48]	ResNet50	44.15	.908	.911	.944	.039	.923	.917	.963	.023	.908	.915	.945	.041	.728	.717	.804	.129	.900	.900	.943	.042	.886	.894	.930	.048
CoNet[23]	ResNet50	43.66	.896	.893	.937	.046	.912	.893	.948	.027	.923	.932	.959	.029	.696	.693	.782	.147	.905	.901	.947	.037	.860	.873	.917	.058
BBS-Net[9]	ResNet50	49.77	.921	.919	.949	.035	.931	.918	.961	.023	.882	.870	.912	.058	.693	.680	.763	.150	.908	.903	.942	.041	.879	.884	.922	.055
JL-DCF[11]	VGG16	143.52	.877	.892	.941	.066	.931	.918	.965	.022	.894	.891	.927	.048	.581	.546	.708	.213	.900	.895	.942	.044	0.885	.894	.931	.049
SPNet[84]	Res2Net50	67.88	.925	.928	.957	.029	.927	.919	.962	.021	.895	.899	.933	.045	.710	.715	.798	.129	.907	.906	.949	.037	.894	.904	.933	.043
CMINet[75]	ResNet50	188.12	.929	.934	.957	.029	.932	.922	.963	.021	.912	.913	0938	.038	.725	.726	.800	.121	.918	.916	.951	.032	.899	.910	.939	.040
DCF[22]	ResNet50	53.92	.904	.905	.943	.039	.922	.910	.957	.024	.925	.930	.956	.030	.709	.715	.790	.135	.906	.904	.948	.037	.874	.886	.922	.052
VST[39]	T2T-ViT$_t$-14	53.83	.922	.920	.951	.035	.932	.920	.962	.024	.943	.948	.969	.024	.759	.763	.826	.113	.913	.907	.951	.038	.904	.915	.944	.040
VST-T++ [36]	Swin-T	100.51	.928	.929	.958	.031	.933	.921	.964	.022	.944	.948	.969	.024	.756	.757	.819	.114	.916	.911	.950	.037	.903	.914	.944	.039
SPSN[27]	VGG16	-	-	-	-	-	.923	.912	.960	.023	-	-	-	-	-	-	-	-	.907	.902	.945	.036	.892	.900	.936	.043
CAVER[51]	ResNet50	55.79	.920	.924	.953	.032	.929	.921	.964	.022	.931	.939	.962	.028	.730	.724	.802	.121	.914	.911	.951	.034	.893	.906	.934	.043
VSCode-T	Swin-T	54.09	**.941**	**.945**	**.967**	**.025**	**.938**	**.930**	**.966**	**.020**	**.952**	**.959**	**.974**	**.019**	**.766**	**.771**	**.831**	**.105**	**.928**	**.926**	**.957**	**.030**	**.917**	**.936**	**.955**	**.032**
VSCode-S	Swin-S	74.72	**.944**	.949	**.970**	**.022**	.941	.932	.968	.018	**.960**	**.967**	**.980**	**.015**	**.777**	**.776**	**.829**	**.100**	.931	.928	.958	.028	**.924**	**.942**	**.958**	**.029**
VSCode-B	Swin-B	117.41	**.944**	**.950**	.969	.023	**.944**	**.939**	**.971**	**.017**	.959	**.967**	.978	.017	.772	.771	.828	.101	**.933**	**.931**	**.960**	**.028**	.913	.936	.950	.034

Table 6. **Quantitative comparison of our proposed VSCode with other 14 SOTA RGB-D SOD methods on six benchmark datasets.**

the VSCode-T and VSCode-S versions. Here, we compare our VSCode-B with FSPNet [17] in the RGB COD task, which employs a backbone with similar parameters to Swin-B [42], i.e. DeiT-B [59].

It's worth noting that we omit comparisons with some state-of-the-art methods for RGB COD. For example, DCOFD [82] employs a significantly larger image size of 416, which exceeds our approach's specifications. ZoomNet [49] and MFFN [81] use multi-scale input images. All these settings lead to an unfair comparison with our method.

6. Visual Comparison with State-of-the-art Methods

In this section, we provide visual comparison results alongside state-of-the-art methods for four SOD tasks (RGB SOD, RGB-D SOD, RGB-T SOD, and VSOD) and three COD tasks (RGB COD, VCOD, and RGB-D COD). The results, as depicted in Figure 2, Figure 3, Figure 4, Figure 5, Figure 6, Figure 7, and Figure 8, showcase the exceptional capabilities of our VSCode model across a variety of challenging scenarios. These scenarios include handling significantly small and large objects, multiple objects, occluded objects, and situations with uncertain boundaries, where existing methods often encounter difficulties.

Summary	Backbone	Params (M)	VT821[63]				VT1000[62]				VT5000[61]			
			S_m	F_m	E_m	M	S_m	F_m	E_m	M	S_m	F_m	E_m	M
FCMF[78]	VGG16	-	.760	.667	.810	.081	.873	.851	.921	.037	0.814	.758	.866	.055
ADF[61]	VGG16	-	.808	.749	.841	.077	.909	.908	.950	.034	.863	.837	.911	.048
ECFFNet[85]	ResNet34	-	.877	.835	.911	.034	.924	.919	.959	0.021	.876	.850	.922	.037
CGFNet[64]	VGG16	69.92	.881	.866	.920	.038	.923	.923	.959	.023	.883	.852	.926	.039
CSRNet[18]	ESPNetv2	1.01	.885	.855	.920	.037	.919	.901	.952	.027	.868	.821	.912	.045
MGAI[57]	Res2Net50	87.09	.891	.870	.933	.030	.929	.921	.965	.024	.884	.846	.930	.037
MIDD[60]	VGG16	52.43	.871	.847	.916	.044	.916	.904	.956	.030	.868	.834	.919	.045
TNet[5]	ResNet50	87.41	.899	.885	.936	.030	.929	.921	.965	.024	.895	.864	.936	.036
CGMDRNet[2]	Res2Net50	-	.894	.872	.932	.035	.931	.927	.966	.020	.896	.877	.939	.032
VST-T++ [36]	Swin-T	100.51	.894	.861	.923	.034	.941	.931	.972	.020	.895	.854	.933	.037
CAVER[51]	ResNet50	55.79	.891	.874	.933	.033	.936	.927	.970	.021	.892	.857	.935	.035
VSCode-T	Swin-T	54.09	**.921**	**.906**	**.951**	**.021**	**.949**	**.944**	**.981**	**.017**	**.918**	**.892**	**.954**	**.028**
VSCode-S	Swin-S	74.72	.926	.910	.954	.021	.952	.947	.981	.016	.925	.900	.959	.026
VSCode-B	Swin-B	117.41	**.928**	**.915**	**.956**	**.021**	**.953**	**.949**	**.984**	**.016**	**.930**	**.907**	**.962**	**.025**

Table 7. **Quantitative comparison of our proposed VSCode with other 11 SOTA RGB-T SOD methods on three benchmark datasets.**

Summary	Backbone	Params (M)	DAVIS [53]				FBMS[47]				ViSal[66]				SegV2[29]				DAVSOD-Easy[8]				DAVSOD-Normal[8]			
			S_m	F_m	E_m	M	S_m	F_m	E_m	M	S_m	F_m	E_m	M	S_m	F_m	E_m	M	S_m	F_m	E_m	M	S_m	F_m	E_m	M
PDB[56]	ResNet50	-	.880	.851	.949	.030	.850	.821	.882	.072	.926	.922	.970	.024	.871	.820	.867	.024	-	-	-	-	-	-	-	-
FGRN[31]	VGGNet	-	.839	.786	.918	.043	.822	.783	.871	.084	.867	.852	.954	.041	.737	.660	.904	.037	-	-	-	-	-	-	-	-
RCRNet[70]	ResNet50	53.79	.884	.845	.947	.028	.873	.850	.902	.055	.933	.925	.971	.020	.829	.747	.901	.038	.726	.601	.773	.078	0.692	.550	.760	.102
SSAV[8]	ResNet50	-	.891	.857	.945	.029	.880	.856	.922	.043	.944	.940	.983	.018	.934	.797	.922	.024	-	-	-	-	-	-	-	-
PCSA[14]	MobileNetV3	2.63	.901	.878	.961	.023	.874	.847	.914	.043	.946	.941	.984	.016	.887	.850	.940	.019	.725	.590	.759	.077	-	-	-	-
DCFNet[77]	ResNet101	71.66	.914	.899	.970	.016	.883	.853	.910	.041	**.952**	.953	**.990**	**.010**	.903	.870	.953	.013	.729	.612	.781	.065	.708	.601	.791	**.077**
FSNet[21]	ResNet50	102.3	.922	.909	.972	.019	.875	.867	.918	.048	-	-	-	-	.849	.773	.920	.023	.760	.637	.796	.063	.732	.623	.789	.088
CoSTFormer[37]	ResNet50	-	.923	.906	**.978**	**.014**	-	-	-	-	-	-	-	-	.874	.813	.943	.018	.779	.667	.819	.060	.730	.614	.777	.082
UFO[58]	VGG16	55.92	.918	.906	**.978**	.015	.858	.868	.911	.051	.926	.917	.969	.020	.888	.850	.951	.014	.747	.626	.799	.063	.711	.605	.773	.088
VSCode-T	Swin-T	54.09	**.930**	**.913**	.970	**.014**	**.891**	**.880**	**.923**	**.037**	**.952**	**.954**	.989	**.010**	**.943**	**.937**	**.984**	**.008**	**.792**	**.696**	**.831**	**.053**	**.738**	**.631**	**.797**	.078
VSCode-S	Swin-S	74.72	**.936**	.922	.973	**.013**	**.905**	**.902**	.939	**.029**	.955	**.957**	**.991**	**.009**	.946	**.937**	**.984**	**.008**	.800	.710	.835	.052	.758	.666	.815	.071
VSCode-B	Swin-B	117.41	**.936**	**.923**	**.974**	.014	.900	.901	**.940**	.031	**.957**	.948	**.991**	**.009**	**.947**	**.937**	**.984**	**.008**	**.812**	**.728**	**.847**	**.047**	**.769**	**.690**	**.845**	**.069**

Table 8. **Quantitative comparison of our proposed VSCode with other 9 SOTA VSOD methods on six benchmark datasets.**

Summary	Backbone	Params (M)	COD10K[6]				NC4K[44]				CAMO[26]			
			S_m	F_m	E_m	M	S_m	F_m	E_m	M	S_m	F_m	E_m	M
SINet[6]	ResNet50	48.95	.771	.676	.868	.051	.808	.775	.883	.058	.752	.706	.831	.100
CRLS[44]	ResNet50	-	.805	.732	.892	.037	.840	.815	.907	.048	.787	.753	.854	.080
MGL[74]	ResNet50	63.60	.814	.738	.890	.035	-	-	-	-	.776	.741	.842	.089
UJSC[28]	ResNet50	121.63	.817	.750	.902	.033	.856	.835	.920	.040	.803	.775	.867	.071
SegMar[24]	ResNet50	56.21	.833	.755	.907	.034	.841	.827	.907	.046	.816	.803	.884	.071
FEDER[15]	ResNet50	44.13	.822	.768	.905	.032	.847	.833	.915	.044	.802	.789	.873	.071
VSCode-T	Swin-T	54.09	**.847**	**.795**	.925	**.028**	**.874**	**.853**	**.930**	**.038**	**.838**	**.821**	**.909**	**.060**
EVP[40]	SegFormer-B4	64.52	.845	.794	.926	.029	.874	.855	.933	.039	.849	.833	.918	.058
VSCode-S	Swin-S	74.72	**.869**	**.827**	**.942**	**.024**	**.891**	**.878**	**.944**	**.032**	**.873**	**.861**	**.938**	**.047**
FSPNet	DeiT-B	274.24	.851	.794	.931	.026	.879	.859	.937	.035	.856	.846	.928	.050
VSCode-B	Swin-B	117.41	**.876**	**.838**	**.947**	**.022**	**.902**	**.892**	**.952**	**.029**	**.882**	**.875**	**.940**	**.044**

Table 9. **Quantitative comparison of our proposed VSCode with other 8 SOTA RGB COD methods on three benchmark datasets.**

Summary	Backbone	Params (M)	CAD [1]				MoCA-Mask[4]			
			S_m	F_m	E_m	M	S_m	F_m	E_m	M
PNS-Net[20]	Res2Net50	26.87	.671	.473	.787	.054	.514	.068	.599	.030
RCRNet[70]	ResNet50	53.79	.664	.405	.786	.051	.559	.170	.593	.025
MG[72]	VGG	-	.608	.378	.673	.069	.500	.138	.514	.078
SLT-Net[4]	PVT	164.68	.715	.542	**.823**	.036	.624	.327	.768	.019
VSCode-T	Swin-T	54.09	**.757**	**.659**	.808	**.034**	**.650**	**.339**	**.787**	**.013**
VSCode-S	Swin-S	74.72	.790	**.680**	**.853**	**.026**	.665	.386	.796	.012
VSCode-B	Swin-B	117.41	**.791**	.678	.852	.027	**.678**	**.430**	**.832**	**.011**

Table 10. **Quantitative comparison of our proposed VSCode with other 4 SOTA VCOD methods on two benchmark datasets.**

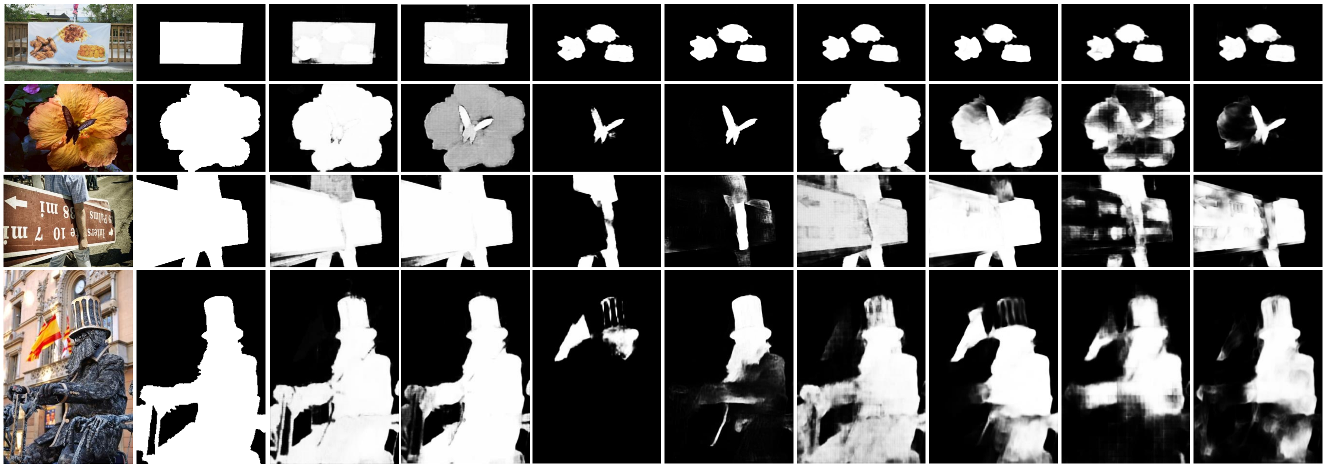

Figure 2. **Qualitative comparison of our model against state-of-the-art RGB SOD methods.** (GT: ground truth.)

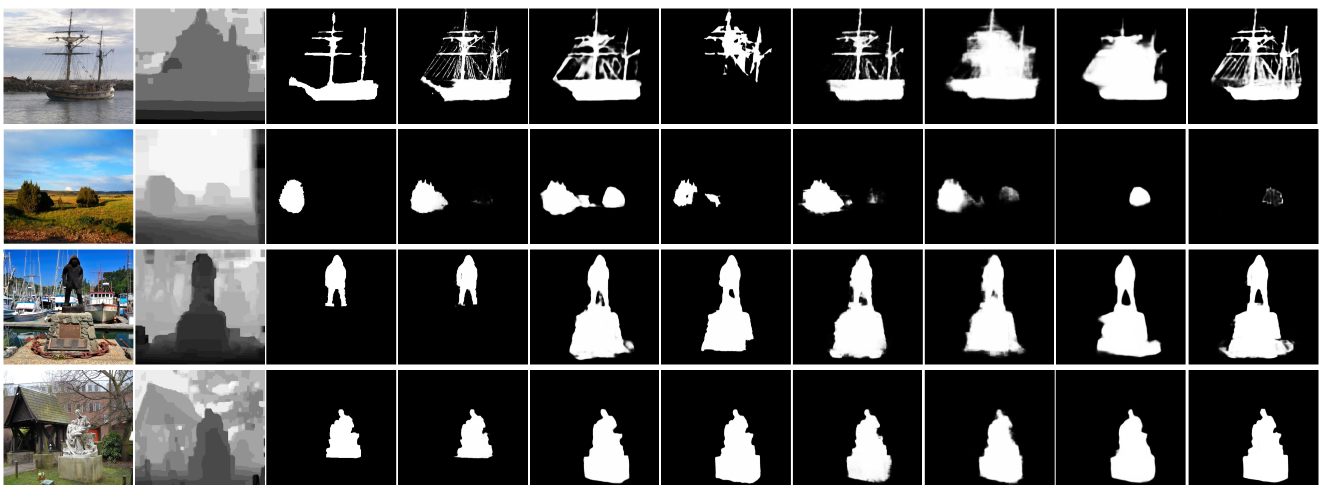

Figure 3. **Qualitative comparison of our model against state-of-the-art RGB-D SOD methods.** (GT: ground truth.)

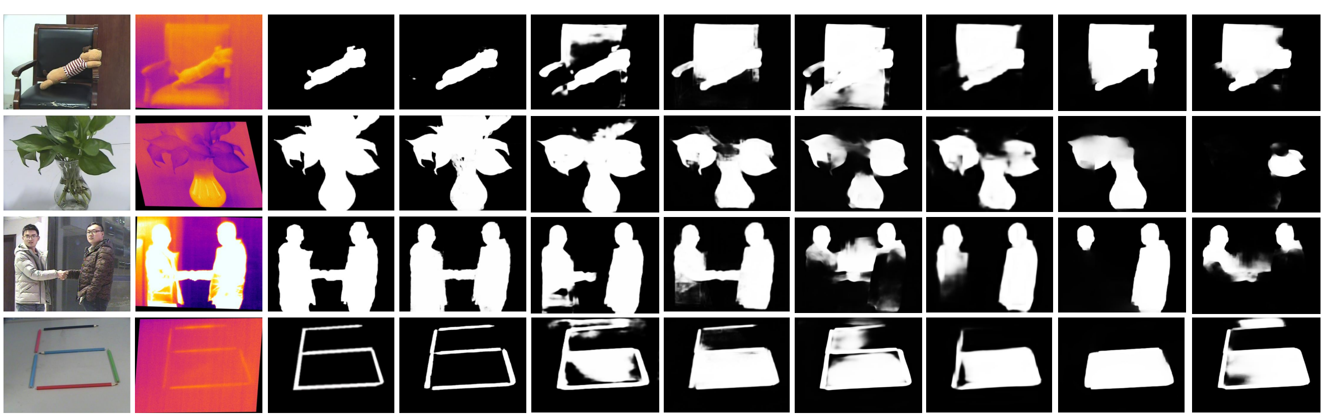

Figure 4. **Qualitative comparison of our model against state-of-the-art RGB-T SOD methods.** (GT: ground truth.)

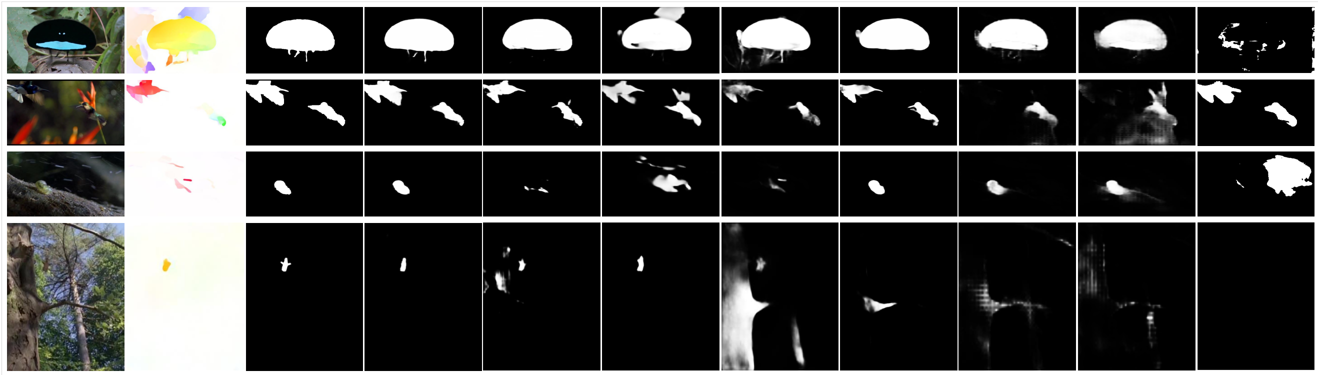

Figure 5. **Qualitative comparison of our model against state-of-the-art VSOD methods.**(GT: ground truth.)

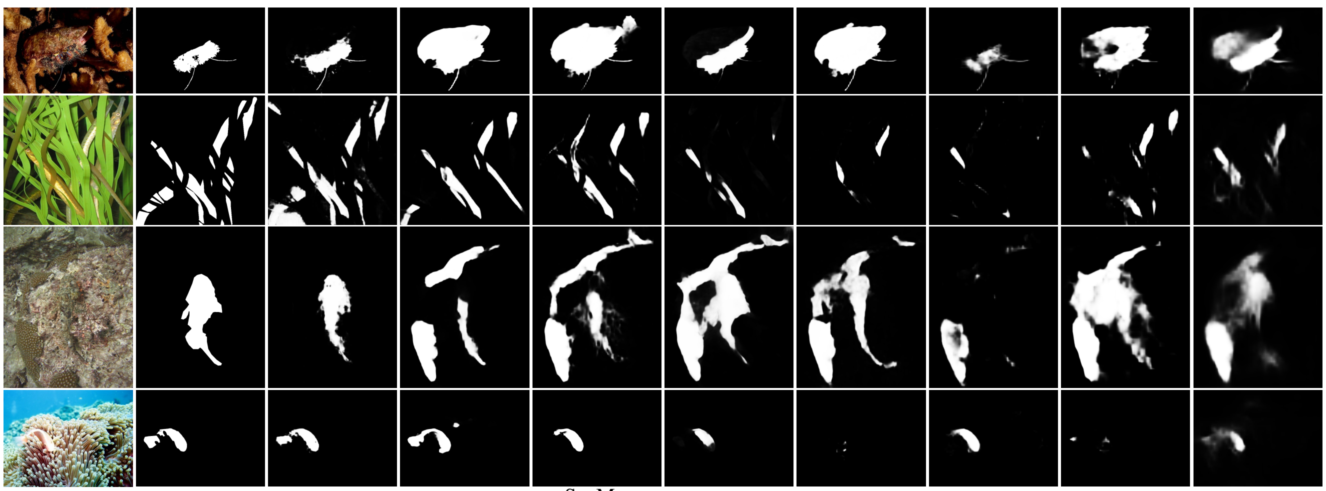

Figure 6. **Qualitative comparison of our model against state-of-the-art RGB COD methods.**(GT: ground truth.)

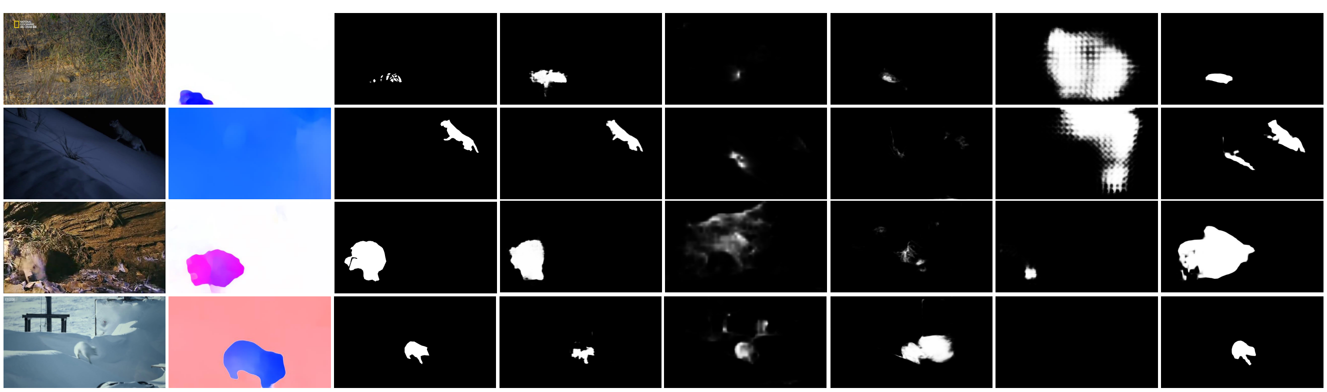

Figure 7. **Qualitative comparison of our model against state-of-the-art VCOD methods.**(GT: ground truth.)

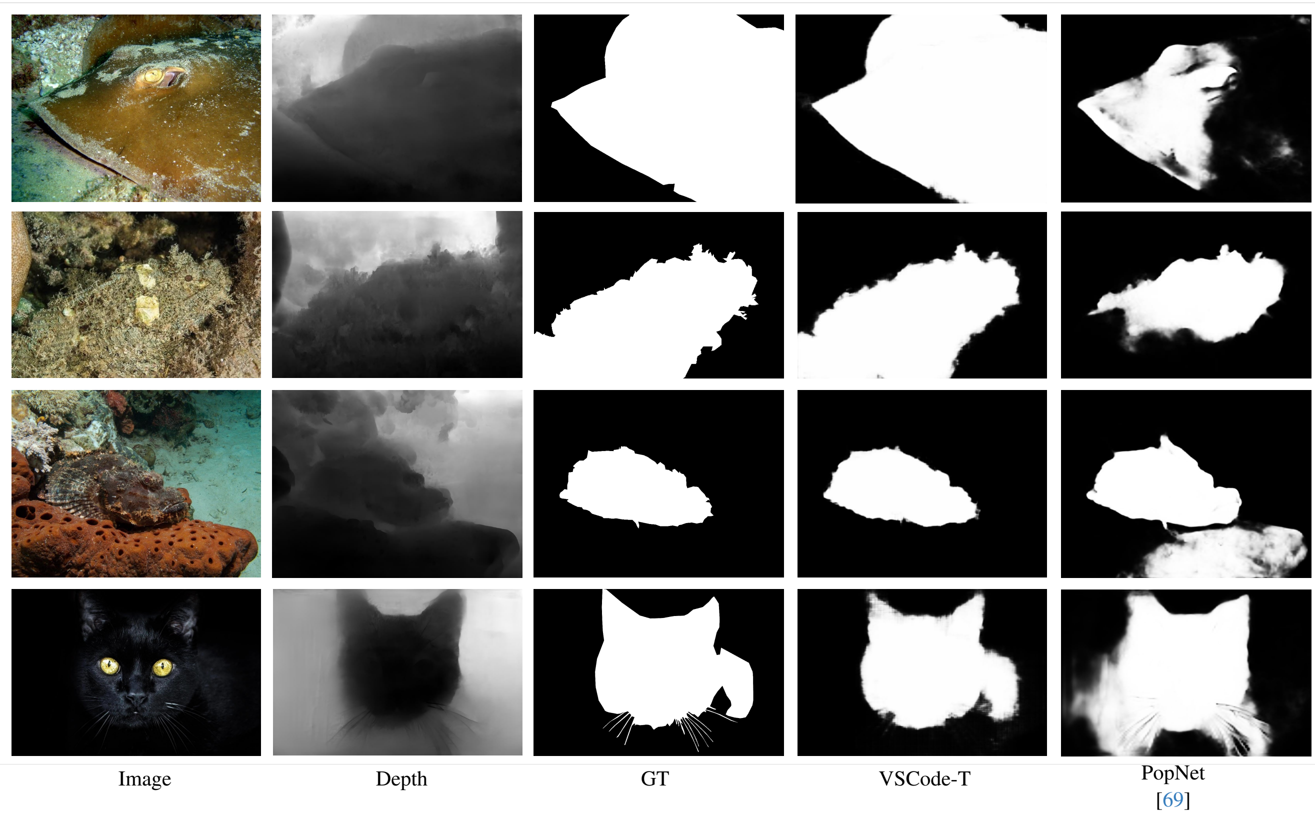

Figure 8. **Qualitative comparison of our model against state-of-the-art RGB-D COD methods.**(GT: ground truth.)

References

[1] Pia Bideau and Erik Learned-Miller. It's moving! a probabilistic model for causal motion segmentation in moving camera videos. In *ECCV*, pages 433–449. Springer, 2016. 2, 5

[2] Gang Chen, Feng Shao, Xiongli Chai, Hangwei Chen, Qiuping Jiang, Xiangchao Meng, and Yo-Sung Ho. Cgmdrnet: Cross-guided modality difference reduction network for rgb-t salient object detection. *TCSVT*, 32(9):6308–6323, 2022. 5

[3] Bowen Cheng, Ishan Misra, Alexander G Schwing, Alexander Kirillov, and Rohit Girdhar. Masked-attention mask transformer for universal image segmentation. In *CVPR*, pages 1290–1299, 2022. 2

[4] Xuelian Cheng, Huan Xiong, Deng-Ping Fan, Yiran Zhong, Mehrtash Harandi, Tom Drummond, and Zongyuan Ge. Implicit motion handling for video camouflaged object detection. In *CVPR*, pages 13864–13873, 2022. 5, 7

[5] Runmin Cong, Kepu Zhang, Chen Zhang, Feng Zheng, Yao Zhao, Qingming Huang, and Sam Kwong. Does thermal really always matter for rgb-t salient object detection? *TMM*, 2022. 5, 6

[6] Deng-Ping Fan, Ge-Peng Ji, Guolei Sun, Ming-Ming Cheng, Jianbing Shen, and Ling Shao. Camouflaged object detection. In *CVPR*, pages 2777–2787, 2020. 3, 5, 7

[7] Deng-Ping Fan, Zheng Lin, Zhao Zhang, Menglong Zhu, and Ming-Ming Cheng. Rethinking rgb-d salient object detection: Models, data sets, and large-scale benchmarks. *IEEE TNNLS*, 32(5):2075–2089, 2020. 3, 4

[8] Deng-Ping Fan, Wenguan Wang, Ming-Ming Cheng, and Jianbing Shen. Shifting more attention to video salient object detection. In *CVPR*, pages 8554–8564, 2019. 5, 7

[9] Deng-Ping Fan, Yingjie Zhai, Ali Borji, Jufeng Yang, and Ling Shao. Bbs-net: Rgb-d salient object detection with a bifurcated backbone strategy network. In *ECCV*, pages 275–292, 2020. 4

[10] Mengyang Feng, Huchuan Lu, and Errui Ding. Attentive feedback network for boundary-aware salient object detection. In *CVPR*, pages 1623–1632, 2019. 4

[11] Keren Fu, Deng-Ping Fan, Ge-Peng Ji, and Qijun Zhao. Jl-dcf: Joint learning and densely-cooperative fusion framework for rgb-d salient object detection. In *CVPR*, pages 3052–3062, 2020. 4

[12] Shang-Hua Gao, Ming-Ming Cheng, Kai Zhao, Xin-Yu Zhang, Ming-Hsuan Yang, and Philip Torr. Res2net: A new multi-scale backbone architecture. *IEEE TPAMI*, 43(2):652–662, 2019. 4

[13] Shang-Hua Gao, Yong-Qiang Tan, Ming-Ming Cheng, Chengze Lu, Yunpeng Chen, and Shuicheng Yan. Highly efficient salient object detection with 100k parameters. In *ECCV*, pages 702–721, 2020. 4

[14] Yuchao Gu, Lijuan Wang, Ziqin Wang, Yun Liu, Ming-Ming Cheng, and Shao-Ping Lu. Pyramid constrained self-attention network

for fast video salient object detection. In *AAAI*, volume 34, pages 10869–10876, 2020. 5
[15] Chunming He, Kai Li, Yachao Zhang, Longxiang Tang, Yulun Zhang, Zhenhua Guo, and Xiu Li. Camouflaged object detection with feature decomposition and edge reconstruction. In *CVPR*, pages 22046–22055, 2023. 3, 5, 7
[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 4
[17] Zhou Huang, Hang Dai, Tian-Zhu Xiang, Shuo Wang, Huai-Xin Chen, Jie Qin, and Huan Xiong. Feature shrinkage pyramid for camouflaged object detection with transformers. In *CVPR*, pages 5557–5566, 2023. 4
[18] Fushuo Huo, Xuegui Zhu, Lei Zhang, Qifeng Liu, and Yu Shu. Efficient context-guided stacked refinement network for rgb-t salient object detection. *IEEE TCSVT*, 32(5):3111–3124, 2021. 5, 6
[19] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *ICCV*, pages 2462–2470, 2017. 1
[20] Ge-Peng Ji, Yu-Cheng Chou, Deng-Ping Fan, Geng Chen, Huazhu Fu, Debesh Jha, and Ling Shao. Progressively normalized self-attention network for video polyp segmentation. In *MICCAI*, pages 142–152. Springer, 2021. 5, 7
[21] Ge-Peng Ji, Keren Fu, Zhe Wu, Deng-Ping Fan, Jianbing Shen, and Ling Shao. Full-duplex strategy for video object segmentation. In *ICCV*, pages 4922–4933, 2021. 5, 7
[22] Wei Ji, Jingjing Li, Shuang Yu, Miao Zhang, Yongri Piao, Shunyu Yao, Qi Bi, Kai Ma, Yefeng Zheng, Huchuan Lu, and Li Cheng. Calibrated rgb-d salient object detection. In *CVPR*, pages 9471–9481, June 2021. 4, 6
[23] Wei Ji, Jingjing Li, Miao Zhang, Yongri Piao, and Huchuan Lu. Accurate rgb-d salient object detection via collaborative learning. In *ECCV*, pages 52–69, 2020. 4
[24] Qi Jia, Shuilian Yao, Yu Liu, Xin Fan, Risheng Liu, and Zhongxuan Luo. Segment, magnify and reiterate: Detecting camouflaged objects the hard way. In *CVPR*, pages 4713–4722, 2022. 3, 5, 7
[25] Ran Ju, Ling Ge, Wenjing Geng, Tongwei Ren, and Gangshan Wu. Depth saliency based on anisotropic center-surround difference. In *ICIP*, pages 1115–1119, 2014. 2, 3, 4
[26] Trung-Nghia Le, Tam V Nguyen, Zhongliang Nie, Minh-Triet Tran, and Akihiro Sugimoto. Anabranch network for camouflaged object segmentation. *CVIU*, 184:45–56, 2019. 2, 3, 5
[27] Minhyeok Lee, Chaewon Park, Suhwan Cho, and Sangyoun Lee. Spsn: Superpixel prototype sampling network for rgb-d salient object detection. In *ECCV*, pages 630–647. Springer, 2022. 3, 4, 6
[28] Aixuan Li, Jing Zhang, Yunqiu Lv, Bowen Liu, Tong Zhang, and Yuchao Dai. Uncertainty-aware joint salient object and camouflaged object detection. In *CVPR*, pages 10071–10081, 2021. 3, 5, 7
[29] Fuxin Li, Taeyoung Kim, Ahmad Humayun, David Tsai, and James M Rehg. Video segmentation by tracking many figure-ground segments. In *ICCV*, pages 2192–2199, 2013. 2, 5
[30] Gongyang Li, Zhi Liu, Linwei Ye, Yang Wang, and Haibin Ling. Cross-modal weighting network for rgb-d salient object detection. In *ECCV*, pages 665–681, 2020. 4
[31] Guanbin Li, Yuan Xie, Tianhao Wei, Keze Wang, and Liang Lin. Flow guided recurrent neural encoder for video salient object detection. In *CVPR*, pages 3243–3252, 2018. 5
[32] Guanbin Li and Yizhou Yu. Visual saliency based on multiscale deep features. In *CVPR*, pages 5455–5463, 2015. 4
[33] Siyang Li, Bryan Seybold, Alexey Vorobyov, Xuejing Lei, and C-C Jay Kuo. Unsupervised video object segmentation with motion-based bilateral networks. In *ECCV*, pages 207–223, 2018. 7
[34] Yin Li, Xiaodi Hou, Christof Koch, James M Rehg, and Alan L Yuille. The secrets of salient object segmentation. In *CVPR*, pages 280–287, 2014. 4
[35] Nian Liu, Junwei Han, and Ming-Hsuan Yang. Picanet: Learning pixel-wise contextual attention for saliency detection. In *CVPR*, pages 3089–3098, 2018. 4
[36] Nian Liu, Ziyang Luo, Ni Zhang, and Junwei Han. Vst++: Efficient and stronger visual saliency transformer. *arXiv preprint arXiv:2310.11725*, 2023. 3, 4, 5, 6
[37] Nian Liu, Kepan Nan, Wangbo Zhao, Xiwen Yao, and Junwei Han. Learning complementary spatial–temporal transformer for video salient object detection. *TNNLS*, 2023. 5, 7
[38] Nian Liu, Ni Zhang, Ling Shao, and Junwei Han. Learning selective mutual attention and contrast for rgb-d saliency detection. *IEEE TPAMI*, 44(12):9026–9042, 2021. 3, 4
[39] Nian Liu, Ni Zhang, Kaiyuan Wan, Ling Shao, and Junwei Han. Visual saliency transformer. In *ICCV*, pages 4722–4732, October 2021. 3, 4, 6
[40] Weihuang Liu, Xi Shen, Chi-Man Pun, and Xiaodong Cun. Explicit visual prompting for universal foreground segmentations. *arXiv preprint arXiv:2305.18476*, 2023. 2, 4, 5, 6
[41] Yi Liu, Qiang Zhang, Dingwen Zhang, and Jungong Han. Employing deep part-object relationships for salient object detection. In *ICCV*, pages 1232–1241, 2019. 4
[42] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, 2021. 2, 3, 4
[43] Ao Luo, Xin Li, Fan Yang, Zhicheng Jiao, Hong Cheng, and Siwei Lyu. Cascade graph neural networks for rgb-d salient object detection. In *ECCV*, pages 346–364, 2020. 4
[44] Yunqiu Lv, Jing Zhang, Yuchao Dai, Aixuan Li, Bowen Liu, Nick Barnes, and Deng-Ping Fan. Simultaneously localize, segment and rank the camouflaged objects. In *CVPR*, pages 11591–11601, 2021. 3, 5, 7
[45] Vida Movahedi and James H Elder. Design and perceptual validation of performance measures for salient object segmentation. In

CVPR, pages 49–56, 2010. 4
[46] Yuzhen Niu, Yujie Geng, Xueqing Li, and Feng Liu. Leveraging stereopsis for saliency analysis. In *CVPR*, pages 454–461, 2012. 3, 4
[47] Peter Ochs, Jitendra Malik, and Thomas Brox. Segmentation of moving objects by long term video analysis. *PAMI*, 36(6):1187–1200, 2013. 5
[48] Youwei Pang, Lihe Zhang, Xiaoqi Zhao, and Huchuan Lu. Hierarchical dynamic filtering network for rgb-d salient object detection. In *ECCV*, pages 235–252, 2020. 4
[49] Youwei Pang, Xiaoqi Zhao, Tian-Zhu Xiang, Lihe Zhang, and Huchuan Lu. Zoom in and out: A mixed-scale triplet network for camouflaged object detection. In *CVPR*, pages 2160–2170, 2022. 4, 7
[50] Youwei Pang, Xiaoqi Zhao, Lihe Zhang, and Huchuan Lu. Multi-scale interactive network for salient object detection. In *CVPR*, pages 9413–9422, 2020. 4
[51] Youwei Pang, Xiaoqi Zhao, Lihe Zhang, and Huchuan Lu. Caver: Cross-modal view-mixed transformer for bi-modal salient object detection. *TIP*, 32:892–904, 2023. 3, 4, 5, 6
[52] Houwen Peng, Bing Li, Weihua Xiong, Weiming Hu, and Rongrong Ji. Rgbd salient object detection: A benchmark and algorithms. In *ECCV*, pages 92–109, 2014. 3, 4
[53] Federico Perazzi, Jordi Pont-Tuset, Brian McWilliams, Luc Van Gool, Markus Gross, and Alexander Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *CVPR*, pages 724–732, 2016. 5
[54] Yongri Piao, Wei Ji, Jingjing Li, Miao Zhang, and Huchuan Lu. Depth-induced multi-scale recurrent attention network for saliency detection. In *ICCV*, pages 7254–7263, 2019. 3, 4
[55] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *ICCV*, pages 12179–12188, 2021. 3
[56] Hongmei Song, Wenguan Wang, Sanyuan Zhao, Jianbing Shen, and Kin-Man Lam. Pyramid dilated deeper convlstm for video salient object detection. In *ECCV*, pages 715–731, 2018. 5, 7
[57] Kechen Song, Liming Huang, Aojun Gong, and Yunhui Yan. Multiple graph affinity interactive network and a variable illumination dataset for rgbt image salient object detection. *IEEE TCSVT*, 2022. 5, 6
[58] Yukun Su, Jingliang Deng, Ruizhou Sun, Guosheng Lin, Hanjing Su, and Qingyao Wu. A unified transformer framework for group-based segmentation: Co-segmentation, co-saliency detection and video salient object detection. *TMM*, 2023. 5, 7
[59] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *ICML*, pages 10347–10357. PMLR, 2021. 4
[60] Zhengzheng Tu, Zhun Li, Chenglong Li, Yang Lang, and Jin Tang. Multi-interactive dual-decoder for rgb-thermal salient object detection. *IEEE TIP*, 30:5678–5691, 2021. 5, 6
[61] Zhengzheng Tu, Yan Ma, Zhun Li, Chenglong Li, Jieming Xu, and Yongtao Liu. Rgbt salient object detection: A large-scale dataset and benchmark. *IEEE TMM*, 2022. 2, 5
[62] Zhengzheng Tu, Tian Xia, Chenglong Li, Xiaoxiao Wang, Yan Ma, and Jin Tang. Rgb-t image saliency detection via collaborative graph learning. *IEEE TMM*, 22(1):160–173, 2019. 5
[63] Guizhao Wang, Chenglong Li, Yunpeng Ma, Aihua Zheng, Jin Tang, and Bin Luo. Rgb-t saliency detection benchmark: Dataset, baselines, analysis and a novel approach. In *IJIG*, pages 359–369. Springer, 2018. 5
[64] Jie Wang, Kechen Song, Yanqi Bao, Liming Huang, and Yunhui Yan. Cgfnet: Cross-guided fusion network for rgb-t salient object detection. *IEEE TCSVT*, 32(5):2949–2961, 2021. 5, 6
[65] Lijun Wang, Huchuan Lu, Yifan Wang, Mengyang Feng, Dong Wang, Baocai Yin, and Xiang Ruan. Learning to detect salient objects with image-level supervision. In *CVPR*, pages 136–145, 2017. 2, 4
[66] Wenguan Wang, Jianbing Shen, and Ling Shao. Consistent video saliency using local gradient flow optimization and global refinement. *TIP*, 24(11):4185–4196, 2015. 5
[67] Yi Wang, Ruili Wang, Xin Fan, Tianzhu Wang, and Xiangjian He. Pixels, regions, and objects: Multiple enhancement for salient object detection. In *CVPR*, pages 10031–10040, 2023. 4, 6
[68] Jun Wei, Shuhui Wang, Zhe Wu, Chi Su, Qingming Huang, and Qi Tian. Label decoupling framework for salient object detection. In *CVPR*, pages 13025–13034, 2020. 4
[69] Zongwei Wu, Danda Pani Paudel, Deng-Ping Fan, Jingjing Wang, Shuo Wang, Cédric Demonceaux, Radu Timofte, and Luc Van Gool. Source-free depth for object pop-out. In *ICCV*, pages 1032–1042, 2023. 8
[70] Pengxiang Yan, Guanbin Li, Yuan Xie, Zhen Li, Chuan Wang, Tianshui Chen, and Liang Lin. Semi-supervised video salient object detection using pseudo-labels. In *ICCV*, pages 7284–7293, 2019. 5, 7
[71] Qiong Yan, Li Xu, Jianping Shi, and Jiaya Jia. Hierarchical saliency detection. In *CVPR*, pages 1155–1162, 2013. 4
[72] Charig Yang, Hala Lamdouar, Erika Lu, Andrew Zisserman, and Weidi Xie. Self-supervised video object segmentation by motion grouping. In *ICCV*, pages 7177–7188, 2021. 5, 7
[73] Chuan Yang, Lihe Zhang, Huchuan Lu, Xiang Ruan, and Ming-Hsuan Yang. Saliency detection via graph-based manifold ranking. In *CVPR*, pages 3166–3173, 2013. 4
[74] Qiang Zhai, Xin Li, Fan Yang, Chenglizhao Chen, Hong Cheng, and Deng-Ping Fan. Mutual graph learning for camouflaged object detection. In *CVPR*, pages 12997–13007, 2021. 5, 7
[75] Jing Zhang, Deng-Ping Fan, Yuchao Dai, Xin Yu, Yiran Zhong, Nick Barnes, and Ling Shao. Rgb-d saliency detection via cascaded mutual information minimization. In *ICCV*, 2021. 3, 4, 6
[76] Miao Zhang, Sun Xiao Fei, Jie Liu, Shuang Xu, Yongri Piao, and Huchuan Lu. Asymmetric two-stream architecture for accurate rgb-d saliency detection. In *ECCV*, pages 374–390, 2020. 4

[77] Miao Zhang, Jie Liu, Yifei Wang, Yongri Piao, Shunyu Yao, Wei Ji, Jingjing Li, Huchuan Lu, and Zhongxuan Luo. Dynamic context-sensitive filtering network for video salient object detection. In *ICCV*, pages 1553–1563, 2021. 5, 7

[78] Qiang Zhang, Nianchang Huang, Lin Yao, Dingwen Zhang, Caifeng Shan, and Jungong Han. Rgb-t salient object detection via fusing multi-level cnn features. *IEEE TIP*, 29:3321–3335, 2019. 5

[79] Jia-Xing Zhao, Jiang-Jiang Liu, Deng-Ping Fan, Yang Cao, Jufeng Yang, and Ming-Ming Cheng. Egnet:edge guidance network for salient object detection. In *ICCV*, pages 8779–8788, 2019. 4

[80] Xiaoqi Zhao, Youwei Pang, Lihe Zhang, Huchuan Lu, and Lei Zhang. Suppress and balance: A simple gated network for salient object detection. In *ECCV*, pages 35–51, 2020. 4, 6

[81] Dehua Zheng, Xiaochen Zheng, Laurence T Yang, Yuan Gao, Chenlu Zhu, and Yiheng Ruan. Mffn: Multi-view feature fusion network for camouflaged object detection. In *WACV*, pages 6232–6242, 2023. 4

[82] Yijie Zhong, Bo Li, Lv Tang, Senyun Kuang, Shuang Wu, and Shouhong Ding. Detecting camouflaged object in frequency domain. In *CVPR*, pages 4504–4513, 2022. 4

[83] Huajun Zhou, Xiaohua Xie, Jian-Huang Lai, Zixuan Chen, and Lingxiao Yang. Interactive two-stream decoder for accurate and fast saliency detection. In *CVPR*, pages 9141–9150, 2020. 4

[84] Tao Zhou, Huazhu Fu, Geng Chen, Yi Zhou, Deng-Ping Fan, and Ling Shao. Specificity-preserving rgb-d saliency detection. In *ICCV*, 2021. 4

[85] Wujie Zhou, Qinling Guo, Jingsheng Lei, Lu Yu, and Jenq-Neng Hwang. Ecffnet: Effective and consistent feature fusion network for rgb-t salient object detection. *IEEE TCSVT*, 32(3):1224–1235, 2021. 5

[86] Mingchen Zhuge, Deng-Ping Fan, Nian Liu, Dingwen Zhang, Dong Xu, and Ling Shao. Salient object detection via integrity learning. *IEEE TPAMI*, 2022. 4, 6