A Fast Unified System for 3D Object Detection and Tracking

Thomas Heitzinger and Martin Kampel
Computer Vision Lab, TU Wien
Vienna, Austria
{thomas.heitzinger, martin.kampel}@tuwien.ac.at

Abstract

We present FUS3D, a fast and lightweight system for real-time 3D object detection and tracking on edge devices. Our approach seamlessly integrates stages for 3D object detection and multi-object-tracking into a single, end-to-end trainable model. FUS3D is specially tuned for indoor 3D human behavior analysis, with target applications in Ambient Assisted Living (AAL) or surveillance. The system is optimized for inference on the edge, thus enabling sensor-near processing of potentially sensitive data. In addition, our system relies exclusively on the less privacy-intrusive 3D depth imaging modality, thus further highlighting the potential of our method for application in sensitive areas. FUS3D achieves best results when utilized in a joint detection and tracking configuration. Nevertheless, the proposed detection stage can function as a fast standalone object detection model if required. We have evaluated FUS3D extensively on the MIPT dataset and demonstrated its superior performance over comparable existing state-of-the-art methods in terms of 3D object detection, multi-object tracking, and, most importantly, runtime.

1. Introduction

We present a **F**ast **U**nified **S**ystem for **3D** Object Detection and Tracking, abbreviated as **FUS3D** Object Detection and Tracking. The proposed approach performs both 3D object detection and multi-object-tracking in a joint framework that solely relies on 3D depth data. Our work is the first to demonstrate that such a system can be sufficiently optimized to achieved real time inference speeds on edge devices such as the Nvidia Jetson Nano singleboard computer. Despite the tight integration our system is flexible in its use. If required, our object detection stage can be used as a standalone model, thus allowing for even faster runtime at the cost of a small drop in detection accuracy.

The FUS3D system is best suited to small scale indoor environments and a static camera setting. Typical examples of such settings are human monitoring systems in the domains of ambient assisted living (AAL) and surveillance, which can be particularly well served with a combination of 3D depth sensors and edge-based model inference due to the intrinsic characteristics of the respective technologies. Depth sensors can operate continuously and largely independently of lighting condition, as they do not require an external source of illumination. The absence of color intensity information also renders the invasion of people's privacy a less prominent issue, and is further reduced when potentially sensitive data can be analyzed sensor-near. FUS3D is decidedly application-oriented, and we believe it can bring immediate benefits to human-centered support and monitoring systems in AAL and related fields.

Our work is evaluated on the MIPT [12] dataset which consists of sequences of depth data focused on human indoor activity. We outperform comparable previous state-of-the-art methods in terms of object detection performance, tracking metrics, and, most importantly, runtime.

The main contributions of our paper are:

- FUS3D is more than 10 times faster than state-of-the-art 3D object detection methods while also incorporating a transformer-based tracking stage. FUS3D is the first method to demonstrate a unification of 3D object detection and multi-object-tracking on severely resource constrained hardware. Model code is publicly available. [1]
- We present several approaches to improve 3D detection performance with little to no computational overhead, including an auxiliary orientation estimation loss, use of global context for tracking, and a novel dense target assignment (DTA) scheme.
- A seamlessly integrated transformer-based tracking stage that outperforms existing trackers and allows for end-to-end training along with a simplified approach to track association and acquisition.

Section 2 gives an overview over related work in the areas of joint detection-and-tracking and transformer-based

[1] https://github.com/theitzin/FUS3D

systems in vision tasks. We continue in Section 3 with a detailed description of the proposed pipeline. The method is evaluated in Section 4 by giving ablation studies to validate design choices and comparisons with the current state-of-the art. Finally we concluded in Section 5 with a discussion of limitations and reflection on the presented work.

2. Related Work

For our review of related work we categorize existing approaches into four groups. We begin with notable work operating under the tracking-by-detection paradigm and follow up with the category of joint-detection-and-tracking. Then we focus on transformer architectures integrated in vision tasks for purposes other than tracking, and last is existing work that integrates transformers for the explicit purpose of object tracking over time. The latter includes both methods intended for use on 2D as well as 3D data.

Tracking-by-detection Tracking methods operating under the tracking-by-detection paradigm feature separate tracking and detection mechanisms, with the latter yielding a set of detections at each time step. A given tracking algorithm then creates associations between these sets to form object trajectories over time. A prominent approach in this category is SORT [5], which uses Kalman filters [40] to model bounding box attributes. New detections are optimally associated with existing tracks using the Hungarian algorithm [20] and the Mahalanobis distance as a matching cost function. In its extended DeepSORT [41] variant it introduces deep learning feature similarity as an additional factor in the association cost calculation. This principle of association-by-appearance is found in different variations [45, 43, 17] throughout tracking literature. For applications featuring frequent occlusions and crowded scenes however, appearance-based methods may struggle to produce sufficiently differentiating feature vectors. Other *motion* based approaches try to palliate the object-to-track association problem through modeling of object trajectories. The principles behind these methods range from constant velocity assumptions [1, 5] to the social force model [28, 44, 22] and optical flow estimation [16].

Joint-detection-and-tracking In contrast to previous methods, the joint-detection-and-tracking approach aims to combine both the detection and the tracking task into a single system. Methods in this category operate under the assumption that the classical tracking-by-detection paradigm can and should be extended by a reverse detection-by-tracking principle. They use this assumption to perform cross-frame track regression [11] or utilize track-assisted detection proposals [2, 52, 47]. Other methods achieve combined detection and tracking through simultaneous prediction of objects and their appearance embedding from a common backbone [39, 46].

Use of transformers in vision tasks With the emergence of the transformer architecture [37] the deep learning toolkit was extended by a powerful set-to-set translation mechanism. Aware of the potential of such a general tool, the authors of DETR [6] were among the first to successfully apply transformers in the computer vision domain. They propose an end-to-end 2D object detection approach that builds on top of a CNN backbone, allowing them to eliminate the need for handcrafted spatial anchors and non-maximum-suppression. Ubiquitous utilizations of the attention mechanism to the tasks of image classification [9], object detection [6, 53] and segmentation [49, 23] among many others have since demonstrated its versatility and applicability throughout all areas of computer vision.

Transformer-based tracking In line with the naming convention of other tracker categories, transformer based trackers have been grouped under the term *tracking-by-attention* – a term coined by the authors of the TrackFormer [27]. The intended use-case of the TrackFormer architecture is simultaneous 2D object detection and tracking based on RGB data. It uses the attention mechanism to jointly reason about the initialization, termination, and spatio-temporal propagation of tracks, and in this regard shares similarities with our proposed approach. Both the TrackFormer and TransTrack [32] develop the notion of *object queries* and *track queries* which are also used in our approach. Object queries are learned tokens that are used as queries for the detection of objects that newly appear in the current frame, whereas track queries are features passed on from the previous time steps. Each track query represents an object that has already existed at the previous time step. In contrast to both TrackFormer and TransTrack our approach leverages the light-weight Perceiver [14] architecture, thus requiring no encoder and only a single decoder stage for joint arbitration of both object and track queries. Efforts towards the unification of 2D single-object-tracking (SOT) and multi-object-tracking (MOT) on top of a joint detection/tracking architecture have resulted in the UniTrack [38] and UTT [26] architectures. This branch of development, however, is largely unrelated to the subject of this paper.

Finally, some of the mentioned concepts have been applied to tasks in the 3D data domain. LTTR [7], PTTR [51] and PTT-Net [15] all demonstrates an approach for 3D single-object-tracking in large scale outdoor environments captured as LiDAR pointclouds. Both LTTR and PTTR are not concerned with runtime optimization. They utilize a Siamese-like [4] tracking pipeline featuring either a 3D sparse CNN or PointNet [30]-based backbone followed by a tokenization step and feature fusion using an attention mechanism. While PTT-Net is optimized for runtime, it decidedly differs from our approach in terms of: 1. target applications, 2. data formats (LiDAR vs depth maps), 3. real

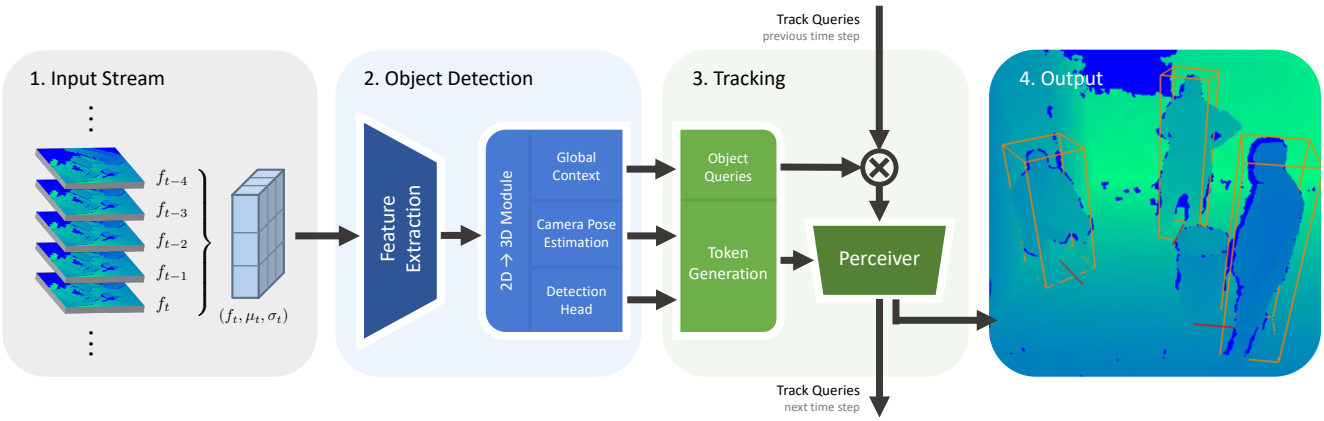

Figure 1. The basic structure of the FUS3D detection and tracking pipeline: 1. Input frames, preprocessing and background model, 2. 2D CNN backbone, restructuring of features to a 3D representation and outputs for preliminary single-frame-based 3D bounding boxes, estimation of the camera pose and a global context summary, 3. Tokenization of features, assembly of new object queries and existing track queries for processing by a Perceiver model and 4. Projection and filtering of output bounding boxes.

time inference on desktop hardware vs edge devices and 4. single-object-tracking vs multi-object-tracking.

To the best of our knowledge, no previous work has extended the transformer-based tracking paradigm to 3D object detection and multi-object-tracking in depth maps with the intention of real-time inference on edge devices.

3. Methodology

We now present the individual components of our tracking pipeline. Next to initial preprocessing we present a two-stage learning-based approach dedicated to 3D object detection and multi-object-tracking respectively. A simplified overview of the key pipeline components is given in Figure 1. More in-depth illustrations and detailed descriptions are given in the respective sections.

The input data representation expected by our system are depth maps. In order to maximize inference speed, both preprocessing and early feature extraction are performed in 2D image space. If required, our method permits the direct incorporation of other image-based modalities such as RGB or thermal images by simply stacking the individual images at the input.

Our approach for object detection is similar to the baseline system presented by the authors of the MIPT dataset [12], which was designed for similar use-cases. For easy reference we label their system as $\mathcal{S}_{\text{MIPT}}$. Both $\mathcal{S}_{\text{MIPT}}$ and FUS3D use a background model as proposed by Wren *et al.* [42] to enable extraction of basic temporal information at the pixel level. It exploits the targeted static-sensor setting and gives a significant boost to the systems' performance at negligible computational cost.

3.1. Object Detection

Our approach to 3D object detection can be categorized as a single stage detector. Predictions are made with respect to a predefined grid of possible object centers, where the grid is obtained from subdivision of the sensors field of view in image-space into a set of *cells*. While the grid is uniform when viewed in image space, cells appear irregularly shaped in world space, as illustrated in Figure 2 (left). The main goal of early feature extraction in the object detection stage (detailed illustration see Figure 3) is the extraction of one-dimensional feature vectors $\boldsymbol{c} \in \mathbb{R}^c$ for each cell. Given a cell grid of height h, width w and depth d, the desired shape of the full feature tensor is (h, w, d, c). To ensure fast inference, the first step in feature extraction is a 2D CNN backbone. This backbone is configured to produce an output tensor of shape (h, w, dc), which is further passed to a 2D-to-3D conversion module to achieve the desired 4D shape. We achieve this conversion step with a reshape operation, followed by a 3D convolutional residual block. By associating cell neighborhoods in all three dimensions, the addition of the 3D residual block enables the model to better learn the ordering of cells in the depth dimension. Cell feature vectors are subsequently projected down to *detections* $\boldsymbol{d} \in \mathbb{R}^{10+n_{cls}}$ that express potential bounding boxes for a detection problem with n_{cls} classes. The bounding box representation consists of box-center location $loc \in \mathbb{R}^3$, box dimensions $dim \in \mathbb{R}^3$, facing direction as a normalized orientation vector in the ground plane $dir \in \mathbb{R}^2$, a confidence score $conf \in \mathbb{R}$, an indication of the centerness of a cell within the bounding box (only relevant for dense target assignment – see below) $center \in \mathbb{R}$ and a class probability distribution $cls \in \mathbb{R}^{n_{cls}}$. The manner of cell assignment to ground truth objects is critical and leads us to the development of a custom target assignment scheme.

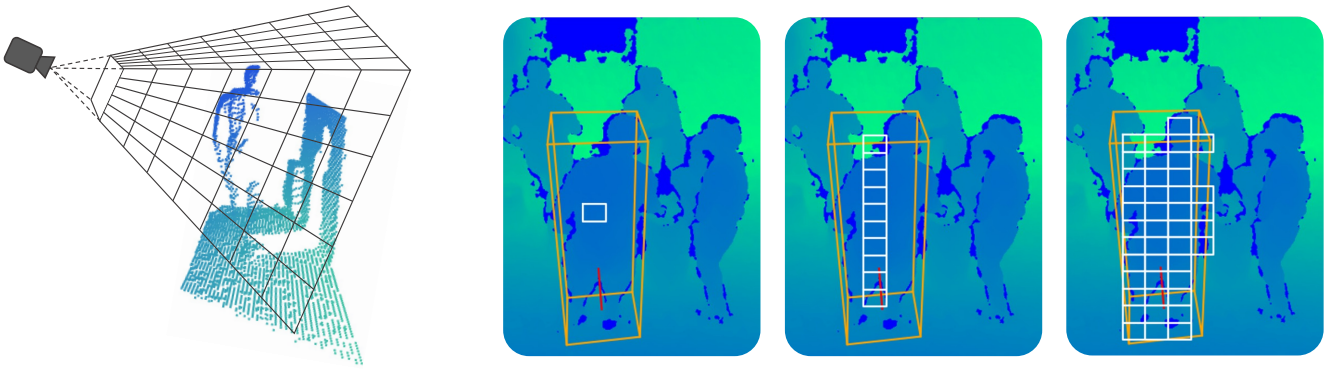

Figure 2. Target assignment strategy overview. Cells viewed in world space have irregular shape (grid shown in black on the left) whereas their size is uniform in image space (shown in white on the right). Cells along the depth axis occlude each other. Assignment strategies from right to left: dense target assignment (DTA), broadcasted target assignment (BTA) and center-based assignment.

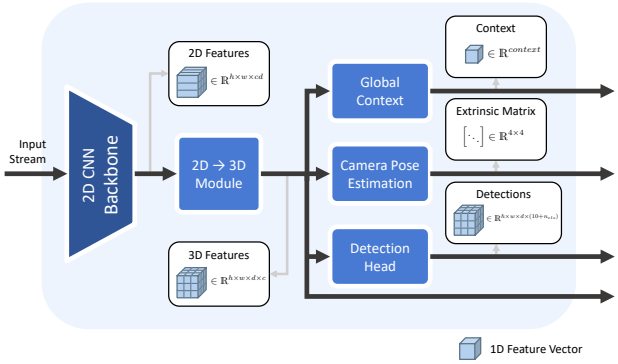

Figure 3. Detailed overview of individual components in the 3D object detection stage and tensor shapes. Arrows to the right show objects forwarded to the tracking stage.

Target assignment In the naive approach, only the cell closest to the center of a ground truth box would have an assigned prediction target (Figure 2 second from the left). Assigned, or *positive*, cells are optimized to match the ground truth bounding box parameters loc, dim, dir and cls, and have the confidence target $conf = 1$, while other, *negative*, cells are only optimized to predict a confidence value of $conf = 0$. In the case of the location property we do not directly predict the ground truth location but instead the offset of the cell center to the ground truth location. The training loss is computed as a weighted sum over mean squared error for loc and dim, negative cosine similarity for dir, cross-entropy loss for cls and binary cross-entropy for $conf$ (see supplementary material for a more detailed description). In practice it becomes apparent that the severe class imbalance between positive and negative cells at a typical ratio of $10^4 : 1$ or higher is difficult to optimize. A possible workaround for this problem is to assign multiple cells per ground truth target, as proposed in the 2D case by the FCOS detector [36] and in simplified form by $\mathcal{S}_{\text{MIPT}}$ in the form of *broadcasted target assignment (BTA)* (Figure 2 second from right). With FUS3D we propose what we consider to be a logical extension of this concept and regard the set of cells contained *anywhere* within a ground truth bounding box to be positive (Figure 2 rightmost) and refer to this approach as *dense target assignment (DTA)*. To be more precise, for each ground truth bounding box $\mathbf{b}$ we define the linear transformation $\tau_{\mathbf{b}} : \mathbb{R}^3 \rightarrow \mathbb{R}^3$ into a local *box coordinate system* which is aligned with the orientation of the box such that $\tau_{\mathbf{b}}(\mathbf{b}_{loc}) = 0$ and $\left\|\tau_{\mathbf{b}}(\cdot)\right\|_\infty = 1$ for any point on the boundary $\partial\mathbf{b}$ of the respective box. Given the set of cell centers $\mathcal{C}$ and a ground truth bounding box $\mathbf{b}$ the expression

$$\mathcal{C}_{\mathbf{b}} := \left\{x \in \mathcal{C} : \left\|\tau_{\mathbf{b}}(x - \mathbf{b}_{loc})\right\|_\infty < 1\right\}, \qquad (1)$$

defines the set of cells which are assigned $\mathbf{b}$ as an optimization target.

Since objects in 3D objects cannot overlap significantly, the problem of overlapping bounding boxes as described in FCOS is highly unlikely and its resolution a non-issue. Using DTA, the prediction of cell confidence is thus not unlike a form of coarse 3D segmentation. If FUS3D is intended to be used merely as an object detection network, then no further network components are required. In this case, sensor intrinsics and its pose description are used to transform predicted image-space locations to world-space coordinates. Further postprocessing using non-maximum-suppression yields a set of bounding box predictions.

Centerness Following FCOS [36], we introduce an additional *centerness* bounding box attribute that is zero at the bounding box center and monotonically increases to one at its boundary. Compelling the network to learn not only whether cells are within a bounding box, but also how close they are to the center, leads to a tangible improvement in detection performance at negligible additional processing cost. The centerness definition originally given by the au-

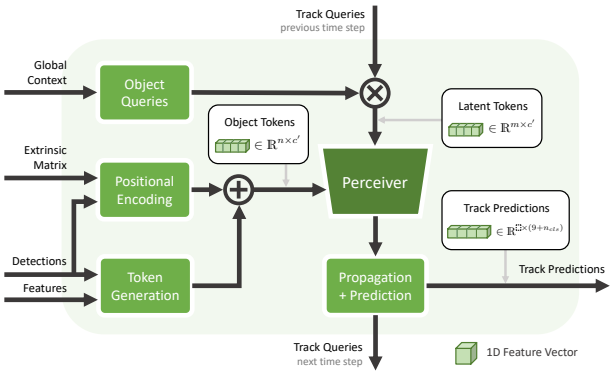

Figure 4. Detailed overview of individual components in the tracking stage and tensor shapes. Objects originating from the detection stage are visualized as arrows from the left, while tracks from previous time steps are introduced from the top. Updated track queries are output at the bottom and the final track predictions of FUS3D are output to the right.

thors of FCOS was intended for 2D object detection and is therefore extended to the 3D case.

Orientation estimation The proposed FUS3D system performs object detection in image space. In order to obtain predictions in world space a transformation and knowledge of the sensors pose is required. While, for the purpose of all evaluations, we assume that the sensor pose is known, we find during experimentation that the introduction of a auxiliary camera pose estimation task along with an accompanying network branch is beneficial to detection performance. The branch takes the 3D feature tensor as input (see Figure 3), applies global average pooling to all spatial dimensions and uses the resulting one-dimensional (image-space-based) *global context* $\boldsymbol{g} \in \mathbb{R}^c$ to predict both an orientation estimate $orientation \in \mathbb{R}^3$ corresponding to the world-space "up" direction as well as a scalar $offset$ indicating the height difference between sensor and ground level.

Global context The global context is identical to the intermediate one-dimensional feature vector $\boldsymbol{g}$ obtained for orientation estimation. It is passed to the tracking stage as supplementary information.

3.2. Tracking

The object detection stage uses cells as units for processing. Each cell represents a region of 3D space and is described by a one-dimensional feature vector. Thus, each cell has both semantic meaning and a location associated with it, and it is natural to transfer them directly to the tracking stage where, after some preprocessing, they take on the role of *object tokens*.

Token generation As an initial step in object token generation, we apply a confidence threshold that uses the confidence value predicted in the object detection stage. This filters out cells that are assumed to correspond to empty space, thereby greatly reducing the number of cells that are included in further processing. The earlier proposed dense target assignment scheme has the opposite effect. It increases the number of positive cells and ensures that a comprehensive description of the processed scene is passed on to the tracking phase. Using the center-only target assignment scheme, bounding boxes would be largely expressed by individual cells, reducing the role of the tracking stage to mere object propagation. After extensive experimentation we find that for the generation of object tokens, best results are achieved by combining both cell features $\boldsymbol{c}$ and detections $\boldsymbol{d}$. We concatenate them at the cell level and apply a multilayer perceptron (MLP) [19] to obtain object tokens $\boldsymbol{t} \in \mathbb{R}^{c'}$, where typically $c' > c$ (see block *Token Generation* in Figure 4). The object tokens are further augmented using fixed sine/cosine based encodings as originally proposed by Vaswani *et al*. [37]. Coordinates associated with cells are not scalar but three-dimensional, so we concatenate multiply positional encodings computed for the individual coordinate entries. Both cell centers and locations predicted in the object detection stage require knowledge of sensor extrinsics to compute and are viable choices to use for the positional encodings. We find that predicted locations perform slightly better. The set of object tokens represents a pool of knowledge about an individual frame that we want to distill into a much smaller set of *track tokens*. We use the Perceiver [14] architecture to realize this distillation step. While TransTrack uses a two-decoder setup, we simplify this approach to a single Perceiver module that jointly performs both object detection and object propagation.

Object and track queries As proposed by both TransTrack [32] and the TrackFormer [27] we use the notion of object queries and track queries, which represent two variants of latent tokens. Track queries are features passed from previous time steps (input *Track Queries* in Figure 4). Each such token represents an object that has already existed at the previous time step and is passed on to the current time step. Combined with the background model applied during preprocessing, our approach achieves both early and late fusion of features over time, thereby allowing it to form temporal associations quickly and effectively. Object queries, on the other hand, are used for detection of objects that newly appear in the current frame. They are learned and context dependent. We obtain them from a token-wise MLP that takes both an object query index and the global scene context $\boldsymbol{g}$, passed from the object detection stage, as input (see block *Object Queries* in Figure 4). Our evaluations show that the inclusion of the global context in

Table 1. Comparison against state-of-the-art models in single-class object detection. Best method intended for edge devices is marked in bold (bottom section), overall best is underlined. FPS measured on Nvidia TITAN X [TX] and Jetson Nano [Jet], mAP and mAHS numbers given at @ 0.25 IoU.

Configuration	mAP	mAHS	FPS [TX]	FPS [Jet]
VoteNet [29]	64.4	50.8	6.3	0.8
PointPillars [21]	55.8	45.2	6.0	0.7
H3DNet [48]	65.7	53.2	4.2	0.5
Group-Free 3D [24]	<u>67.3</u>	<u>55.8</u>	3.9	0.3
$\mathcal{S}_{\text{MIPT}}$ [12] (Large)	63.9	54.4	<u>10.3</u>	<u>1.2</u>
$\mathcal{S}_{\text{MIPT}}$ [12]	43.7	37.2	71.5	11.3
Ours (no Tracker)	**44.6**	**42.4**	**72.8**	**11.4**

Table 2. Ablation study of components in the object detection stage. Multi-class setting (pose classification into "Standing", "Sitting", "Lying" included). FPS measured on an Nvidia Jetson Nano, mAP and mAHS number given at @ 0.25 IoU.

Configuration	mAP	mAHS	FPS
No BG-Model	20.91	19.46	11.5
No 2D $\rightarrow$ 3D Block	40.74	34.98	**12.7**
Target Assignment: Center	34.14	30.39	11.4
Target Assignment: BTA	39.13	34.33	11.4
No Centerness	39.87	33.96	11.4
No Orientation Estimation	40.53	34.32	11.4
Ours (no Tracker)	42.40	35.90	11.4
Ours (with Tracker)	**43.20**	**39.24**	6.4

this step yields tangible improvements in the acquisition of new object tracks.

Track propagation and prediction Track predictions are obtained by again applying a token-wise MLP to the set of latent tokens (see output *Track Predictions* in Figure 4). We predict the same set of bounding box parameters and use the same loss function as in the object detection stage, with the exception of the centerness attribute. The association strategy between tracks of previous time steps and current predictions differs between training and inference phases.

During training we match predicted with ground truth tracks in a two-step process: 1. Matching of newly appeared ground truth tracks with object queries and 2. Matching of ground truth tracks that already existed in the previous time step with track queries. To realize the first step, each object query is associated with an anchor box. Anchor boxes are a set of bounding boxes that were obtained through k-Means clustering on the training set and represent the most commonly found objects in the dataset. They are matched to the bounding boxes of new ground truth tracks using the Hungarian Algorithm [20] and a matching cost heuristic based on Euclidean distance and the Distance-IoU loss [50]. Matching with existing tracks is achieved implicitly by enforcing that the track query appears at the same latent token index as on the previous time step. Assigned queries are optimized to predict bounding boxes with confidence one, and zero otherwise.

In the inference case updates are conducted by imposing *add* and *remove* thresholds $\tau_{add}, \tau_{remove} \in [0, 1]$ on the confidence of predicted bounding boxes. Object queries with confidence higher than τ_{add} are promoted to a track query, provided that they do not have large IoU overlap with existing tracks. Track queries with confidence lower than τ_{remove} are dropped. The resulting new set of track queries is passed on to the next time step.

4. Experiments

We perform our evaluations on the MIPT [12] dataset. It consists of $85k$ individual frames split across 20 indoor sequences. The content is focused on human activity and captured by a static depth sensor using structured light technology. Trajectories of recorded individuals are annotated fully in 3D with oriented bounding boxes as well as a per-frame and per-track pose categorization into the classes "Standing", "Sitting" or "Lying". To the best of our knowledge there currently exists no other public dataset featuring similar characteristics in terms of 1. sequential depth data (not LiDAR), 2. 3D trajectory annotation, 3. environment scales on the order of indoor scenes, 4. a static sensor setting and 5. dataset size. The MIPT dataset further provides sensor intrinsics and extrinsics for each sequence. These ground truth parameters are used for all of our evaluations. The CNN backbone employed for feature extraction uses the MobileNetV2 [31] architecture. Despite extensive testing we found it to perform better than newer MobileNetv3 [13], MnasNet [33] or EfficientNets [34, 35] architectures. Our interpretation of this effect is that runtime and accuracy improvements presented by such backbones are largely achieved with the RGB modality in mind.

Object detection comparison against the state-of-the-art We start off with a comparison of our proposed object detection against the state-of-the-art and follow the evaluation protocol given in the MIPT paper. For fair comparison with existing methods, the experiment is performed without use of any temporal information, thus neither the proposed background model nor the tracking stage are used. Performance numbers are given by frame-per-second (FPS) numbers on fixed hardware, the standard *mean average precision (mAP)* [10] metric as well as *mean average heading similarity (mAHS)* [18] which is an extension of mAP weighing down a true positive detection by its heading similarity to the ground truth. Both mAP and mAHS number

Table 3. Ablation study of components in the tracking stage and comparison with the DeepSORT tracker. Using HOTA, CLEARMOT and Mostly-Tracked/Partly-Tracked/Mostly-Lost metrics as well as mAP and mAHS @ 0.25 IoU and FPS measured on an Nvidia Jetson Nano.

Configuration / Method	HOTA	DetA	AssA	MOTA	MOTP	MT	PT	ML	mAP	mAHS	FPS
DeepSORT	32.7%	53.4%	20.2%	**60.4%**	75.3%	5	16	1	42.40	35.90	–
Ours (no global context)	35.3%	**53.6%**	23.4%	54.3%	75.4%	7	15	0	42.12	38.06	6.4
Ours (no anchor boxes)	34.2%	46.5%	24.5%	43.8%	69.4%	6	14	2	38.82	36.11	6.4
Ours (only cell features)	36.0%	53.0%	**25.4%**	53.9%	76.8%	7	15	0	40.31	38.51	**6.6**
Ours (only detection features)	35.8%	52.8%	22.7%	55.0%	75.0%	7	13	2	41.78	38.80	6.5
Ours (no overlap check)	31.1%	40.4%	24.0%	15.0%	78.3%	7	15	0	37.90	33.60	6.5
Ours	**36.5%**	53.5%	25.1%	58.0%	**78.4%**	7	15	0	**43.20**	**39.24**	6.4

are given at an IoU threshold of 0.25. Furthermore metrics are computed for a single "Human" class, no pose classification takes place. Performance numbers given in Table 1 are split between methods intended for use on edge devices (bottom) and general use (top). Our approach can outperform the baseline $\mathcal{S}_{\mathrm{MIPT}}$ model in the bottom category both in terms of detection accuracy and runtime. Especially notable is the small difference between mAP and mAHS with 6.5 percentage point on $\mathcal{S}_{\mathrm{MIPT}}$ compared to 2.4 percentage points for our method, indicating that it is significantly better at estimation of object orientations.

Object detection ablation study Next we validate the design choices of our object detection stage. The evaluation metrics are identical to the comparison against the state-of-the-art with the exception of FPS numbers. They are now given on an Nvidia Jetson Nano singleboard computer, which is our primary target device. In contrast to the previous experiment we now consider the multi-class setting and give performance numbers averaged over the three pose classes "Standing", "Sitting" and "Lying". The base configuration used in the ablation study is the full object detection stage with postprocessing in the form of non-maximum-suppression. We verify the efficacy of each proposed additional component and finally include an additional test configuration including the proposed tracking stage. As shown in Table 2, the removal of most components leads to a 2 to 3 percentage point drop in terms of mAP and mAHS. Outliers are the naive center-based target assignment strategy, giving a drop of 8 points and most significantly-but not unexpectedly, the removal of the background model which roughly halves the mAP score. The removal of components has for the most part no significant impact on model runtime, with the exception of the 2D $\rightarrow$ 3D Block which increases the framerate by 11%, at the cost of a drop in detection performance. Additional inclusion of the FUS3D tracking stage shows an *increase* in both mAP and mAHS of 0.8 and 3.3 percentage points but also reduces the framerate by 44% which may or may not be desirable in applications without a tracking requirement. The more pronounced increase in mAHS compared to mAP suggests that the additional temporal associates allow the transformer stage to better judge object orientations.

Tracking ablation study and baseline comparison We continue with an evaluation of the proposed FUS3D tracking stage. Since no comparable joint detection and tracking systems currently exists, we elect to instead compare the full FUS3D system with a pipeline consisting of the FUS3D detection stage and the established DeepSORT [41] tracker. Additionally we verify the presented design choices with an ablation study on the tracking stage. Next to the previously used mAP and mAHS object detection metrics, we choose a set of tracking metrics in accordance with the MOT20 benchmark [8], including HOTA [25], CLEAR MOT metrics [3]. For completeness we also include FPS numbers in the ablation study. However, this metric is not included for the baseline DeepSORT tracker, since no sensible comparison between classical and GPU-accelerated trackers can be made. As shown in Table 3, our approach shows superior metrics on all metrics except on multi-object-tracking accuracy (MOTA), where our proposed configuration is in second place by a small margin. Our model tends to produce increase localization precision as indicated by the higher MOTP metric and the shifted IoU distribution in Figure 5 top-left. Our tracking stage is capable of suppressing false positive detection proposals produced by the detection stage, however conversely we see a slight increase in false negatives and ultimately lower MOTA where the tracking stage overly aggressively prunes detections it deems incorrect. Higher association accuracy (AssA) of the "only cell features" is a direct consequence of the tracking stages higher reliance on temporal context when no bounding box attributes by the detection stage are available. Overall the ablation study demonstrates the positive impact of each of the components and strategies involved in the FUS3D tracking stage. Across the set of configuration evaluated in the ablation study there is no significant variation in runtime.

Advantages of a transformer-based second stage We compare object detection characteristics of the standalone FUS3D detection stage combined with non-maximum-suppression postprocessing and the full FUS3D pipeline.

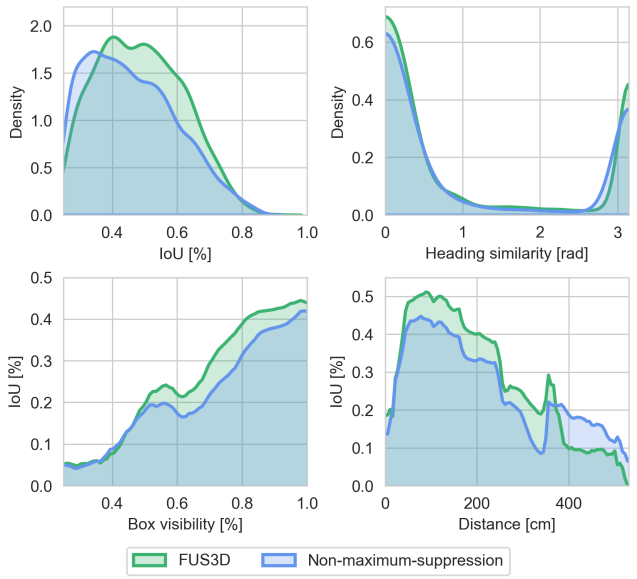

Figure 5. Visualization of object detection characteristics: IoU distribution (top left), deviation from ground truth heading direction (top right), IoU as a function of box visibility (bottom left) and IoU as a function of box distance (bottom right).

For the purpose of this experiment we write NMS to refer to the first configuration.

We begin with a visualization of IoU distributions of all true positive bounding boxes (Figure 5 top-left). Here the FUS3D IoU distribution has noticeably higher mean compared to the NMS IoU distribution. Both approach zero for IoU rates over 80%.

Second, we analyze the distributions of heading similarity deviations between predicted and ground truth bounding boxes. A deviation of zero indicates a perfect heading angle while a deviation of π means that the bounding box is pointed in opposite direction. As can be seen in the upper right quadrant of Figure 5, the distributions are bimodal, showing that both of the tested configurations share a common error mode where the front and back of pictured individuals are confused. This effect is not surprising when working solely with 3D depth data.

Third, we plot the average IoU of true positive bounding boxes as a function over their visibility. We define $visibility \in [0, 1]$ as $visibility := 1 - occlusion$, where the occlusion rate of a bounding box is calculated using a heuristic that measures the fraction of the number of pixels in a projected 2D image space bounding box compared to the number of points in the corresponding pointcloud that lie withing the 3D bounding box.

Finally, we visualize the average IoU of true positives as a function of their distance to the sensor. For higher distances sensor noise increases and cell resolution (as viewed in world space) diminishes. In consequence we can observe a decreasing trend.

Table 4. Comparison of the full FUS3D system inference speed on Jetson Nano (Device: Nano) and Jetson Orin Nano (Device: Orin), as well as effects of lower precision datatypes on runtime and performance metrics.

Datatype	Device	mAP	mAHS	MOTA	FPS
float32	Nano	43.20	39.24	58.0	6.4
float32	Orin	43.20	39.24	58.0	16.4
float16	Orin	42.34	38.53	56.6	23.6
int8	Orin	13.02	9.82	12.43	36.6

Overall the FUSED configuration yields superior results compared to the NMS configuration in all four cases.

Lower precision datatypes We present additional evaluations of our system's performance when utilizing lower precision datatypes (see Table 4). In place of the Jetson Nano target device, the presented evaluations are conducted using the more recent Jetson Orin Nano platform, introduced in March 2023. The decision to adopt this platform is based on two primary considerations. Firstly, the Jetson Nano's lack of support for the int8 datatype renders it unsuitable for the purpose. Secondly, although the float16 datatype is supported, deploying half precision models with small batch sizes on this device results in significant CPU overheads, a phenomenon that renders it impractical for practical implementation.

Using float16 precision inference, we observe a notable enhancement in inference speed accompanied by a slight reduction in detection and tracking performance. However, attempting to further reduce the model to int8 precision without quantization aware training leads to a severe degradation in accuracy, rendering the model unusable. Runtime evaluations using the TensorRT framework, reveal comparable speed results to a model fully traced and optimized for inference using PyTorch's built-in functionality.

In general, comparing the Jetson Nano platform to the newer Jetson Orin Nano platform, we note an 150% increase in inference speed when incorporating the tracking stage, and a speedup by roughly 200% when excluding the tracking stage.

Backbone evaluation In Table 5, we give an overview of CNN backbones tailored for mobile inference, along with their corresponding runtime and performance metrics when used in our detection stage. The final version of the FUS3D system uses the MobileNet v2 [31] architecture. Despite the availablility of more modern backbones such as MobileNetv3 [13], MnasNet [33] or EfficientNets [34, 35] we find it to give the best compromise between runtime and performance on our target devices. For the FUS3D system we observe higher detection accuracy than Mobilenetv3 and MnasNet, while displaying 65% faster inference speed than EfficientNet-B0.

Table 5. Comparison of light-weight CNN backbones when used in the FUS3D detection stage in terms of runtime and performance metrics.

Backbone	mAP	mAHS	FPS
Mobilenet v2 (Ours) [31]	44.6	42.4	11.4
Mobilenet v3 large [13]	40.3	38.8	12.6
Mobilenet v3 small [13]	38.6	37.5	**15.0**
MNasNet [33]	40.6	38.9	12.3
EfficientNet-B0 [34, 35]	**46.9**	**45.5**	6.9

Table 6. Evaluation of tracking performance and runtime of alternative two-stage systems compared to FUS3D. FPS numbers only reflect backbone inference speed.

Configuration	HOTA	MOTA	FPS
Ours	36.5%	58.0%	**11.4**
Mobilenet v2/DeepSORT	32.7%	60.4%	**11.4**
VoteNet/DeepSORT	**45.1%**	**74.7%**	1.3
H3DNet/DeepSORT	44.3%	73.0%	0.9

Comparison with two-stage approaches Finally, in Table 6, we provide a comparison between the tracking performance of our FUS3D system compared to alternative two-stage approaches. To maintain consistency with models already presented in our paper, we combine the VoteNet [29] and H3DNet [48] object detection models with the baseline DeepSORT tracker, resulting in the creation of two-stage systems. Notably, these systems are not specifically designed for fast inference times. We observe a 25% increase in tracking accuracy at the cost of a roughly 90% decrease in runtime. The frames per second (FPS) figures solely reflect the runtime of the detection model, ensuring a comparable basis for evaluation, as the FUS3D tracker operates on the GPU, while the DeepSORT tracker is executed on the CPU.

5. Conclusion

Discussion of limitations The FUS3D system is tailored to a specific set of circumstances and applications. Among the most predominant requirements is the use of a *static* depth sensor to capture a given scene. This enables the use of a background model, in some cases almost doubling detection performance in terms of mAP (Table 2) with negligible computational cost. Also related to the application setting is the given environment size. As discussed in Section 3.1, the subdivision of the sensor's field of view into cells is essential to both the object detection and tracking stages of our approach due to the cell-to-token transformation procedure. If the environment size L is uniformly increased while keeping cell volume at a constant scale, the number of cells required will grow at a rate of $\mathcal{O}(L^3)$ (or $\mathcal{O}(L^2)$ if we disregard the height dimension), which is the main reason why our method is inapplicable to datasets and applications related to the task of autonomous driving. Last, our method in its presented form does not utilize a multi-scale approach to obtain cells of different scales. With the exception of an increase in model size and higher runtime there is no inert reason preventing us from doing so. However, such approaches tend to improve performance in settings with objects of varied sizes. This is not the case in the presented work, which focuses predominantly on human detection and tracking – a setting featuring objects that are always of roughly equal size. In more general cases it may be desirable, or even necessary, to extend our approach in the suggested manner.

Reflection We have presented a new approach for simultaneous 3D object detection and multi-object-tracking in 3D depth data. The FUS3D system unifies both tasks into a single monolithic, end-to-end trainable neural network. It was developed with fast inference time as a primary design goal, and our evaluations have shown that inference speeds of $6 - 11$ frames per second can be achieved on the Nvidia Jetson Nano single-board computer. Our system has been extensively evaluated on the MIPT dataset which is intended for indoor 3D human detection and tracking. We have demonstrated that the FUS3D detection stage can act as a viable standalone object detection model, and have shown that our system can outperform all previous methods intended for inference on the edge. Detailed ablation studies have proven that all proposed components contribute positively to the overall performance of the system, both in the case of the detection and the tracking stage. Further studies of IoU and heading similarity distributions, as well as IoU as a function of bounding box visibility and distance in the context of object detection demonstrated the clear benefit of a trainable tracking stage.

Based on the frequently demonstrated ability of transformers to scale well with massive datasets, we believe that our approach has not yet reached its full potential. Therefore, for future work, we propose the development of larger and more diverse public databases that feature labeling for both 3D object detection and multi-object-tracking. Furthermore, since our method enables end-to-end training, we propose to investigate downstream tasks built on top of FUS3D such as human action or interaction recognition.

6. Acknowledgements

This work was partly supported by the Austrian Research Promotion Agency (FFG) under the Grant Agreement No. 879744 and the Vienna Science and Technology Fund (WWTF) under the Grant Agreement No. ICT20-055.

References

[1] Anton Andriyenko and Konrad Schindler. Multi-target tracking by continuous energy minimization. In *CVPR 2011*, pages 1265–1272. IEEE, 2011. 2

[2] Philipp Bergmann, Tim Meinhardt, and Laura Leal-Taixe. Tracking without bells and whistles. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 941–951, 2019. 2

[3] Keni Bernardin and Rainer Stiefelhagen. Evaluating multiple object tracking performance: the clear mot metrics. *EURASIP Journal on Image and Video Processing*, 2008:1–10, 2008. 7

[4] Luca Bertinetto, Jack Valmadre, Joao F Henriques, Andrea Vedaldi, and Philip HS Torr. Fully-convolutional siamese networks for object tracking. In *European conference on computer vision*, pages 850–865. Springer, 2016. 2

[5] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. Simple online and realtime tracking. In *2016 IEEE international conference on image processing (ICIP)*, pages 3464–3468. IEEE, 2016. 2

[6] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer, 2020. 2

[7] Yubo Cui, Zheng Fang, Jiayao Shan, Zuoxu Gu, and Sifan Zhou. 3d object tracking with transformer. *arXiv preprint arXiv:2110.14921*, 2021. 2

[8] Patrick Dendorfer, Hamid Rezatofighi, Anton Milan, Javen Shi, Daniel Cremers, Ian Reid, Stefan Roth, Konrad Schindler, and Laura Leal-Taixé. Mot20: A benchmark for multi object tracking in crowded scenes. *arXiv preprint arXiv:2003.09003*, 2020. 7

[9] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 2

[10] Mark Everingham, Luc Gool, Christopher K. Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *Int. J. Comput. Vision*, 88(2):303–338, June 2010. 6

[11] Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. Detect to track and track to detect. In *Proceedings of the IEEE international conference on computer vision*, pages 3038–3046, 2017. 2

[12] Thomas Heitzinger and Martin Kampel. A foundation for 3d human behavior detection in privacy-sensitive domains. In *32nd British Machine Vision Conference 2021, BMVC 2021, Online, November 22-25, 2021*, page 305. BMVA Press, 2021. 1, 3, 6

[13] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. Searching for mobilenetv3. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1314–1324, 2019. 6, 8, 9

[14] Andrew Jaegle, Felix Gimeno, Andy Brock, Oriol Vinyals, Andrew Zisserman, and Joao Carreira. Perceiver: General perception with iterative attention. In *International conference on machine learning*, pages 4651–4664. PMLR, 2021. 2, 5

[15] Shan Jiayao, Sifan Zhou, Yubo Cui, and Zheng Fang. Real-time 3d single object tracking with transformer. *IEEE Transactions on Multimedia*, 2022. 2

[16] Kai Kang, Hongsheng Li, Junjie Yan, Xingyu Zeng, Bin Yang, Tong Xiao, Cong Zhang, Zhe Wang, Ruohui Wang, Xiaogang Wang, et al. T-cnn: Tubelets with convolutional neural networks for object detection from videos. *IEEE Transactions on Circuits and Systems for Video Technology*, 28(10):2896–2907, 2017. 2

[17] Chanho Kim, Fuxin Li, Arridhana Ciptadi, and James M Rehg. Multiple hypothesis tracking revisited. In *Proceedings of the IEEE international conference on computer vision*, pages 4696–4704, 2015. 2

[18] Jason Ku, Melissa Mozifian, Jungwook Lee, Ali Harakeh, and Steven L Waslander. Joint 3d proposal generation and object detection from view aggregation. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1–8. IEEE, 2018. 6

[19] Miroslav Kubat. Neural networks: a comprehensive foundation by simon haykin, macmillan, 1994, isbn 0-02-352781-7. *The Knowledge Engineering Review*, 13(4):409–412, 1999. 5

[20] Harold W Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955. 2, 6

[21] Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 12697–12705, 2019. 6

[22] Laura Leal-Taixé, Michele Fenzi, Alina Kuznetsova, Bodo Rosenhahn, and Silvio Savarese. Learning an image-based motion context for multiple people tracking. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3542–3549, 2014. 2

[23] Justin Liang, Namdar Homayounfar, Wei-Chiu Ma, Yuwen Xiong, Rui Hu, and Raquel Urtasun. Polytransform: Deep polygon transformer for instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9131–9140, 2020. 2

[24] Ze Liu, Zheng Zhang, Yue Cao, Han Hu, and Xin Tong. Group-free 3d object detection via transformers. *arXiv preprint arXiv:2104.00678*, 2021. 6

[25] Jonathon Luiten, Aljosa Osep, Patrick Dendorfer, Philip Torr, Andreas Geiger, Laura Leal-Taixé, and Bastian Leibe. Hota: A higher order metric for evaluating multi-object tracking. *International journal of computer vision*, 129(2):548–578, 2021. 7

[26] Fan Ma, Mike Zheng Shou, Linchao Zhu, Haoqi Fan, Yilei Xu, Yi Yang, and Zhicheng Yan. Unified transformer tracker for object tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8781–8790, 2022. 2

[27] Tim Meinhardt, Alexander Kirillov, Laura Leal-Taixe, and Christoph Feichtenhofer. Trackformer: Multi-object tracking with transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8844–8854, 2022. 2, 5

[28] Stefano Pellegrini, Andreas Ess, Konrad Schindler, and Luc Van Gool. You'll never walk alone: Modeling social behavior for multi-target tracking. In *2009 IEEE 12th international conference on computer vision*, pages 261–268. IEEE, 2009. 2

[29] Charles R Qi, Or Litany, Kaiming He, and Leonidas J Guibas. Deep hough voting for 3d object detection in point clouds. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9277–9286, 2019. 6, 9

[30] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017. 2

[31] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4510–4520, 2018. 6, 8, 9

[32] Peize Sun, Jinkun Cao, Yi Jiang, Rufeng Zhang, Enze Xie, Zehuan Yuan, Changhu Wang, and Ping Luo. Transtrack: Multiple object tracking with transformer. *arXiv preprint arXiv:2012.15460*, 2020. 2, 5

[33] Mingxing Tan, Bo Chen, Ruoming Pang, Vijay Vasudevan, Mark Sandler, Andrew Howard, and Quoc V Le. Mnasnet: Platform-aware neural architecture search for mobile. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2820–2828, 2019. 6, 8, 9

[34] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019. 6, 8, 9

[35] Mingxing Tan and Quoc Le. Efficientnetv2: Smaller models and faster training. In *International Conference on Machine Learning*, pages 10096–10106. PMLR, 2021. 6, 8, 9

[36] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9627–9636, 2019. 4

[37] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 2, 5

[38] Zhongdao Wang, Hengshuang Zhao, Ya-Li Li, Shengjin Wang, Philip Torr, and Luca Bertinetto. Do different tracking tasks require different appearance models? *Advances in Neural Information Processing Systems*, 34:726–738, 2021. 2

[39] Zhongdao Wang, Liang Zheng, Yixuan Liu, Yali Li, and Shengjin Wang. Towards real-time multi-object tracking. In *European Conference on Computer Vision*, pages 107–122. Springer, 2020. 2

[40] Greg Welch, Gary Bishop, et al. An introduction to the kalman filter. 1995. 2

[41] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. Simple online and realtime tracking with a deep association metric. In *2017 IEEE international conference on image processing (ICIP)*, pages 3645–3649. IEEE, 2017. 2, 7

[42] Christopher Richard Wren, Ali Azarbayejani, Trevor Darrell, and Alex Paul Pentland. Pfinder: Real-time tracking of the human body. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 19(7):780–785, 1997. 3

[43] Jiarui Xu, Yue Cao, Zheng Zhang, and Han Hu. Spatial-temporal relation networks for multi-object tracking. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 3988–3998, 2019. 2

[44] Kota Yamaguchi, Alexander C Berg, Luis E Ortiz, and Tamara L Berg. Who are you with and where are you going? In *CVPR 2011*, pages 1345–1352. IEEE, 2011. 2

[45] Fengwei Yu, Wenbo Li, Quanquan Li, Yu Liu, Xiaohua Shi, and Junjie Yan. Poi: Multiple object tracking with high performance detection and appearance feature. In *European Conference on Computer Vision*, pages 36–42. Springer, 2016. 2

[46] Yifu Zhang, Chunyu Wang, Xinggang Wang, Wenjun Zeng, and Wenyu Liu. A simple baseline for multi-object tracking. *arXiv preprint arXiv:2004.01888*, 3(4):6, 2020. 2

[47] Zheng Zhang, Dazhi Cheng, Xizhou Zhu, Stephen Lin, and Jifeng Dai. Integrated object detection and tracking with tracklet-conditioned detection. *arXiv preprint arXiv:1811.11167*, 2018. 2

[48] Zaiwei Zhang, Bo Sun, Haitao Yang, and Qixing Huang. H3dnet: 3d object detection using hybrid geometric primitives. In *Proceedings of the European Conference on Computer Vision*, 2020. 6, 9

[49] Sixiao Zheng, Jiachen Lu, Hengshuang Zhao, Xiatian Zhu, Zekun Luo, Yabiao Wang, Yanwei Fu, Jianfeng Feng, Tao Xiang, Philip HS Torr, et al. Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6881–6890, 2021. 2

[50] Zhaohui Zheng, Ping Wang, Wei Liu, Jinze Li, Rongguang Ye, and Dongwei Ren. Distance-iou loss: Faster and better learning for bounding box regression. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 12993–13000, 2020. 6

[51] Changqing Zhou, Zhipeng Luo, Yueru Luo, Tianrui Liu, Liang Pan, Zhongang Cai, Haiyu Zhao, and Shijian Lu. Pttr: Relational 3d point cloud object tracking with transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8531–8540, 2022. 2

[52] Xingyi Zhou, Vladlen Koltun, and Philipp Krähenbühl. Tracking objects as points. In *European Conference on Computer Vision*, pages 474–490. Springer, 2020. 2

[53] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159*, 2020. 2