2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

AutoLay: Benchmarking amodal layout estimation for autonomous driving

Kaustubh Mani*[1,2], N. Sai Shankar*[1], Krishna Murthy Jatavallabhula[3,4], and K. Madhava Krishna[1,2]

[1]Robotics Research Center, KCIS, [2]IIIT Hyderabad, [3]Mila - Quebec AI Institute, Montreal, [4]Université de Montréal

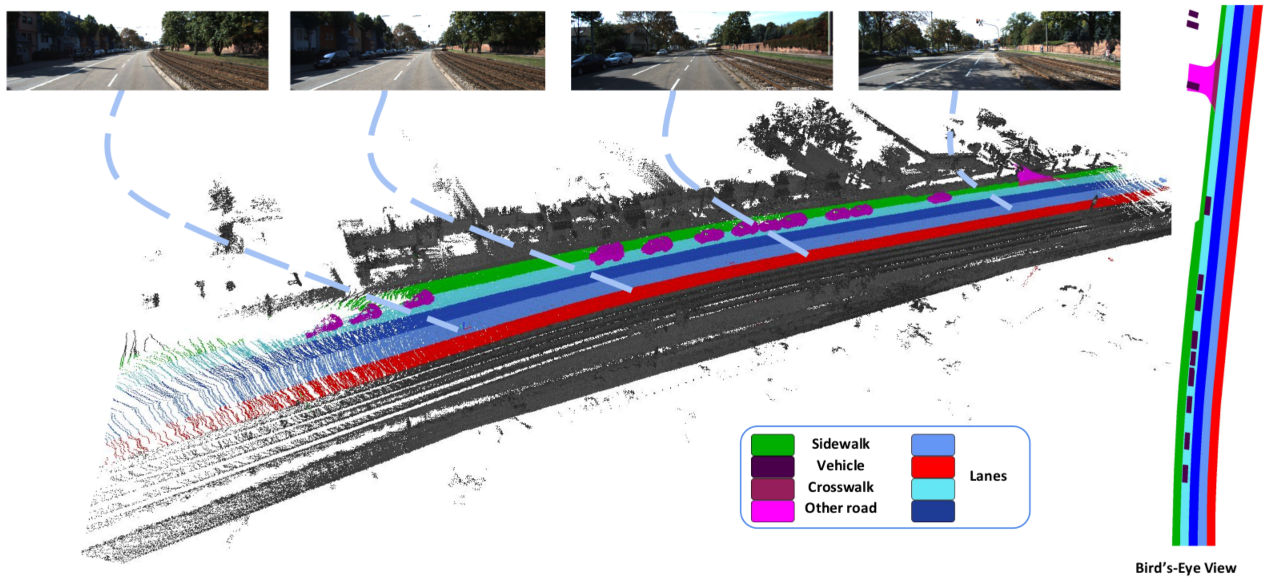

Fig. 1: **Amodal layout estimation** is the task of estimating a semantic occupancy map in bird's eye view, given a monocular image or video. The term *amodal* implies that we estimate occupancy and semantic labels even for parts of the world that are occluded in image space. In this work, we introduce ***AutoLay***, a new dataset and benchmark for this task. *AutoLay* provides annotations in 3D, in bird's eye view, and in image space. A sample annotated sequence (from the KITTI dataset [1]) is shown below. We provide high quality labels for sidewalks, vehicles, crosswalks, and lanes. We evaluate several approaches on sequences from the KITTI [1] and Argoverse [2] datasets.

***Abstract*— Given an image or a video captured from a monocular camera, amodal layout estimation is the task of predicting semantics and occupancy in bird's eye view. The term *amodal* implies we also reason about entities in the scene that are occluded or truncated in image space. While several recent efforts have tackled this problem, there is a lack of standardization in task specification, datasets, and evaluation protocols. We address these gaps with *AutoLay*, a dataset and benchmark for amodal layout estimation from monocular images. *AutoLay* encompasses driving imagery from two popular datasets: KITTI [1] and Argoverse [2]. In addition to fine-grained attributes such as lanes, sidewalks, and vehicles, we also provide semantically annotated 3D point clouds. We implement several baselines and bleeding edge approaches, and release our data and code.[1].**

I. Introduction

Commercial interest in autonomous driving has led to the emergence of several interesting and challenging problems, particularly in terms of perception. In this work, we focus on the problem of *amodal scene layout estimation* (introduced in [3]). *Amodal* perception, as studied by cognitive scientists, refers to the phenomenon of "imagining" or "hallucinating" parts of the scene that do not result in sensory stimulation. In the context of urban driving, we formulate the *amodal layout estimation* task as predicting a bird's eye view semantic map from monocular images that explicitly reasons about entities in the scene that are not visible from the image(s).

Amodal scene layout estimation is an emerging area of study where prior efforts [3], [4], [5] have only focused on amodal completion for *coarse* categories such as roads and sidewalks. While recent efforts have incorporated dynamic objects [3], they fall short of modeling higher-level behaviors in urban driving scenarios such as lane-keeping, merging, etc. Further, each of these efforts [3], [4], [5] lack consensus in problem specifications and evaluation protocols.

We address both the above issues. We first formalize the problem of *amodal scene layout estimation* that unifies existing work and extends the task to several fine-grained categories. To foster research in this area, we provide *AutoLay*, a dataset comprising of over 16000 images over a distance of 12 kilometers, annotated with fine-grained *amodal* completions in 3D as well as in bird's eye view. *AutoLay* includes

* denotes equal contribution
Corresponding author: kaustubh3095@gmail.com
[1]Project page: https://hbutsuak95.github.io/AutoLay/

44 sequences from KITTI [1] and 89 sequences from the Argoverse [2] datasets. We implement several baselines and prior art, and open-source our code, models, and data [3], [4], [6]. We additionally propose *VideoLayout*, a simple approach that achieves top performance on the *AutoLay* benchmark by leveraging temporal information across image sequences, advancing the state-of-the-art of amodal layout estimation.

II. Related work

Interpreting 3D scenes from an RGB image has been a longstanding challenge for computer vision systems. In this section, we enlist a few approaches to *amodal* perception. For a comprehensive survey of general 3D scene understanding, we refer the interested reader to Ozyecsil *et al.* [7].

***Amodal perception for indoor scene understanding**:* In the context of indoor 3D scene understanding, a few approaches to *amodal* perception have been proposed [8], [9]. While these approaches use coarse voxel grids to represent objects, we instead focus on an orthographic top-down view that restricts semantic understanding to road regions. We note that most of the semantic understanding necessary for driving-related tasks can be obtained on the road regions, making this simplification attractive.

***Amodal perception for autonomous driving**:* To the best of our knowledge, there are no previous works that provide a unified approach to obtain an amodal scene layout estimation in bird's eye view for fine-grained static and dynamic classes like lanes, sidewalks, vehicles, etc. Existing approaches [4], [5], [10], [11] reason about "coarse-grained" static classes such as roads and sidewalks. Schulter *et al.* [5] obtain occlusion-aware bird's eye view road layouts by hallucinating behind objects tagged as *foreground* in an image. Wang *et al.* [10] develop a parametric representation of road scenes that reason about the number of lanes, width of each lane, presence of an intersection, etc. Lu *et al.* [4] use a variational autoencoder to predict road, sidewalk, terrain, and non free-space in bird's eye view. Lu *et al.* [11] also perform vehicle shape completion, albeit independent of the static layout estimation task. Garnett *et al.* [12] predict static road and lane layouts by estimating lane boundaries in 3D.

For dynamic scene understanding, several approaches aim to detect object layouts in 3D. While some of these [13], [14], [15] combine information from images and lidar, others [16], [17], [18] operate by converting images to bird's eye view representations, followed by object detection. However, these techniques are often developed independently of static layout estimation methods.

In earlier work, we presented MonoLayout [3]- perhaps the first approach to amodally reason about the static and dynamic layout of an urban driving scene. While the interest in this nascent area is rapidly increasing, the absence of standardized task specification and evaluation have been slowing down research. This forms the primary motivation behind the *AutoLay* dataset and benchmark.

III. Amodal scene layout estimation

We begin by formally defining the task of amodal scene layout estimation. Given an image (or a sequence of images) $\mathcal{I}$, usually in perspective view, we wish to learn a *labeling function* Φ of *all* points within an area $\mathcal{R}$ (usually rectangular) around the camera. The region of interest $\mathcal{R}$ is often in a different view than that of the perspective image. In this paper, we focus on bird's eye view, which refers to a top-down orthographic view of the ground plane. The labeling function Φ must produce a label distribution (over a set of N classes) for world points within the swath $\mathcal{R}$, *regardless of whether or not they are imaged in* $\mathcal{I}$.

In particular, we propose the estimation of the following semantic classes tasks in the context of this problem: road, sidewalk, crosswalk, lanes (ego-lane, other lanes), other road, vehicle. Depending on the exact setup used (single-image vs image-sequences, semantic categories to be estimated, etc.), this enables fair evaluation of all approaches. We defer a detailed description of evaluation protocols and its challenges to Sec. VI-D.

IV. AutoLay Dataset

Over the last decade, several public datasets [1], [19], [20], [21], [2] have enabled massive strides in autonomous driving research. While nearly all the above datasets provide out-of-the-box support (annotations, evaluation utilities, metrics, benchmarking support) for popular tasks like object detection, semantic segmentation, multi-object tracking, trajectory forecasting, none of these currently benchmark amodal scene layout estimation.

Obtaining training data for the task of amodal layout estimation involves multiple challenges. It involves perceiving scene segments that are not even imaged by the camera. While most datasets have lidar scans that often span a larger sensor swath, such lidar beams are not dense enough to perceive thin scene structures. Approaches such as *MonoLayout* [3] compensate for this lack of precision training data by adversarial learning, using OpenStreetMap [22] data. More recently, Wang *et al.* [10] release a dataset that provides a *parametric* bird's eye view representation for images from the KITTI [1] and nuScenes [20] datasets, but the top-view representations are synthesized views of a simulated parametric surface model. In summary, there is currently no dataset that provides off-the-shelf support for benchmarking amodal layout estimation. We fill this void with *AutoLay*, a dataset for amodal layout estimation in bird's eye view (Fig. 1).

A. Dataset overview

We use 44 video sequences from the KITTI Raw dataset [1] in *AutoLay*. We provide per-frame annotations in perspective, orthographic (bird's eye view), as well as in 3D. Of the 44 annotated sequences, 24 sequences—containing 10705 images—are used for training. The other 20 sequences—comprising 5626 images—form the test set. This makes for nearly 16K annotated images, across a distance of 12 Km, and a variety of urban scenarios (*residential*, *city*, *road*) (*cf.* Fig 2). The semantic classes considered in this dataset are *road*, *sidewalk*, *vehicle*, *crosswalk*, and *lane*. Each *lane* segment is provided a unique id, which we classify

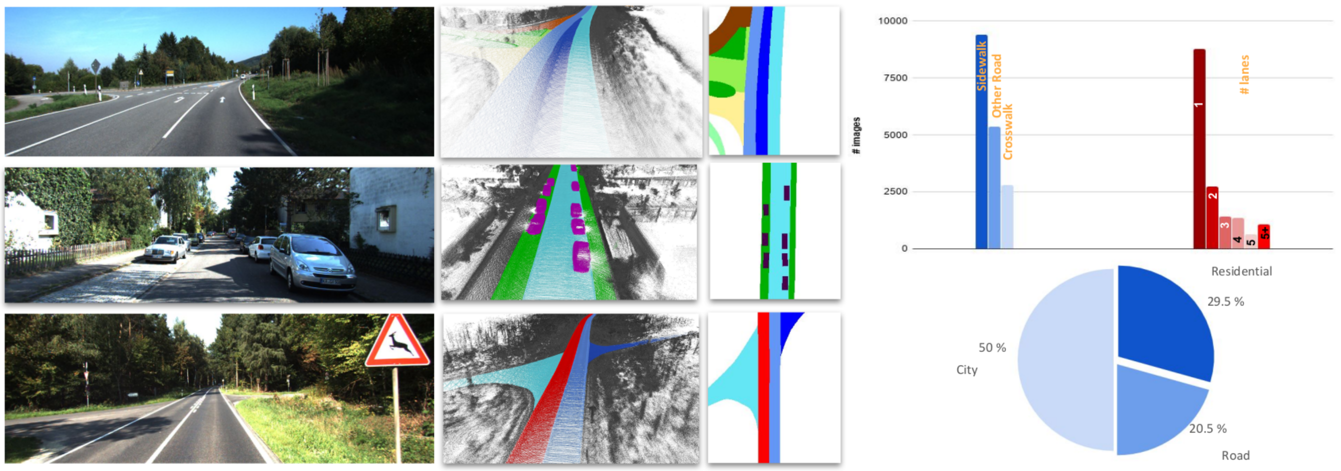

Fig. 2: **Dataset**: (*Left to right*) Sample images from the *KITTI* split of *AutoLay*. Corresponding annotated lidar pointclouds. Amodal scene layout in bird's eye view. (*Last column*) Distribution of semantic classes (bar plot), and scene types (pie chart).

further[2] The *lane* class is further classified as *ego-lane* and *other lane*. We also have an *other road* class for road areas that do not fall under any of the above categories.

B. Data annotation

Since annotating amodal layout from a single image is a hard task even for professional annotators, we leverage multiple sources of information to guide this procedure. We first build a point cloud map of the entire scene, annotate the map (in full 3D), and then generate the other views (bird's eye view). To build a map of the entire area, we use the lidar scans corresponding to each frame and register them to a global view. We use the precise odometry information provided in KITTI for registration, to be robust to outliers, as many scenes contain dynamic objects (vehicles/pedestrians). To reduce manual effort, we semi-automate the data annotation pipeline to the extent possible. As a first step, we identify lane markers by choosing a range of remission values of the lidar and thresholding the 3D map. Once a set of lane markers are identified, we tag as *lanes* the area between pairs of lane markers. Manual verification is performed to ensure no discrepancies creep in. Each identified *lane* is labeled with a different *lane id*, unique within a given sequence. Annotators are then presented with this preprocessed map in an orthographic view and asked to amodally complete annotations. Specifically, chunks of missing points in a lane are *filled in* as *lanes*, and so on. Other identified chunks are annotated as *sidewalk*, *crosswalk*, or *other road*. We repurpose the annotation tool from Semantic KITTI [23] for this.

Note that the above annotation procedure can only be used for static parts of the scene. For dynamic scene components, such as vehicles, a 3D bounding box annotation is performed per-frame. This entire annotated sequence is then projected into bird's eye view and semantics are made available in every camera frame in the sequence.

[2]We perform this subclassification for our specific task, by projecting lane segments to bird's eye view.

V. Approach

Learning the mapping function described in Sec. III which maps the perspective RGB images to the static and dynamic layouts in bird's eye view is quite a challenging problem. The problem demands an accurate understanding and modeling of the 3D scene which necessitates the requirement for learning good visual features that can encode the depth and the semantics of the 3D scene. Most approaches [5], [3], [4] to monocular layout estimation operate on a single RGB image input, which often leads to temporally inconsistent predictions.

In this section, we tackle the aforementioned problem by proposing a neural network architecture that takes a sequence of images as input and generates temporally consistent layouts as output.

A. VideoLayout: *Temporally Consistent Layout Estimation*

Given a sequence of RGB images $\mathcal{I}_1, \mathcal{I}_2, ..., \mathcal{I}_t$ as input, our model predicts the posterior distribution $P(\mathcal{S}_t, \mathcal{D}_t | \mathcal{I}_1, \mathcal{I}_2, ..., \mathcal{I}_t)$, where $\mathcal{S}_t$ and $\mathcal{D}_t$ denote the predicted static and dynamic layouts corresponding to image $\mathcal{I}_t$.

Our network architecture consists of four subparts:

- A feature encoder which takes as input a sequence of RGB images $\mathcal{I}_1, \mathcal{I}_2, ..., \mathcal{I}_t$ and generates rich multi-scale features maps $\mathcal{C}_1, \mathcal{C}_2, ..., \mathcal{C}_t$ corresponding to each rgb image.
- A stacked Convolutional LSTM submodule which aggregates image features $\mathcal{C}_1, \mathcal{C}_2, ..., \mathcal{C}_t$ and encodes a temporal representation useful for estimating consistent layouts.
- Similar to [3], we maintain two set of decoder weights corresponding to the static and dynamic decoders. Both decoders share a similar architecture except for the use of sampling layer at the beginning of the static decoder in order to generate smooth looking road and lane layouts. Instead of directly processing the encoded output representation of convLSTM, we instead sample

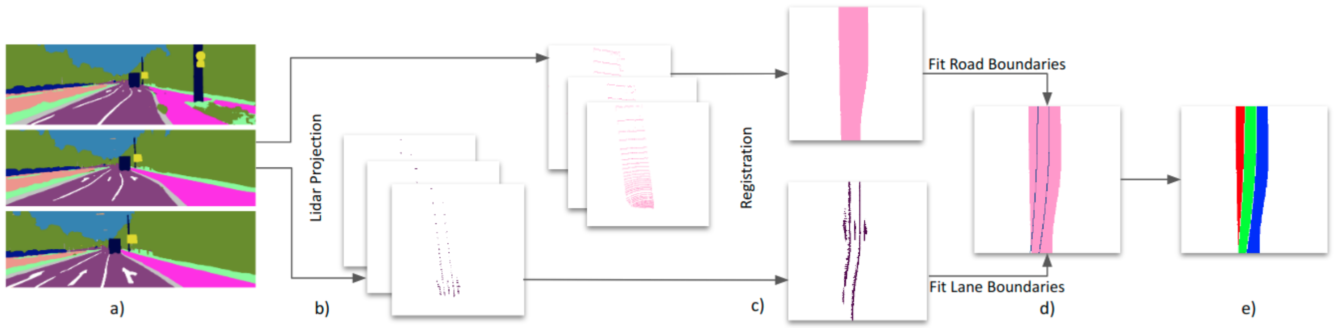

Fig. 3: **Weak Supervision setup**: We show our automated (noisy) label generation scheme herein. Points from the lidar frame are projected down to semantically segmented images (a), to obtain sparse static layouts (b). These sparse layouts are stacked across an image sequence to generate dense static layouts (c). In the next step (d), road and lane boundaries are extracted and combined to obtain lane layouts (e).

from a gaussian distribution with the feature map as the mean values and a fixed standard deviation.

- A Refinement Network, which helps in improving the quality of the output static and dynamic layouts by regularizing the predicted layouts in order to resemble the true data distribution.

B. Consistency Loss

The weights of the *VideoLayout* architecture are updated via backpropagation using the loss function defined in Eq. 4.

$$\mathcal{L}_{sup} = \sum_{i=1}^{N} f(\bar{S}_i, S_i) + f(\bar{D}_i, D_i) \tag{1}$$

$$\mathcal{L}^c_{short} = \sum_{i=1}^{N} \sum_{j=1}^{seqlen-1} f(\bar{S}^j_i, \bar{S}^{j+1}_i) + f(\bar{D}^j_i, \bar{D}^{j+1}_i) \tag{2}$$

$$\mathcal{L}^c_{long} = \sum_{i=1}^{N} \sum_{j=1}^{seqlen-1} \sum_{k=j+2}^{seqlen} f(\bar{S}^j_i, \bar{S}^k_i) + f(\bar{D}^j_i, \bar{D}^k_i) \tag{3}$$

$$\mathcal{L} = \lambda_{sup} * \mathcal{L}_{sup} + \lambda^c_{short} * \mathcal{L}^c_{short} + \lambda^c_{long} * \mathcal{L}^c_{long} \tag{4}$$

Here, $\bar{S}$ and $\bar{D}$ denote the predicted static and dynamic layouts respectively, S and D denote the ground truth (weak/strong) layouts. N is the mini batch size, $seqlen$ is the length of the input image sequence per sample. $f(\cdot,\cdot)$ is the cross-entropy loss function. $\mathcal{L}_{sup}$ is a supervised loss term that penalizes the deviation of the predicted static and dynamic layouts. $\mathcal{L}^c_{short}$ is the short-range consistency loss and $\mathcal{L}^c_{long}$ is the long-range consistency loss. λ_{sup}, λ^c_{short} and λ^c_{long} are the weights corresponding to the supervised, short-range consistency and long-range consistency losses respectively. Finally, $\mathcal{L}$ is the total weighted loss used for backpropagating gradients through the network. [3]. Consistency loss acts as a soft constraint on the layout prediction objective $\mathcal{L}_{sup}$, which enables the model to learn more temporally consistent layout.

[3] $\lambda_{sup} > \lambda^c_{short} >> \lambda^c_{long}$

C. Refinement Network

Similar to [3], we use adversarial regularization in order to restrict the encoder-decoder network into predicting conceivable layouts. For adversarial regularization, we use a Patch Discriminator which takes the output of the encoder-decoder and a sample from the true data distribution. Unlike [3], we make use of ground-truth layouts to do paired training with the discriminator which further improves the performance of the model. Adding the sampling layer in the static decoder converts the Encoder-Decoder network into a generator and further helps in regularization for static layouts.

D. Data Preparation: Weak Supervision

Since the amodal scene layout labels are not provided for almost every autonomous driving dataset, we propose a generic layout generation method to generate noisy or weak supervision layout labels for static classes like roads, lanes, and sidewalks (*cf.* Fig. 3). Our weak supervision layout generation method is similar to the sensor fusion approach adopted in [3]. Further, we provide a solution to obtain weak supervision layout labels for lanes as well.

The process of aggregating / registering the lidar points for several image frames and projecting them to an orthographic view by assuming a flat plane is shown in Fig. 3. To obtain lanes, we perform the registering process for road and lane marker semantic classes. We then obtain lane boundary curves by isolating individual lane boundaries through Density Based Spatial Clustering and fit third order polynomial curves on these individual lane boundaries. From the registered road layout, we estimate the road boundaries and group the road and lane boundaries together to obtain lane contours.

Additionally, we also obtain improved weak supervision layout for roads by addressing cases where there are too many stationary obstacles in a sequence occluding lidar from capturing the road layout. We register lidar points belonging to those stationary obstacles that lie on the road.

VI. Experiments

So far, there has been a lack of consensus on datasets and evaluation protocol for amodal layout estimation. We

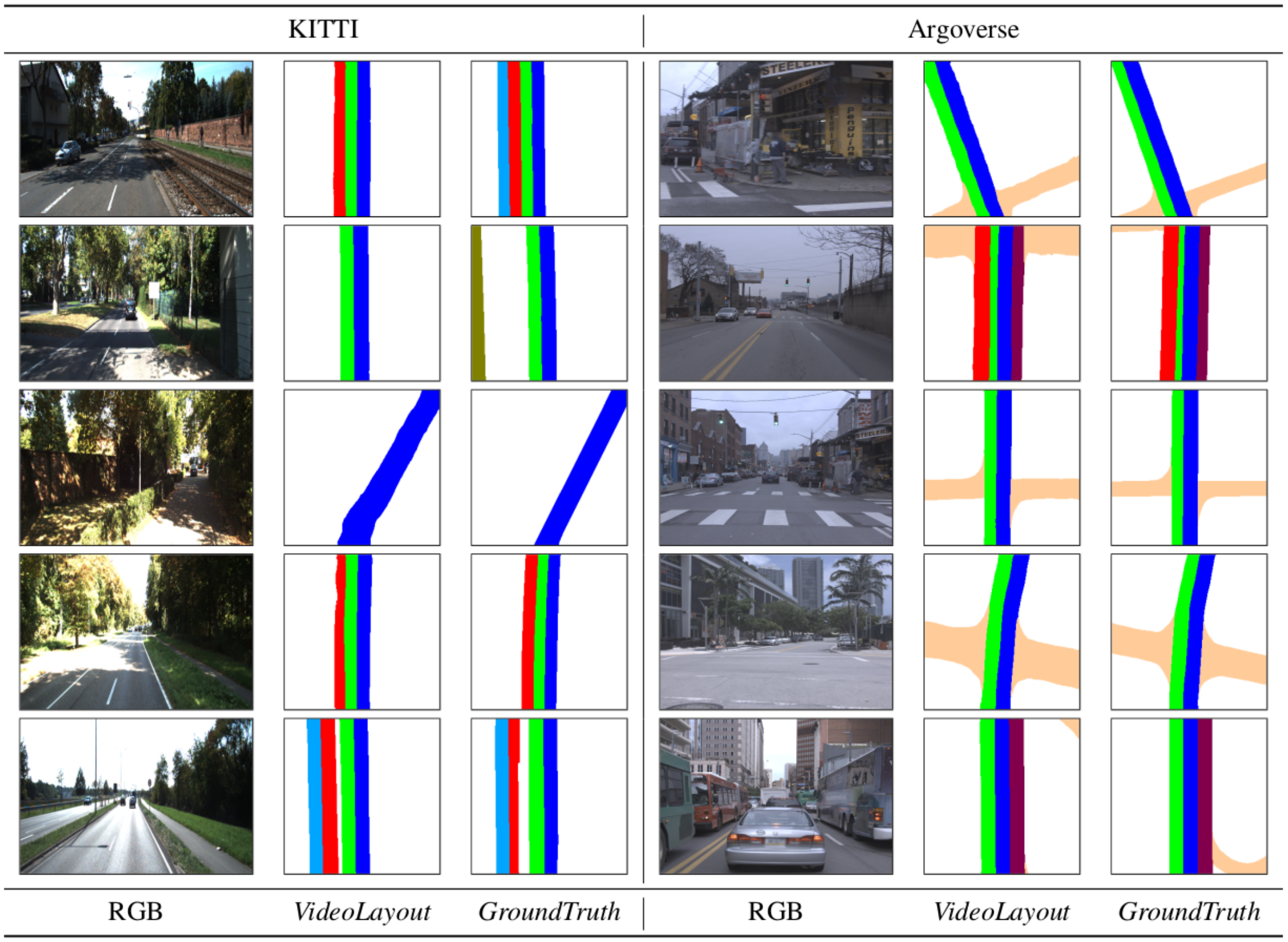

Fig. 4: **Static layout estimation**: *VideoLayout* predicts fine-grained attributes of the road scene including lanes, side roads and ego-lane. Each individual color represents a single lane. Ego-lane is shown in blue and side-roads are shown in light orange. *VideoLayout* produces decent static layout estimates for both AutoLay and Argoverse datasets. Its able to hallucinate occluded regions in the scene very reliably specially for Argoverse dataset.

describe the *AutoLay* evaluation protocol in detail, and evaluate baselines and existing approaches on the benchmark.

A. Datasets

To ensure fair comparison with existing methods, we show our results on the Argoverse [2] and KITTI RAW split used in [3] and [5] respectively. Argoverse [2] provides detailed high-resolution semantic maps in bird's eye view for roads, vehicles and lanes, which makes it suitable for evaluation on all the tasks proposed in Sec. III. We present a breakdown of the *AutoLay* benchmark below.

1) ***KITTI* split**: The *KITTI* split of *AutoLay* comprises 10705 train images and 5626 test images. We report results for models trained using weak (proposed data preparation) and strong (ground-truth) supervision.
2) ***KITTI RAW* split**: For fair evaluation with Schulter *et al.* [5], we use the KITTI RAW split comprising 16269 train images and 2305 test images).
3) ***Argoverse* split**: The Argoverse [2] split contains 6722 train images and 2418 test images.

B. Implementation Details

1) PseudoLidar input: We evaluate the performance of a set of approaches that take PseudoLidar[18] as input for amodal scene layout estimation. The PseudoLidar representation is a processed point cloud obtained by projecting the RGB image pixels to the camera coordinate system using the per-pixel depth values obtained from unsupervised monocular depth estimation methods like [24]. We substitute the remission value by the normalized mean of RGB channel intensities. We then obtain a voxel representation of the point cloud by slicing it vertically along the X-Z plane with a resolution of 0.15625 m and horizontally along the Y-axis into 10 channels starting from 0.4 m above the camera to 2 m below the camera. This PseudoLidar representation is provided as the input to ENet [25] or Unet [26] for amodal scene layout estimation.

2) Lane layout convention: In order to convert lane estimation, which is essentially an instance segmentation/detection task in bird's eye view to an occupancy prediction task, we adhere to the following convention while performing lane layout estimation. While creating the per frame training labels using both weak and strong supervision, we classify the individual lane layouts on the road with the ego-vehicle, (*ego-road*), as either *ego lane* or *other lanes*. The ego lane and the other lanes are provided individual lane ids. The lane ids to the other lanes are given on the basis of their position with respect to the ego-lane.

C. Evaluation Metrics

1) Vehicle Layout: Similar to [3], We evaluate the vehicle layouts on both mean intersection-over-union(mIoU) and mean average-precision(mAP). Since we are evaluating different methods on the task of vehicle occupancy prediction in bird's eye view, we don't go for the traditional evaluation metric($AP_{IoU>0.7}$ or $AP_{IoU>0.5}$) used by 3D object detection methods or object detection methods in bird's eye view.

2) Road Layout: We adopt mean intersection-over-union(mIoU) and mean average-precision(mAP) as our metric for evaluating road layouts on the datasets. To evaluate hallucination capabilities of different methods, we also make use of the occluded mean intersection-over-union (IoU) used in [5], [3]. This involves calculating IoU for only the portions of the road which are occluded.

3) Lane Layout: As mentioned in Sec. III, Lane Layouts are evaluated on two separate tasks: Ego-lane estimation and Overall lane detection.

1) *Ego-Lane Estimation:* Ego-Lane estimation task is evaluated on the mean intersection-over-union (mIoU) and mean average-precision (mAP) metric.
2) *Overall Lane Detection:* For this task, we use $AP_{IoU>0.7}$, a popular metric used for evaluating object detection and instance segmentation methods. A predicted lane is counted as a detection if its intersection-over-union(IoU) with one of the ground truth lanes is greater than 0.7(70%). $AP_{IoU>0.7}$ provides information about the average-precision of predicted detections. We also report $Recall_{IoU>0.7}$, which gives us an idea about the prediction capability of the method.

D. Evaluated Methods

We evaluate the performance of the following approaches.

- *Schulter* et al.: The static scene layout estimation approach proposed in [5].
- *MonoOccupancy*: The static scene layout estimation approach proposed in [4].
- *MonoOccupancy-ext*: We extend MonoOccupancy [4] to predict vehicle occupancies.
- *PseudoLidar-ENet*: A ENet [25] architecture with PseudoLidar input for amodal scene layout estimation.
- *PseudoLidar-UNet*: A UNet [26] architecture with PseudoLidar input for amodal scene layout estimation.
- *MonoLayout*: Amodal scene layout estimation architecture proposed in [3]
- *VideoLayout*: The full *VideoLayout* architecture trainned with temporal-consistency loss and the adversarial refinement network.

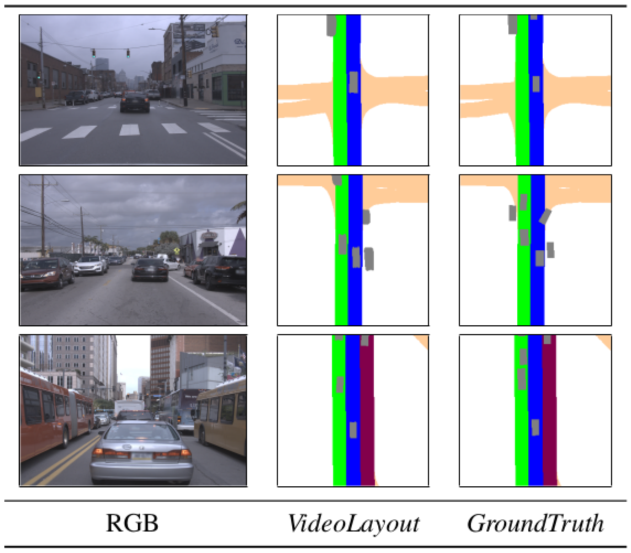

Fig. 5: **Dynamic Layout Estimation:**. *VideoLayout* provides crisp and accurate vehicle occupancies. The grey boxes indicate vehicle occupancies. Observe the ability of *VideoLayout* to precisely localize vehicles that are distant and partially occluded.

E. Road Layout Estimation

We evaluate PseudoLidar based (PseudoLidar-UNet, PseudoLidar-ENet) and RGB image based architectures (MonoOccupancy [4], *MonoLayout* and *VideoLayout*) for the task of road layout estimation using the ground truth labels. Table I provides a detailed benchmark of these methods on the *AutoLay* and Argoverse[2] datasets. We also perform state-of-the-art comparision(Table II) with previously proposed methods in literature on KITTI RAW split proposed in [5] on the task of road and lane layout estimation. For this comparision, we use the weak data generation method mentioned in Sec V-D.

We observe that *MonoLayout* and *VideoLayout* perform better than PseudoLidar-based methods for road layout estimation. This can be attributed to the sparsity of the PseudoLidar points as the distance from camera increases. Also, PseudoLidar based inputs don't allow the use of deep feature encoders like ResNet[27] needed to extract the rich visual features necessary for hallucinating amodal scene layouts. From Table I it can be seen that MonoOccupancy-ext performs better than the PseudoLidar counterparts. This further exemplifies the narrative that with better supervision, an RGB image-based architecture can reason for occluded regions better than pseudo-lidar based approaches.

Within the RGB image based approaches, we observe that *VideoLayout* outperforms *MonoLayout*, MonoOccupancy [4] by considerable margins for all the evaluation metrics. The same can be also be observed in the results on KITTI RAW (*cf.* Table II).

F. Vehicle Layout Estimation

From Table I, we can see that PseudoLidar-based methods have inferior performance in comparision to monocular

Dataset	Method	Vehicle Layout		Static Layout Estimation					
				Road		Lane			
						Ego-Lane		Overall	
		mIoU	mAP	mIoU	mAP	mIoU	mAP	$AP_{iou>0.7}$	$Recall_{iou>0.7}$
KITTI	PseudoLidar-UNet	20.10	52.47	57.62	70.90	51.98	68.94	8.53	19.09
	PseudoLidar-ENet	15.26	40.05	55.70	67.62	54.42	68.15	13.01	24.50
	MonoOccupancy-ext	27.03	42.92	63.10	77.56	58.39	69.16	26.11	35.36
	MonoLayout[3]	30.88	53.63	63.83	77.53	58.27	72.08	16.09	34.91
	VideoLayout	**34.60**	**52.87**	**68.73**	**84.89**	**62.68**	**76.80**	**38.56**	**44.33**
Argoverse	MonoOccupancy-ext	16.22	38.66	72.41	79.62	75.26	85.51	39.76	51.65
	MonoLayout[3]	28.31	46.07	73.25	84.56	71.73	82.00	26.18	42.61
	VideoLayout	**32.77**	**50.48**	**76.42**	**88.01**	**77.62**	**87.77**	**46.03**	**54.86**

TABLE I: **Quantitative results**: We benchmark *VideoLayout*, *MonoLayout* [3], MonoOccupancy-ext and PseudoLidar based approaches on KITTI and Argoverse datasets.

Method	Road	Sidewalk	Road + Sidewalk	Lane
	mIOU	mIOU	occ mIOU	$AP_{iou>0.7}$
MonoOccupancy[4]	56.16	18.18	28.24	26.64
Schulter *et al.* [5]	68.89	30.35	61.06	-
MonoLayout[3]	73.86	32.86	67.42	16.20
VideoLayout	**75.92**	**37.74**	**71.01**	**36.64**

TABLE II: State-of-the-art comparision for static layout(Road, Sidewalk and Lane) with weak supervision(*cf.* Sec. V-D) on KITTI RAW split used in [5]

Method	Ego-Lane		OverAll	
	mIOU	mAP	$AP_{iou>0.7}$	$R_{iou>0.7}$
VideoLayout (no samp., no reg.)	59.81	73.13	20.95	36.55
VideoLayout (no reg.)	61.15	75.72	35.89	41.92
VideoLayout (full)	62.68	76.80	38.56	44.33

TABLE III: Analyzing effect of various subparts of the *VideoLayout* architecture on Lane Layout estimation performance. The sampling layer and adversarial regularization improve performance.

Method	$AP_{iou>0.5}$	$R_{iou>0.5}$	$AP_{iou>0.7}$	$R_{iou>0.7}$
PseudoLidar-UNet	19.32	43.24	8.53	19.09
PseudoLidar-ENet	25.10	47.39	13.01	24.50
MonoLayout[3]	25.38	55.06	16.09	34.91
VideoLayout	57.02	65.49	38.56	44.33

TABLE IV: Analyzing Lane layout scores at IoU thresholds of 0.5 and 0.7. AP denotes average precision. R denotes Recall. *VideoLayout* consistently performs better across IoU thresholds.

methods[3], [4] on vehicle layout estimation. This can again be attributed to the fact that monocular depths are inprecise and sparse at large distances, which makes the PseudoLidar input unreliable for this task.

Similar to road layout estimation, *VideoLayout* performs better than *MonoLayout* due to the temporal consistency loss and the adversarial regularization via the refinement network (*cf.* Fig. 5). Also, MonoOccupany [4] performs poorly on vehicle layout estimation, presumably due to *blurriness* in the variational autoencoder used therein leading to low mAP scores.

G. Lane Layout Estimation

We split the lane layout estimation into two tasks: *ego-lane estimation* and *overall lane detection*. This exclusive treatment of ego-lane estimation is due to its prominence in semi and fully autonomous driving. Unlike objects, which can be “detected" as bounding boxes, lanes are objects whose extents span potentially the entire width/height of the bird’s eye view image, and this calls for an appropriate evaluation protocol. We thus treat lane estimation akin to a “segmentation" problem, and compare performance in terms of average precision and recall at varying IoU thresholds (0.5, 0.7). From Table I, we see that that MonoOccupancy [4] performs significantly better than PseudoLidar methods on ego-lane estimation.

VideoLayout significantly outperforms other methods, owing to its ability to encode temporal information and the use of a separate refinement network. The refinement network corrects blobby lane estimates that do not match the true data distribution, resulting in performance boosts at higher IoU thresholds. A few qualitative results are presented in Fig. 4.

H. Ablation Studies

We conduct ablation studies to analyze the performance of various components in the pipeline. These lead to a number of interesting observations that we enlist below.

1) Sensitivity of lane estimates to IoU thresholds: In Table IV, we analyze the sensitivity of lane estimates to IoU thresholds. At lower thresholds ($AP_{IoU>0.5}$ and $R_{IoU>0.5}$), PseudoLidar-based techniques have similar performance to that of MonoLayout [3]. But as the threshold increases ($AP_{IoU>0.7}$ and $R_{IoU>0.7}$), PseudoLidar methods exhibit a dramatic performance drop, particularly in terms of recall. We attribute this to the fact that PseudoLidar relies on monocular depth estimates which are sparse and inaccurate for vehicles farther from the camera. On the other hand, *VideoLayout* performs consistently across both IoU thresholds. Particularly at $AP_{IoU>0.5}$, we observe a significant performance increase compared to *MonoLayout*.

2) Effect of Sampling and Adversarial Refinement: Table III shows the performance of different variants of *VideoLayout* for lane layout estimation. *VideoLayout* (no samp., no reg.) refers to a variant without a sampling layer in the static decoder and without adversarial regularization. *VideoLayout* (no reg.) uses sampling layer in the static decoder. *VideoLayout* (full) uses both sampling layer and an adversarial regularizer. It can be seen that each component meaningfully contributes to performance boosts. The sampling layer enhances the capability of the model to capture

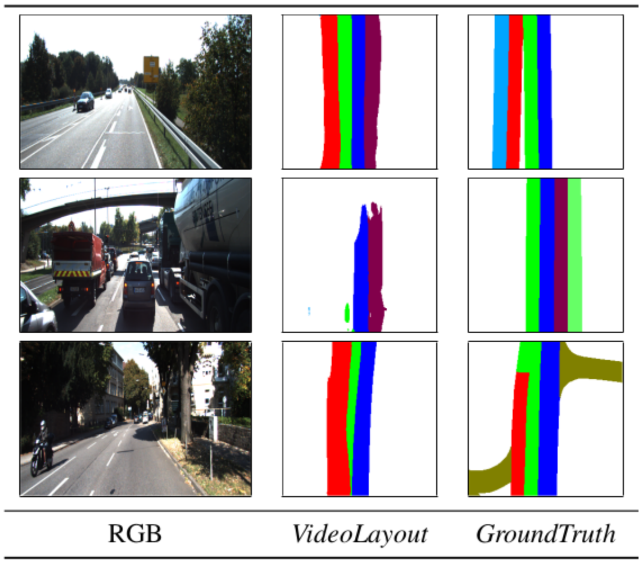

Fig. 6: **Failure Cases** of *VideoLayout*. (*Top row*) Failure to predict ego-lane. (*Middle row*) Failure in heavy traffic scenarios. (*Bottom row*) Failure in predicting lane width.

the distribution of *plausible* layouts, improving precision ($AP_{IoU>0.7}$). The adversarial regularizer improves the sharpness of the predicted samples, improving the recall $R_{IoU>0.7}$.

I. *Failure cases*

Despite its impressive performance, *VideoLayout* fails in multiple scenarios. A few representative failure cases are shown in Fig. 6. Often, such failures are due to high-dynamic range conditions, where shadows are mistaken for lanes, or in heavy traffic scenarios.

J. *Runtime*

On an NVIDIA RTX 1080Ti, the inference rate of *VideoLayout* is slightly over 40 frames per second, which is suitable for real-time performance.

VII. Conclusion

Amodal layout estimation is an emerging perception task for autonomous driving. Solving this task necessitates an understanding of not just entities present in an image, but also a reasoning of occluded portions. Benchmarks have driven progress in allied perception tasks such as object detection, segmentation, tracking, depth estimation, and more. In that spirit, we introduce a new benchmark (*AutoLay*) for amodal layout estimation. We implement several baselines and prior art, and make all our code and data available. We further propose *VideoLayout*, a simple yet effective technique, and demonstrate that reasoning about image sequences leads to coherent and robust layout estimates. We hope the dataset and benchmark serve as a breeding ground for a new class of approaches that will take us a step closer to understanding the 3D world from a sequence of its projections.

References

[1] A. Geiger, P. Lenz, and R. Urtasun, “Are we ready for autonomous driving? the kitti vision benchmark suite,” in *CVPR*, 2012.
[2] M.-F. Chang, J. Lambert, *et al.*, “Argoverse: 3d tracking and forecasting with rich maps,” in *CVPR*, 2019.
[3] K. Mani, S. Daga, *et al.*, “Monolayout: Amodal layout estimation from a single image,” *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2020.
[4] C. Lu, M. J. G. van de Molengraft, and G. Dubbelman, “Monocular semantic occupancy grid mapping with convolutional variational encoder-decoder networks,” *IEEE Robotics and Automation Letters*, 2019.
[5] S. Schulter, M. Zhai, N. Jacobs, and M. Chandraker, “Learning to look around objects for top-view representations of outdoor scenes,” in *ECCV*, 2018.
[6] Y. You, Y. Wang, *et al.*, “Pseudo-lidar++: Accurate depth for 3d object detection in autonomous driving,” *arXiv preprint*, 2019.
[7] O. Özyeşil, V. Voroninski, R. Basri, and A. Singer, “A survey of structure from motion*.” *Acta Numerica*, vol. 26, pp. 305–364, 2017.
[8] A. Kar, S. Tulsiani, J. Carreira, and J. Malik, “Amodal completion and size constancy in natural scenes,” in *International Conference on Computer Vision (ICCV)*, 2015.
[9] S. Tulsiani, S. Gupta, *et al.*, “Factoring shape, pose, and layout from the 2d image of a 3d scene,” in *Computer Vision and Pattern Regognition (CVPR)*, 2018.
[10] Z. Wang, B. Liu, S. Schulter, and M. Chandraker, “A parametric top-view representation of complex road scenes,” in *CVPR*, 2019.
[11] C. Lu and G. Dubbelman, “Hallucinating beyond observation: Learning to complete with partial observation and unpaired prior knowledge,” *arXiv preprint*, 2019.
[12] N. Garnett, R. Cohen, *et al.*, “3d-lanenet: end-to-end 3d multiple lane detection,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 2921–2930.
[13] J. Ku, M. Mozifian, *et al.*, “Joint 3d proposal generation and object detection from view aggregation,” in *IROS*, 2018.
[14] X. Chen, H. Ma, *et al.*, “Multi-view 3d object detection network for autonomous driving,” in *CVPR*, 2017.
[15] M. Liang, B. Yang, S. Wang, and R. Urtasun, “Deep continuous fusion for multi-sensor 3d object detection,” in *ECCV*, 2018.
[16] B. Yang, W. Luo, and R. Urtasun, “Pixor: Real-time 3d object detection from point clouds,” in *CVPR*, 2018.
[17] T. Roddick, A. Kendall, and R. Cipolla, “Orthographic feature transform for monocular 3d object detection,” *arXiv preprint*, 2018.
[18] Y. Wang, W.-L. Chao, *et al.*, “Pseudo-lidar from visual depth estimation: Bridging the gap in 3d object detection for autonomous driving,” in *CVPR*, 2019.
[19] W. Maddern, G. Pascoe, C. Linegar, and P. Newman, “1 Year, 1000km: The Oxford RobotCar Dataset,” *The International Journal of Robotics Research (IJRR)*, vol. 36, no. 1, pp. 3–15, 2017.
[20] H. Caesar, V. Bankiti, *et al.*, “nuscenes: A multimodal dataset for autonomous driving,” *arXiv preprint arXiv:1903.11027*, 2019.
[21] X. Huang, P. Wang, *et al.*, “The apolloscape open dataset for autonomous driving and its application,” *arXiv preprint arXiv:1803.06184*, 2018.
[22] OpenStreetMap contributors, “Planet dump retrieved from https://planet.osm.org ,” https://www.openstreetmap.org, 2017.
[23] J. Behley, M. Garbade, *et al.*, “Semantickitti: A dataset for semantic scene understanding of lidar sequences,” in *ICCV*, 2019.
[24] C. Godard, O. Mac Aodha, M. Firman, and G. Brostow, “Digging into self-supervised monocular depth estimation,” *arXiv preprint*, 2018.
[25] A. Paszke, A. Chaurasia, S. Kim, and E. Culurciello, “Enet: A deep neural network architecture for real-time semantic segmentation,” *arXiv preprint arXiv:1606.02147*, 2016.
[26] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” in *International Conference on Medical image computing and computer-assisted intervention*, 2015.
[27] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.