TransCenter: Transformers With Dense Representations for Multiple-Object Tracking

Yihong Xu†, Yutong Ban†, Guillaume Delorme, Chuang Gan†,
Daniela Rus†, Fellow, IEEE, and Xavier Alameda-Pineda†, Senior Member, IEEE

Abstract—Transformers have proven superior performance for a wide variety of tasks since they were introduced. In recent years, they have drawn attention from the vision community in tasks such as image classification and object detection. Despite this wave, an accurate and efficient multiple-object tracking (MOT) method based on transformers is yet to be designed. We argue that the direct application of a transformer architecture with quadratic complexity and insufficient noise-initialized sparse queries — is not optimal for MOT. We propose TransCenter, a transformer-based MOT architecture with dense representations for accurately tracking all the objects while keeping a reasonable runtime. Methodologically, we propose the use of image-related dense detection queries and efficient sparse tracking queries produced by our carefully designed query learning networks (QLN). On one hand, the dense image-related detection queries allow us to infer targets’ locations globally and robustly through dense heatmap outputs. On the other hand, the set of sparse tracking queries efficiently interacts with image features in our TransCenter Decoder to associate object positions through time. As a result, TransCenter exhibits remarkable performance improvements and outperforms by a large margin the current state-of-the-art methods in two standard MOT benchmarks with two tracking settings (public/private). TransCenter is also proven efficient and accurate by an extensive ablation study and comparisons to more naive alternatives and concurrent works. The code is made publicly available at https://github.com/yihongxu/transcenter.

Index Terms—Multiple-object tracking, efficient transformer, dense image-related detection queries, sparse tracking queries

1 INTRODUCTION

The task of tracking multiple objects, usually understood as the simultaneous inference of the positions and identities of various objects/pedestrians (trajectories) in a visual scene recorded by one or more cameras, became a core problem in computer vision in the past years. Undoubtedly, the various multiple-object tracking (MOT) challenges and associated datasets [11], [46], helped foster research on this topic and provided a standard way to evaluate and monitor the performance of the methods proposed by many research teams worldwide.

Recent progress in computer vision using transformers [70] for tasks such as object detection [8], [39], [96], person re-identification (Re-ID) [25] or image super resolution [81], showed the benefit of the attention-based mechanism. Transformers are good at modeling simultaneously dependencies between different parts of the input and thus at making global decisions. These advantages fit perfectly the underlying challenges of MOT, where current methods often struggle when modeling the interaction between objects, especially in crowded scenes. We, therefore, are motivated to investigate the use of a transformer-based architecture for MOT, enabling global estimations when finding trajectories thus reducing missed or noisy predictions of trajectories.

Current MOT methods follow in principle the predominant tracking-by-detection paradigm where we first detect objects in the visual scene and associate them through time. The detection ability is critical for having a good MOT performance. To this end, MOT methods usually combine probabilistic models [2], [3], [41], [55] or deep convolutional architectures [4], [17], [50], [60], [73], [80], [86] with an integrated or external detector to predict bounding-box outputs. They are often based on overlapping predefined anchors that predict redundant outputs, which might create noisy detections, and thus need hand-crafted post-processing techniques such as non-maximum suppression (NMS), which might suppress correct detections. Instead, concurrent MOT methods [45], [64] built on recent object detectors like DETR [8] leverage transformers where sparse noise-initialized queries are employed, which perform cross-attention with encoded images to output one-to-one object-position predictions, thanks to the one-to-one assignment of objects and queries during training. However, using a fixed-number of sparse queries leads to a lack of detections when the visual scene becomes crowded, and the hyper-
parameters of the model need to be readjusted. For these reasons, we argue that having dense but non-overlapping representations for detection is beneficial in MOT, especially in crowded scenes.

Indeed, the image-size dense queries are naturally compatible with center heatmap predictions, and there exist center-based MOT methods [88], [93] that output dense object-center heatmap predictions directly related to the input image. Moreover, the center heatmap representations are pixel-related and thus inherit the one-to-one assignment of object centers and queries (pixels) without external assignment algorithms. However, the locality of CNN architecture limits the networks to explore, as transformers do, the co-dependency of objects globally. Therefore, we believe that the transformer-based approach with dense image-related (thus non-overlapping) representations is a better choice for building a powerful MOT method.

Designing such transformer-based MOT with dense image-related representations is far from evident. The main drawback is the computational efficiency related to the quadratic complexity in transformers w.r.t. the dense inputs. To overcome this problem, we introduce TransCenter, with powerful and efficient attention-based encoder and decoder, as well as the query generator.

For the encoder, the DETR [8] structure alleviates this issue by introducing a pre-feature extractor like ResNet-50 to extract lower-scale features before inputting the images to the transformers, but the CNN feature extractor itself contributes to the network complexity. Deformable transformer [96] reduces significantly the attention complexity but the ResNet-50 feature extractor is still kept. Alternatively, recent efficient transformers like [72] discard the pre-feature extractor and input directly image patches, following a ViT [13] or CNN structure. They also reduce the attention complexity in transformers by spatial-reduction attention, which makes building efficient transformer-based MOT possible.

In TransCenter Decoder, the use of deformable transformer [96] indeed reduces the attention complexity. However, with our dense queries, the attention calculation remains heavy, which hinders computational efficiency. Alternatively, our image-size detection queries are generated from the lightweight query learning networks (QLN) that input image features from the encoder, and they are already globally related. Therefore, the heavy detection cross-attention with image features can be omitted. Furthermore, the tracking happens in the decoder where we search the current object positions with the known previous ones. With this prior positional information, we design the tracking queries to be sparse, which do not need to search every pixel for finding current object positions and such design significantly speeds up the MOT method without losing accuracy.

To summarize, as roughly illustrated in Fig. 1, TransCenter tackles the MOT problem with image-related dense detection queries and sparse tracking queries. It (i) introduces dense image-related queries to deal with miss detections from insufficient sparse queries or over detections in overlapping anchors of current MOT methods, yielding better MOT accuracy; (ii) solves the efficiency issue inherited in transformers with careful network designs, sparse tracking queries, and removal of useless external overheads, allowing TransCenter to perform MOT efficiently. Overall, this work has the following contributions:

- We introduce TransCenter, the first center-based transformer framework for MOT and among the first to show the benefits of using transformer-based architectures for MOT.
- We carefully explore different network structures to combine the transformer with center representations, specifically proposing dense image-related multi-scale representations that are mutually correlated within the transformer attention and produce abundant but less noisy tracks, while keeping a good computational efficiency.
- Moreover, two more model options, TransCenter-Dual, which further boosts the performance in crowded scenes, and TransCenter-Lite, enhancing the computational efficiency, are provided for different requirements in MOT applications.

Finally, comparing extensively with up-to-date online MOT tracking methods, TransCenter sets a new state-of-the-art baseline both in MOT17 [46] (+4.0% Multiple-Object Tracking Accuracy, MOTA) and MOT20 [11] (+18.8% MOTA) by large margins, leading both MOT competitions by the time of our submission in the published literature.

2 RELATED WORKS
2.1 Multiple-Object Tracking

In MOT literature, initial works [1], [3], [55] focus on how to find the optimal associations between detections and tracks through probabilistic models while [47] first formulates the problem as an end-to-end learning task with recurrent neural networks. Moreover, [57] models the dynamics of objects by a recurrent network and further combines the dynamics with an interaction and an appearance branch. [80] proposes a framework to directly use the standard evaluation measures MOTA and MOTP as loss functions to back-propagate the metrics for an end-to-end tracking system. [4] employs object detection methods for MOT by modeling the problem as a regression task. A person Re-ID network [4], [67] can be added at the second stage to boost the performance. However, it is still not optimal to treat the Re-ID as a secondary task. [88] further proposes a framework that
Fig. 2. Generic pipeline of TransCenter and different variants: Images at $t$ and $t-1$ are fed to the transformer encoder (DETR-Encoder or PVT-Encoder) to produce multi-scale memories $M_t$ and $M_{t-1}$, respectively. They are passed (together with track positions at $t-1$) to the Query Learning Networks (QLN) operating in the feature’s channel. QLN produce (1) dense pixel-level multi-scale detection queries–$DQ$, (2) detection memory–$DM$, (3) (sparse or dense) tracking queries–$TQ$, (4) tracking memory–$TM$. For associating objects through frames, the TransCenter Decoder performs cross attention between $TQ$ and $TM$, producing Tracking Features–$TF$. For detection, the TransCenter Decoder either calculates the cross attention between $DQ$ and $DM$ or directly outputs $DQ$ (in our efficient versions, TransCenter and TransCenter-Lite, see Section 3), resulting in Detection Features–$DF$ for the output branches, $S_t$ and $C_t$: $TF$, together with object positions at $t-1$ (sparse $TQ$) or center heatmap $C_{t-1}$ (omitted in the figure for simplicity) and $DF$ (dense $TQ$), are used to estimate image center displacements $T$: indicating for each center its displacement in the adjacent frames (red arrows). We detail our choice (TransCenter) of QLN and TransCenter Decoder structures in the figure. Other designs of QLN and TransCenter Decoder are detailed in Figs. 3 and 4. Arrows with a dotted line are only necessary for models with sparse $TQ$.

treats the person detection and Re-ID tasks equally and [48] uses quasi-dense proposals from anchor-based detectors to learn Re-ID features for MOT in a dense manner. [62] proposes a fast online MOT with detection refinement. Motion clues in MOT are also important since a well-designed motion model can compensate for missing detections due to occlusions. To this end, [82] unifies the motion and affinity models for MOT. [43] leverages articulation information to perform MOT. [22], [58] focus on the motion-based interpolation based on probabilistic models. [77] unifies the segmentation and detection task in an MOT framework combining historic tracking information as a strong clue for the data association. Derived from the success in single-object tracking, [63] employs siamese networks for MOT. [87] simply leverages low-score detections to explore partially occluded objects. Moreover, traditional graphs are also used to model the positions of objects as nodes and the temporal connection of the objects as edges [26], [32], [65], [66], [67]. The performance of those methods is further boosted by the recent rise of Graph Neural Networks (GNNs): hand-designed graphs are replaced by learnable GNNs [7], [23], [49], [73], [75], [76] to model the complex interaction of the objects.

Most of the above methods follow the tracking by detections/regression paradigm where a detector provides detections for tracking. The paradigm has proven state-of-the-art performance with the progress in object detectors. Unlike classic anchor-based detectors [53], [54], recent progress in keypoint-based detectors [35], [94] exhibited better performance while discarding overlapping and manually-designed box anchors trying to cover all possible object shapes and positions. Built on keypoint-based detectors, [93], [88] and [92] represent objects as centers in a heatmap then reason about all the objects jointly and associate them across adjacent frames with a tracking branch or Re-ID branch.

2.2 Transformers in Multiple-Object Tracking

Transformer is first proposed by [70] for machine translation and has shown its ability to handle long-term complex dependencies between entries in a sequence by using a multi-head attention mechanism. With its success in natural language processing, works in computer vision start to investigate transformers for various tasks, such as image recognition [13], Person Re-ID [25], realistic image generation [29], super-resolution [81] and audio-visual learning [15], [16].

Object detection with Transformer (DETR) [8] can be seen as an exploration and correlation task. It is an encoder-decoder structure where the encoder extracts the image information and the decoder finds the best correlation between the object queries and the encoded image features with an attention module. The attention module transforms the inputs into Query ($Q$), Key ($K$), and Value ($V$) with fully-connected layers. Having $Q$, $K$, $V$, the attended features are calculated with the attention function [70]:

\[
\text{Attention}(Q, K, V) = \text{Softmax} \left( \frac{QK^T}{\sqrt{h}} \right) V
\]

where $h$ is the hidden dimension of $Q$, $K$, and $V$. The attention calculation suffers from heavy computational and memory complexities w.r.t the input size: the feature maps extracted from a ResNet-50 [24] backbone are used to alleviate the problem. Deformable DETR [96] further tackles the issue by proposing deformable attention inspired by [10], drastically speeding up the convergence (10x) and reducing the complexity. The reduction of memory consumption allows in practice using multi-scale features to capture finer details, yielding better detection performance. However, the CNN backbone is still kept in Deformable DETR. This important overhead hinders the Transformer from being efficient. Alternatively, Pyramid Vision Transformer [72] (PVT) extracts the visual features directly from the input images and the attention is calculated with efficient spatial-reduction attention (SRA). Precisely, PVT follows the ViT [13] structure while the feature maps are gradually down-scaled with a patch embedding module (with convolutional layers and layer normalization). To reduce the quadratic complexity of Eq. (1) w.r.t the sequence length $d$ of $Q$, $K$, $V$, the SRA in PVT reduces
beforehand the dimension of $K, V$ from $R^{d \times h}$ to $R^{d \times h}$ (with $r > 1$, the scaling factor) and keeps the dimension of $Q$ unchanged. The complexity can be reduced by $r^2$ times while the size of the output attended features remains unchanged, boosting the efficiency of transformer attention modules.

The use of transformers is still recent in MOT. Before transformers, some attempts with simple attention-based modules have been introduced for MOT. Specifically, [21] proposes a target and distractor-aware attention module to produce more reliable appearance embeddings, which also helps suppress detection drift and [71] proposes hand-designed spatial and temporal correlation modules to achieve long-range information similar to what transformers inherit. After the success in detection using transformers, two concurrent works directly apply transformers on MOT based on the (deformable) DETR framework. First, Trackformer [45] builds directly from DETR [8] and is trained to propagate the queries through time. Second, Transtrack [64] extends [96] to MOT by adding a decoder that processes the features at $t - 1$ to refine previous detection positions. Importantly, both methods stay in the DETR framework with sparse queries and extend it for tracking. However, recent literature [88], [92], [93] also suggests that point-based tracking may be a better option for MOT while the use of pixel-level dense queries with transformers to predict dense heatmaps for MOT has never been studied. In addition, we question the direct transfer from DETR to MOT as concurrent works do [45], [64]. Indeed, the sparse queries without positional correlations might be problematic in two folds. First, the insufficient number of queries could cause severe miss detections thus false negatives (FN) in tracking. Second, queries are highly overlapping, and simply increasing the number of non-positional-correlated queries may end up having many false detections and false positives (FP) in tracking. All of the above motivate us to investigate a better transformer-based MOT framework.

Therefore, we introduce TransCenter a novel design that takes existing drawbacks into account and achieves state-of-the-art performance.

3 TransCenter

TransCenter tackles the detection and temporal association in MOT accurately and efficiently. Different from concurrent transformer-based MOT methods, TransCenter questions the use of sparse queries without image correlations (i.e., noise initialized), and explores the use of image-related dense queries, producing dense representations using transformers. To that aim, we introduce the query learning networks (QLN) responsible for converting the outputs of the encoder into the inputs of the TransCenter Decoder. Different possible architectures for QLN and TransCenter Decoder are proposed, and the choice is made based on both the accuracy and efficiency aspects.

While exploiting dense representations from dense queries\(^1\) can help to sufficiently detect the objects, especially in crowded scenes, the design of dense queries is not trivial. Notably, the quadratic increase of calculation complexity in transformers should be solved, and the noisy tracks from randomly initialized dense queries should be addressed. To this end, we propose image-related dense queries, which have three prominent advantages: (1) the queries are multi-scale and exploit the multi-resolution structure of the encoder, allowing for targets of different scales to be captured by those queries; (2) image-related dense detection queries also make the network more flexible. The number of queries grows with the input resolution. No fixed hyper-parameters as in [64] are needed to be adjusted to re-train the model, which depends on the density of objects in the scene; (3) the query-pixel correspondence discards the time-consuming Hungarian matching [34] for the query-ground-truth object association. Up to our knowledge, we are the first to explore the use of image-related dense detection queries that scale with the input image size. Meanwhile, we

\(^1\)See visualization in Supplementary Material Section B.2., available online.
solve the efficiency issue through a careful network design that appropriately handles the dense representation of queries, which provides an accurate and efficient MOT method.

A generic pipeline of TransCenter is illustrated in Fig. 2. RGB images at $t$ and $t-1$ are input to the weight-shared transformer encoder from which dense multi-scale attended features are obtained, namely memories $M_t$ and $M_{t-1}$. They are the inputs of the QLN. QLN produces two sets of output pairs, detection queries (DQ) and memory (DM) for detecting the objects at time $t$, and tracking queries (TQ) and memory (TM) for associating the objects at $t$ with previous time step $t-1$. Furthermore, TransCenter Decoder, leveraging the deformable transformer [96], is used to correlate DQ/TQ with DM/TM. To elaborate, TQ interacts with TM in the cross-attention module of the TransCenter Decoder, resulting in the tracking features (TF). Similarly, the detection features (DF) are the output of the cross-attention between DQ and DM. To produce the output dense representations, DF are used to estimate the object size $S$, and the center heatmap $C_t$. Meanwhile, TF are used to estimate the tracking displacement $T_t$.

One can argue that the downside of using dense queries is the associated higher memory consumption and lower computational efficiency. One drawback with previous or concurrent approaches is the use of the deformable DETR encoder including the CNN feature extractor ResNet-50 [24], which significantly slows down the feature extraction. Instead, TransCenter leverages PVTv2 [72] as its encoder, the so-called PVT-Encoder and the reasons are three-fold: (1) it discards the ResNet-50 backbone and uses efficient attention heads [72], reducing significantly the network complexity; (2) it has flexible scalability by modifying the feature dimension $h$ and block structures. Specifically, we use B0 (denoted as PVT-Lite) in TransCenter-Lite and B2 (PVT-Encoder) for TransCenter and TransCenter-Dual, see details in [72] and Section 4.1; (3) its feature pyramid structure is suitable for building dense pixel-level multi-scale queries.

Once the transformer encoder extracts the dense memory representations $M_t$ and $M_{t-1}$, they are passed to QLN and then to the TransCenter Decoder. We carefully search the design choices of QLN (see Section 3.1) and TransCenter Decoder (see Section 3.2), and select the best model based on the efficiency and accuracy. In particular, we demonstrate that TQ can be sparse 2) different from DQ, since we have the prior information of object positions at the previous time step that helps to search their corresponding positions at the current time step. Therefore, it is not necessary to search every pixel for this aim. Consequently, the reduced number of tracking queries based on previous object positions (roughly from 14k to less than 500 depending on the number of objects) can significantly speed up the tracking attention calculation in the TransCenter Decoder.

Regarding the detection, the cross attention module between the dense detection queries (DQ) and the detection memory (DM) is beneficial in terms of performance but at the cost of significant computational loads. We solve this by studying the impact of the detection cross-attention on the computational efficiency and the accuracy. To this end, we introduce a variant of the proposed method, TransCenter-Dual, which shares the same structure as TransCenter but having detection cross attention in the decoder part, DDCA as detailed in Section 3.2. To further assess the accuracy-efficiency tradeoff, we provide a light version of TransCenter with a lighter encoder (PVT-Lite), denoted as TransCenter-Lite.

In the following sections, we detail the design choices of the QLN and the TransCenter Decoder, and provide the details of the final output branches (see Section 3.3) as well as the training losses (see Section 3.4).

### 3.1 QLN: Query Learning Networks

QLN are networks that relate the queries and the memories. In our design, (1) DQ are dense and image-related that discover object positions in every pixel; (2) Different from DQ, TQ are sparse and aim at finding object displacements between two different frames, and thus TQ and TM should be produced by input features from different time steps.

Based on these attributes, we design the chosen QLN $Q_{S\rightarrow}$ that produces DQ by passing $M_t$ (attended features from image $t$) through a FFN (feed-forward network with fully-connected layers). For TQ, $Q_{S\rightarrow}$ samples object features with a feature sampler using bilinear interpolation from $M_{t-1}$ using object positions at $t-1$, while outputting TM from features at a different time step, $M_t$. Moreover, different possible variants of QLN are visualized in Fig. 3 and are summarized as follows:

1) $Q_{L\rightarrow}$: $Q_{L\rightarrow}$ with dense (with the subscript “D”) TQ without the feature sampler (feature cropping based on positions with bilinear sampling).
2) $Q_{L\rightarrow}$: $Q_{L\rightarrow}$ with dense TQ from $M_t$ and TM from $M_{t-1}$.
3) $Q_{D\rightarrow}$: $Q_{D\rightarrow}$ with dense TQ from DQ and TM from $M_{t-1}$.
4) $Q_{E}$: $Q_{E}$ with noise-initialized learnable embeddings, denoted with the subscript “E”.
5) $Q_{S\rightarrow}$: $Q_{S\rightarrow}$ with noise-initialized learnable embeddings for detection.

In Section 4.5, different QLN are carefully designed and ablated to produce dense detection queries relative to the input image and sparse tracking queries for accurate and efficient MOT with transformers.

### 3.2 TransCenter Decoder

To successfully find object trajectories, a MOT method should not only detect the objects but also associate them across frames. To do so, TransCenter Decoder tackles in parallel the two sub-tasks: detection and object temporal association (tracking). Concretely, TransCenter Decoder consists of Tracking Deformable Cross-Attention (TDCA), and Detection Deformable Cross-Attention (DDCA) modules.

---

2. See visualization in Supplementary Material Section B.3., available online
Moreover, Tracking Query Self-Attention (TQSA) module is introduced to enhance the interactions among sparse tracking queries through a multi-head self-attention, knowing that the overhead is acceptable because the tracking queries are sparse in TransCenter. TDCA calculates cross attention between tracking queries and memories (TQ and TM), resulting in tracking features (TF). Analogously, DDCA calculates cross attention between detection queries and memories (DQ and DM), producing detection features (DF). Deformable cross attention module [96] with linear complexity w.r.t. input size is used.

From the efficiency perspective, the use of the multi-head attention modules in traditional transformers [70] like DETR [8] implies a complexity growth that is quadratic w.r.t. the input size. Of course, this is undesirable and would limit the scalability and usability of the method. To mitigate this, we resort to the deformable multi-head attention [96–98]—Deformable Cross-Attention (DCA), where the queries are input to produce sampling offsets for displacing the input reference points. The reference points are from either the track position at $t-1$ for tracking in TDCA or the pixel coordinates of the dense queries for detection in DDCA.

The displaced coordinates are used to locate and sample features in DM or TM. The input queries also produce in DCA the attention weights for merging sampled features. However, the cost of calculating the cross attention between DQ and DM is still not negligible because of their multi-scale image resolutions. To solve this, we demonstrate in Section 4.5 (Single versus Dual decoder) that it is possible to output directly DQ as DF for output-branch predictions, with an acceptable loss of accuracy as expected. In addition, under this sparse nature of TQ in TransCenter, we enhance the interactions among queries by adding lightweight TQSA before TDCA.

To conclude, we choose to use a TQSA-Single TransCenter Decoder for the cross attention of TQ and TM while we directly use DQ for the output branches. This is possible thanks to the sparse TQ and dense DQ, which yields a good balance between computational efficiency and accuracy, the overall Decoder design is illustrated in Fig. 4 and the comparison between different design choices are provided in Section 4.5.

### 3.3 The Center, the Size, and the Tracking Branches

Given DF and TF from the TransCenter Decoder, we use them as the input of different branches to output object center heatmap $C_t$, their box size $S_t$, and the tracking displacements $T_t$. DF contain feature maps of four different resolutions, namely $1/32, 1/16, 1/8, 1/4$ of the input image resolution. For the center heatmap and the object size, the feature maps at different resolutions are combined using deformable convolutions [10] and bilinear interpolation up-sampling, following the architecture shown in Fig. 5. They are up-sampled into a feature map of $1/4$ of the input resolution and then input to $C_t \in [0, 1]^{H/4 \times W/4}$ and $S_t \in \mathbb{R}^{H/4 \times W/4 \times 2}$. $H$ and $W$ are the input image height and width, respectively, and the two channels of $S_t$ encode the object size in width and height.

Regarding the tracking branch, the tracking features TF are sparse having the same size as the number of tracks at $t-1$. One tracking query feature corresponds to one track at $t-1$. TF, together with object positions at $t-1$ (in the case of sparse TQ) or center heatmap $C_{t-1}$ and DF (dense TQ), are input to two fully-connected layers with ReLU activation. The latter layers predict the horizontal and vertical displacements $T_t$ of objects at $t-1$ in the adjacent frames.

### 3.4 Model Training

The model training is achieved by jointly learning a 2D classification task for the object center heatmap and a regression task for the object size and tracking displacements, covering the branches of TransCenter. For the sake of clarity, in this section, we will drop the time index $t$.

**Center Focal Loss.** To train the center branch, we need first to build the ground-truth heatmap response $C^* \in [0, 1]^{H/4 \times W/4}$. As done in [93], we construct $C^*$ by considering the maximum response of a set of Gaussian kernels centered at each of the $K > 0$ ground-truth object centers. More formally, for every pixel position $(x, y)$ the ground-truth heatmap response is computed as:

$$C^*_{xy} = \max_{k=1,...,K} G((x, y), (x_k, y_k); \sigma),$$

where $(x_k, y_k)$ is the ground-truth object center, and $G(\cdot; \cdot; \sigma)$ is the Gaussian kernel with spread $\sigma$. In our case, $\sigma$ is proportional to the object’s size, as described in [35]. Given the ground-truth $C^*$ and the inferred $C$ center heatmaps, the center focal loss, $L_c$, is formulated as:

$$L_c = \frac{1}{K} \sum_{xy} \left\{ \frac{1}{\alpha} \log (C^*_{xy}) \frac{1}{(1 - C^*_{xy})^\beta} \log (1 - C_{xy}) \right\} \text{if } C^*_{xy} = 1,$$

$$\frac{1}{(1 - C^*_{xy})^\beta} \log (1 - C_{xy}) \text{otherwise.}$$

where the scaling factors are $\alpha = 2$ and $\beta = 4$, see [88].

---

5. The reference points are omitted in the figures for simplicity.
Sparse Regression Loss. The values of $S$ is supervised only on the locations where object centers are present, i.e., $C_{xy}^* = 1$ using a $L_1$ loss:

$$L_s = \frac{1}{K} \sum_{xy} \left\{ \begin{array}{ll} \|S_{xy} - S_{xy}^*\|_1 & C_{xy}^* = 1, \\ 0 & \text{otherwise.} \end{array} \right. \quad (4)$$

The formulation of $L_s$ for $T$ is analogous to $L_s$ but using the tracking output and ground-truth displacement, instead of the object size. To complete the sparsity of $L_s$, we add an extra $L_1$ regression loss, denoted as $L_R$ with the bounding boxes computed from $S_t$ and ground-truth centers. To summarize, the overall loss is formulated as the weighted sum of all the losses, where the weights are chosen according to the numeric scale of each loss:

$$L = L_c + \lambda_s L_s + \lambda_t L_t + \lambda_R L_R \quad (5)$$

4 EXPERIMENTAL EVALUATION

4.1 Implementation Details

Inference With TransCenter. Once the method is trained, we detect objects by selecting the maximum responses from the output center heatmap $C_t$. Since the datasets are annotated with bounding boxes, we need to convert our estimates into this representation. In detail, we apply (after max pooling) a threshold $r$ (e.g., 0.3 for MOT17 and 0.4 for MOT20 in TransCenter) to the center heatmap, thus producing a list of center positions $\{c_{t,1}\}_{i=1}^{K_t}$. We extract the object size $s_{i,k}$ associated to each position $c_{i,k}$ in $S$. The set of detections produced by TransCenter is denoted as $D_t = \{c_{i,k}, s_{i,k}\}_{k=1}^{K_t}$. In parallel, for associating objects through frames (tracking), given the position of an object $k$ at $t - 1$, we can estimate the object position at the current frame by extracting the corresponding displacement estimate $t_{i,k}$ from $T_i$. Therefore, we can construct a set of tracked positions $P_t = \{c_{t-1,k} + t_{i,k}, s_{i,k}\}_{k=1}^{K_t}$. Finally, we use the Hungarian algorithm [34] to match at $t$ the displaced and tracked positions – $P_t$ and the detection – $D_t$. The matched detections are used to update the tracked object positions at $t$. The birth and death processes are naturally integrated in TransCenter: detections not associated with any tracked object give birth to new tracks, while unmatched tracks are put to sleep for at most $T = 60$ frames before being discarded. An external Re-ID network is often used in MOT methods [4] to recover tracks in sleep, which is proven unnecessary in our experiment in Section 4.5. We also assess inference speed in fps on the test-set results either obtained from [87] or tested under the same GPU setting.

Network and Training Parameters. The input images are resized to $640 \times 1088$ with padding in TransCenter and TransCenter-Dual while it is set to $608 \times 1088$ in TransCenter-Lite. In TransCenter, the PVT-Encoder has $[3, 4, 6, 3]$ layers ($[2, 2, 2, 2]$ in PVT-Lite) for each image feature scale and the corresponding hidden dimension $h = [64, 128, 320, 512]$ ($h = [32, 64, 160, 256]$ in PVT-Lite). $h = 256$ for the TransCenter Decoder with eight attention heads and six layers (four layers in TransCenter-Lite). All TransCenter models are trained with loss weights $\lambda_s = 0.1$, $\lambda_T = 0.5$ and $\lambda_T = 1.0$ by the AdamW optimizer [44] with learning rate $2e^{-4}$. The training converges at around 50 epochs, applying a learning rate decay of $1/10$ at the 40th epoch. The entire network is pre-trained on the pedestrian class of COCO [40] and then fine-tuned on the respective MOT dataset [11, 46]. We also present the results finetuned with extra data like CrowdHuman dataset [61] (see Section 4.3 for details).

4.2 Protocol

Datasets and Detections. We use the standard split of the MOT17 [46] and MOT20 [11] datasets and the testset evaluation is obtained by submitting the results to the MOTChallenge website. The MOT17 testset contains 2,355 trajectories distributed in 17,757 frames. MOT20 testset contains 1,501 trajectories within only 4,479 frames, which leads to a much more challenging crowded-scene setting. We evaluate TransCenter both under public and private detections. When using public detections, we limit the maximum number of birth candidates at each frame to the number of public detections per frame, as in [45, 93]. The selected birth candidates are those closest to the public detections with IOU larger than 0. When using private detections, there is no constraint, and the detections depend only on the network’s detection capacity, the use of external detectors, and more importantly, the use of extra training data. For this reason, we regroup the state-of-the-art comparisons by the use of extra training datasets as detailed in the following. In addition, we evaluate our TransCenter on the KITTI dataset under the autonomous driving setting. It contains annotations of vehicles and pedestrians in 21 and 29 video sequences in the training and the test set, respectively. To train the model for the KITTI dataset, we use also [68] as extra data.

Extra Training Data. To fairly compare with the state-of-the-art methods, we denote the extra data used to train each method, including several pre-prints listed in the MOTChallenge leaderboard, which are marked with * in our result tables: CH for CrowdHuman [61], RR for PathTrack [59], RR1 for the combination of Market1501 [90], CUHK01 and CUHK03 [36] person re-identification datasets, RTE2 replaces CUHK01 [36] with DukeMTMC [56], 5o1 for the use of five extra datasets (ETHZ [14], Caltech Pedestrian [12], [51], CityPersons [85], CUHK-SYS [79], and PRW [91]), 5o1+CH is the same as 5o1 plus CrowdHuman and (5o1+CH) refers to the tracking/detection results of FairMOT [88] trained within the 5o1+CH setting, no stands for using no extra dataset.

Metrics. Standard MOT metrics such as MOTA (Multiple Object Tracking Accuracy) and MOTP (Multiple Object Tracking Precision) [5] are used: MOTA is mostly used since it reflects the average tracking performance including the number of FP (false positives, predicted bounding boxes not enclosing any object), FN (false negatives, missing ground-truth objects) and IDS [37] (identities of predicted trajectories switch through time). MOTP evaluates the quality of bounding boxes from successfully tracked objects. Moreover, we also evaluate IDF1 [56] (the ratio of correctly
identified detections over the average number of ground-truth objects and predicted tracks), MT (the ratio of ground-truth trajectories that are covered by a track hypothesis more than 80% of their life span), and ML (less than 20% of their life span).

4.3 Testset Results and Discussion

MOT17. Table 1 presents the results obtained in the MOT17 testset. The first global remark is that most state-of-the-art methods do not evaluate under both public and private detections, and under different extra-training data settings, while we do. Second, TransCentersets new state-of-the-art performance compared to other methods, in terms of MOTA, under CH and no-extra training data conditions, both for public and private detections. Precisely, the increase of MOTA w.r.t. the state-of-the-art methods is of 8.5% and 4.8% (both including unpublished methods by now) for the public detection setting under CH and no-extra training data, and of 1.7% and 4.0% for the private detection setting, respectively. We can also observe that TransCenter trained with no extra-training data outperforms, not only the methods trained with no extra data but also some methods trained with one extra dataset. Similarly, TransCenter trained on CH performs better than seven methods trained with five or more extra datasets in the private setting, comparable to the best result in 5d1+ch (-0.3% MOTA), showing that TransCenter is less data-hungry. Moreover, trained with 5d1+ch, the performance is further improved while running at around 11 fps. Overall, these results confirm our hypothesis that TransCenter with dense detection representations and sparse tracking representations produced by global relevant queries in transformers is a better choice.

MOT20. Table 2 reports the results obtained in MOT20 testset. In all settings, similar to the case in MOT17, TransCenter leads the competition by a large margin compared to all the other methods. Concretely, TransCenter outperforms current methods by +19.2%/+8.4% in MOTA with the public/private setting trained with CH and +11.1%/18.8% without extra data. From the results, another remarkable achievement of TransCenter is the significant decrease of FN while keeping a relatively low FP number. This indicates that the dense representation of the detection queries can help effectively detect objects sufficiently and accurately. As for tracking, TransCenter maintains low IDS numbers in MOT20 running at around 8 fps in such crowded scenes, thanks to our careful choices of QLN and the TransCenter Decoder. Very importantly, to the best of our knowledge, our study is the first to report the results of all settings on MOT20, demonstrating the tracking capacity of TransCenter even in a densely crowded scenario. The outstanding results of TransCenter in MOT20 further show the effectiveness of our design.

KITTI. Additionally, we show the results of TransCenter evaluated on the KITTI dataset. TransCenter significantly outperforms CenterTrack [93] in pedestrian tracking (+5.3% MOTA) while keeping a close performance in car tracking. However, the KITTI dataset is constructed in an

| Method                  | Data | MOTA | IDP | MT | ML | FP | FN | IDS | FPS |
|-------------------------|------|------|-----|----|----|----|----|-----|-----|
| MOT17 [49]              | 5d1  | 76.0 | 45.9 | 12.1 | 30,190 | 100,999 | 4,626 | 11.7 |     |
| *UnsupTrack [30]        | CH   | 63.0 | 45.6 | 13.6 | 70,443 | 123,532 | 3,528 | 6.8 |     |
| GMx [63]                | CH   | 78.8 | 64.7 | 34.6 | 19,489 | 160,332 | 3,939 | 17.5 |     |
| TrackCenter [45]        | CH   | 65.6 | 34.6 | 21.9 | 199,595 | 3,040 | 12.9 |     |
| *MOT17 [44]             | CH   | 67.4 | 34.6 | 24.5 | 194,600 | 4,992 | 7.3  |     |
| CenterTrack [93]        | CH   | 75.9 | 61.3 | 49.8 | 30,190 | 100,999 | 4,626 | 11.7 |     |
| TratDet [77]            | CH   | 73.8 | 68.9 | 43.8 | 17.2 | 28,998 | 114,104 | 3,699 | 11.9 |
| PermsTrack [69]         | CH   | 74.5 | 68.6 | 46.8 | 11.3 | 28,323 | 112,137 | 4,663 | 10.0 |
| *TrackNet [64]          | CH   | 51.3 | 69.5 | 51.5 | 7.9 | 40,101 | 89,827 | 5,394 | 11.8 |

The cell background color encodes the amount of extra-training data: green for none, orange for one extra dataset, red for (more than) five extra datasets. Methods with * are not associated to a publication. The best result within the same training conditions (background color) is underlined. The best result among published methods is in bold. Best seen in color.
autonomous driving scenario with only up to 15 cars and 30 pedestrians per image but some of the sequences contain no pedestrians. For this reason, we argue that the sparse object distribution cannot fully demonstrate the capacity of TransCenter to detect and track densely crowded objects.

### 4.4 Efficiency-Accuracy Tradeoff Discussion

To have a direct idea of the better design of TransCenter, we discuss in detail the efficiency-accuracy tradeoff comparing our solution to the transformer-based concurrent works – TransTrack [64] and TrackFormer [45]. To have a fairer comparison to these methods using the (deformable) DETR encoder (D. DETR), i.e., ResNet-50 with a deformable transformer encoder, we leverage TransCenter-DETR, having the same encoder as theirs while keeping the rest of its structure unchanged. Moreover, related to our method, we show superior performance compared to the center-based MOT methods – CenterTrack [93] and FairMOT [88].

Very recently, works like ByteTrack [87] and MO3TR-PIQ [95] take advantage of the recent off-the-shelf object detector YOLOX [18] trained with extensive data-augmentation tricks such as MixUp [89], which significantly improves the MOT performance. However, we do not think that the MOT performance improvement from the object detector could be considered as their contribution. To compare with them, we use the results of the YOLOX from ByteTrack [87] as matching and birth candidates to show that TransCenter can have a similar or even better result compared to ByteTrack and MO3TR-PIQ, denoted as TransCenter-YOLOX.

The comparisons take into account the number of model parameters, the model memory footprint during inference, the inference speed (frame per second or FPS), and the MOTA performance as shown in Table 4, completing Tables 1 and 2. Additionally, we compare the center heatmap/query responses of the aforementioned methods in Section B.1 of the Supplementary Material, available online, showing that TransCenter produces sufficient and accurate outputs. All results are evaluated on both MOT17 and MOT20 testsets in the private detection setting except the comparison with ByteTrack in the public detection setting (see Pub. Det. in Table 4). All the models are pretrained on CrowdHuman [61] except for FairMOT [88] trained on 501+CH datasets, and ByteTrack additionally trained on ETHZ [14] and CityPersons [85]. The default input image size for CenterTrack [93] is 544 × 960; 608 × 1088 for FairMOT [88] and TransCenter-Lite, 640 × 1088 for TransCenter TransCenter-Dual, TransCenter-DETR and TransCenter-YOLOX.

### Table 2

Results on MOT20 Testset: The Table is Structured Following the Same Principle as Table 1

| Method | Data | MOTA | MOTP | FP | FN | IDS | FPS |
|--------|------|------|------|----|----|-----|-----|
| Mass   |      | 84.6 | 85.4 | 4,145 | 786 | 353 | 100.0 |
| IMMDP  |      | 82.8 | 82.8 | 5,300 | 422 | 211 | 5.3 |
| AB3D   |      | 83.5 | 85.2 | 4,492 | 1,060 | 126 | 214.7 |
| SMAT   |      | 83.6 | 85.9 | 5,254 | 175 | 198 | 10.0 |
| TrackMPNN [52] |      | 87.3 | 84.5 | 2,577 | 1,298 | 481 | 20.0 |
| CenterTrack [93] |      | 88.8 | 85.0 | 2,703 | 886 | 254 | 22.2 |
| TransCenter |      | 87.3 | 83.8 | 3,189 | 847 | 340 | 18.5 |
| AB3D   |      | 38.9 | 64.6 | 11,744 | 2,135 | 259 | 214.7 |
| TrackMPNN [52] |      | 52.1 | 73.4 | 7,705 | 2,758 | 626 | 20.0 |
| CenterTrack [93] |      | 53.8 | 73.7 | 8,061 | 2,201 | 425 | 22.2 |
| TransCenter |      | 59.1 | 73.2 | 6,889 | 2,142 | 436 | 18.5 |

### Table 3

KITTI Testset Results in MOTA, MOTP, FP, FN, IDS and FPS. Best Results are Underlined

| Method | MOTA | MOTP | FP | FN | IDS | FPS |
|--------|------|------|----|----|-----|-----|
| Mass   | 84.6 | 85.4 | 4,145 | 786 | 353 | 100.0 |
| IMMDP  | 82.8 | 82.8 | 5,300 | 422 | 211 | 5.3 |
| AB3D   | 83.5 | 85.2 | 4,492 | 1,060 | 126 | 214.7 |
| SMAT   | 83.6 | 85.9 | 5,254 | 175 | 198 | 10.0 |
| TrackMPNN [52] | 87.3 | 84.5 | 2,577 | 1,298 | 481 | 20.0 |
| CenterTrack [93] | 88.8 | 85.0 | 2,703 | 886 | 254 | 22.2 |
| TransCenter | 87.3 | 83.8 | 3,189 | 847 | 340 | 18.5 |
| AB3D   | 38.9 | 64.6 | 11,744 | 2,135 | 259 | 214.7 |
| TrackMPNN [52] | 52.1 | 73.4 | 7,705 | 2,758 | 626 | 20.0 |
| CenterTrack [93] | 53.8 | 73.7 | 8,061 | 2,201 | 425 | 22.2 |
| TransCenter | 59.1 | 73.2 | 6,889 | 2,142 | 436 | 18.5 |
sizes with short size of 800; ByteTrack [87] uses an input size of 800 \times 1440.

**Compared to Transformer-Based MOT.** We compare TransCenter to them both in accuracy and inference speed. Unfortunately, TrackFormer [45] by far only shows results on MOT17 and TransTrack [64] does not show results in all settings. One important remark is that TransCenter systematically outperforms TransTrack and TrackFormer in both accuracy (MOTA) and speed (FPS) with a smaller model size (# of model parameters) and less inference memory consumption in all settings. Precisely, using the same training data, TransCenter exhibits better performance compared TransTrack by +1.7% MOTA (by +11.2% versus TrackFormer) in MOT17 and significantly by +8.4% in MOT20. The PVT [72] encoder can indeed bring faster inference speed and provide more meaningful image features that boost the MOT performance. However, using the same encoder as TransTrack, TransCenter-DETR exhibits similar inference speed and much better performance in both MOT20 (+4.3%) and MOT17 (+0.3%), compared to TransTrack. This indicates that the overall design of TransCenter is more efficient and powerful. Moreover, we recall that, unlike our concurrent works, TransCenter leverages pixel-level dense and multi-scale detection queries to predict dense center-based heatmaps, mitigating the miss-tracking problem while keeping relatively good computational efficiency with sparse tracking queries, efficient QLN, and TransCenter Decoder. TransCenter demonstrates thus a significantly better efficiency-accuracy tradeoff. Finally, for different MOT applications, we provide TransCenter-Dual, introduced to further boost the performance in crowded scenes, and TransCenter-Lite for efficiency-critical applications.

**Compared to Center-Based MOT.** With the long-term dependencies and the dense interactions of queries in transformers, unlike pure CNN-based center MOT methods, our queries interact globally with each other and make global decisions. TransCenter exhibits much better accuracy compared to previous center-based MOT methods, CenterTrack and FairMOT [88], [93]. Precisely as shown in Tables 1 and 4, TransCenter tracks more and better compared to CenterTrack, with +8.4% (+10.4%) MOTA and -71,505 (-73,812) FN in MOT17 private (public) detections, using the same dense center representations and trained with the same data. CenterTrack does not show results in MOT20 while FairMOT, similar to CenterTrack, shows good results in MOT20. It alleviates miss detections by training with much more data, leading to much fewer FN but producing much noisier detections (FP). Surprisingly, with much less training data (CH), TransCenterstill outperforms [88] by a large margin (+11.1% MOTA shown in Table 4) even in very crowded MOT20, with cleaner detections (suppressing -74,844 FP) and

| Model | Enc/Det | #params (M) ↓ | IM (MB) ↓ | FPS ↑ | MOTA ↑ |
|-------|---------|--------------|----------|-------|-------|
| FairMOT [88] | DLA-34 | 20.3 | 892 | 25.9 | 73.7 |
| CenterTrack [93] | D. DETR | 20.0 | 892 | 17.5 | 67.8 |
| TrackFormer [45] | 40.0 | 976 | 6.8 | 65.0 |
| TransTrack [64] | 47.1 | 1,002 | 10.0 | 74.5 |
| TransCenter-DETR | 43.6 | 980 | 8.8 | 74.8 |
| ByteTrack [87] | YOLOX | 99.1 | 1,746 | 29.6 | 80.3 |
| MO3TR-PIQ [95] | 39.7 | 938 | 11.9 | 79.8 |
| TransCenter-YOLOX | 35.1 | 938 | 11.8 | 76.2 |
| TransCenter-Dual | 35.1 | 938 | 11.8 | 76.2 |
| TransCenter-Lite | 35.1 | 938 | 11.7 | 75.9 |
| ByteTrack | Pub. Det. | 99.1 | 1,746 | N/A | 67.4 |
| TransCenter | 35.1 | 938 | 11.7 | 75.9 |
| FairMOT [88] | DLA-34 | 20.3 | 892 | 13.2 | 61.8 |
| CenterTrack [93] | N/A | N/A | N/A | N/A |
| TrackFormer [45] | N/A | N/A | N/A | N/A |
| TransTrack [64] | 47.1 | 1,002 | 7.2 | 64.5 |
| TransCenter-DETR | 43.6 | 980 | 7.5 | 68.8 |
| ByteTrack [87] | YOLOX | 99.1 | 1,746 | 17.5 | 77.8 |
| MO3TR-PIQ [95] | N/A | N/A | N/A | N/A |
| TransCenter-YOLOX | 35.1 | 938 | 10.1 | 77.9 |
| TransCenter-Dual | 35.1 | 938 | 5.1 | 73.5 |
| TransCenter | 35.1 | 938 | 8.7 | 72.9 |
| TransCenter-Lite | 8.1 | 838 | 11.0 | 68.0 |
| ByteTrack | Pub. Det. | 99.1 | 1,746 | N/A | 67.0 |
| TransCenter | 35.1 | 938 | 8.4 | 72.8 |

- **Best results are underlined.** They are grouped according to their encoders or used detections, indicated as Enc/Det. Without extra notation, private detections are used by default.

7. MO3TR-PIQ [95] is not yet published and does not provide source code nor sufficient information to evaluate the IM, #params, and the FPS.

---

**TABLE 4**

Testset Comparisons on the MOT17 and MOT20 Among CenterTrack [93], FairMOT [88], Trackformer [45], TransTrack [64], ByteTrack [87], MO3TR-PIQ [95] and our Proposed Models in Number of Model Parameters (#Params), Inference Memory (IM), Frame Per Second (FPS) and MOTA
better tracking associations (-2,618 IDS) shown in Table 2. Indeed, the inference speed is slower compared to CNN-based methods, but the above comparisons have demonstrated that previous center-based MOT methods are not comparable in terms of accuracy to TransCenter, with an acceptable fps around 11 fps for MOT17 and 8 fps for MOT20. Moreover, to adapt to applications with more strict inference constraints, TransCenter-Lite is introduced, keeping a better performance while having competitive inference speed compared to center-based MOT methods.

Compared to YOLOX-Based MOT. The recent works, ByteTrack [87] and MO3TR-PIQ show significant performance gain by using the off-the-shelf object detector YOLOX [18]. We believe that the gain is mainly from the object detector, which is not the contribution of any aforementioned method. This claim is further confirmed by the direct comparison of TransCenter and ByteTrack in public detection, i.e., without YOLOX, where TransCenter outperforms the latter with 75.9% versus 67.4% MOTA in MOT17 and 72.8% versus 67.0% in MOT20 (see Table 4).

For private detection, to have a fair comparison with them, we use the detection results from YOLOX (same as ByteTrack) to demonstrate that YOLOX can also significantly boost TransCenter in terms of MOT. We agree that indeed there is an inference speed discrepancy with CNN-based methods compared to our transformer-based TransCenter. However, in terms of accuracy, TransCenter-YOLOX shows comparable or even better results, compared to all other methods. This indicates that the design of TransCenter is beneficial with different encoders/object detectors.

To conclude, TransCenter expresses both better accuracy and efficiency compared to transformer-based methods [45, 64]; much higher accuracy numbers and competitive efficiency compared to [88], [93], showing better efficiency-accuracy balance. Moreover, with the powerful YOLOX detector, TransCenter can be significantly improved in terms of MOT performance, showing the potential of TransCenter.

### 4.5 Ablation Study

In this section, we first experimentally demonstrate the importance of our proposed image-related dense queries with naive DETR to MOT approaches. Then, we justify the effectiveness of our choices of QLN and TransCenter-Decoders (see illustrations in Figs. 3 and 4, respectively), considering the computational efficiency and accuracy. All results are shown in Table 5. Furthermore, in Table 6, we ablate the impacts of removing the external Re-ID network and the NMS (Non-Maximum Suppression) during inference. Finally, we show an additional ablation of the number of decoder layers in Section A of the Supplementary Material.
Material, available online. For the ablation, we divide the training sets into a train-validation split, we take the first 50% of frames as training data and test on the last 25%. The rest 25% of frames in the middle of the sequences are thrown to prevent over-fitting. All the models are pre-trained on CrowdHuman [61] and tested under the private detection setting.

Dense Representations Are Beneficial. We implemented a naive DETR MOT tracker with its original 100 sparse queries (from learnable embeddings initialized from noise) with the DETR-Encoder and Dual TransCenter Decoder (Line 1 in Table 5). To compare, the same tracker but having dense representations (43,520, i.e., $H/4 \times W/4$) is shown in Line 2. From the results shown in Line 1-2 of Table 5, we see that the limited number of queries (100, by default in [96]) is problematic because it is insufficient for detecting and tracking objects, especially in very crowded scenes MOT20 (+106,467 FN, -24.8% MOTA, compared to Line 2). This indicates that having dense representations is beneficial, especially for handling crowded scenes in MOT. Some visualization examples are shown in Section 4.6.

Naive Dense, Noisy Dense versus Image-Related Dense. One naive way to alleviate the insufficient queries is to greatly increase the number of queries like in Line 2 of Table 5: we drastically increase the number of queries from 100 to 43,520 (i.e., $H/4 \times W/4$) as the dense output of TransCenter. Meanwhile, we compare this naive dense queries implementation to a similar one in Line 3 but with image-related dense queries like in TransCenter. Concretely, we obtain the dense queries from QLN$_{DQ}$ (as described in Fig. 3c) with the memories output from DETR-Encoder. We note that the image-related implementation has indeed 43,450 queries (i.e., the sum of the 1/8, 1/16, 1/32, and 1/64 of the image size), and the up-scale and merge operation (see Fig. 5) forms a dense output having the same number of pixels of 43,520. Therefore, we ensure that the supervisions for the losses are equivalent for both Line 2 and 3.

The main difference between Line 2 and 3 is the queries: the proposed multi-scale dense detection queries are related to the input image where one query represents one pixel. The benefit of image-related pixel-level queries is well-discussed in Section 3. From the experimental aspect in Table 5, for the noise-initialized queries in Line 2, despite the manual one-to-one matching during training, increasing the queries naively and drastically tends to predict noisier detections causing much higher FP (compared to Line 3, +2,308 and +37,850 in MOT17 and MOT20, respectively) and thus much under-performed tracking results (-12.6% and -12.1% MOTA for MOT17 and MOT20, respectively). Although more sophisticated implementations using sparse noise-initialized queries and Hungarian matching loss like in [64] and [45] exhibit improved results, TransCenter shows both better accuracy and efficiency, as discussed in Section 4.4.

Furthermore, we combine TransCenter(-Dual) with QLN$_{SN}$ (see Fig. 3a in red) where the detection queries are of image size as in QLN$_{SN}$ but noise-initialized, denoted as Noise Dense. We note that since they are noise-initialized (i.e., unrelated to the image), it makes no sense to discard the detection attention – DDCA (see Section 3.2) from TransCenter Decoder. The Noisy Dense detection queries thus need to correlate with $M_t$ to extract information from the image $t$ in TransCenter Decoder and cannot speed up by removing DDCA like in TransCenter. The result with such design indicated in Line 18 is thus compared with TransCenter-Dual in line 17, having the same structure but with the chosen QLN$_{SN}$. We observe that the Noise Dense queries produce a decent result thanks to the rest of TransCenter designs like image-related sparse queries, TransCenter Decoder, and the use of PVT encoder but they tend to have more FP caused by the noise initialization for detection queries, leading to the overall worse performance compared to TransCenter-Dual in MOT17, IDF1, and IDS.

QLN Inputs. A QLN generates tracking queries $TQ$ and memories $TM$; detection queries $DQ$ and memories $DM$. Based on the nature of tracking, we argue that $TQ$ and $TM$ should be obtained from information at different time steps since tracking means associating object positions in the adjacent frames. This creates two variants of QLN, namely QLN$_M$ with $TQ$ from $M_t$ and $TM$ are from $M_{t-1}$ (Line 4) and inversely, QLN$_D$, where $TQ$ are from $M_{t-1}$ and $TM$ are from $M_t$ (Line 5). From their results, we do not observe a significant difference in terms of performance, indicating both implementations are feasible. Further, we experimentally compare QLN$_M$ (Line 4) and QLN$_D$ (Line 6), where the only difference is the number of FFN for outputting $TQ$. With an extra FFN for $TQ$, the performance is slightly improved (+2.4% for MOT17 and +1.1% for MOT20).

Efficient TransCenterDecoder. As we discuss in Section 3.2, the design of TransCenter Decoder can have an important

| Setting | MOT17 | MOT20 |
|---------|-------|-------|
| MOTA | IDF1 | FP | FN | IDS | FPS | MOTA | IDF1 | FP | FN | IDS | FPS |
| + ex-ReID | 73.7 | 73.5 | 1,047 | 5,800 | 286 | 4.5 | 83.6 | 75.5 | 15,468 | 34,057 | 1,084 | 2.9 |
| + NMS | 73.8 | 74.1 | 1,300 | 5,548 | 261 | 11.0 | 83.6 | 75.8 | 15,458 | 34,053 | 1,084 | 5.1 |
| TransCenter | 73.8 | 74.1 | 1,302 | 5,540 | 258 | 12.4 | 83.6 | 75.7 | 15,459 | 34,054 | 1,085 | 8.9 |
| + ex-ReID | 69.8 | 72.0 | 2,008 | 5,922 | 248 | 5.2 | 82.9 | 74.9 | 14,894 | 36,498 | 1,147 | 3.1 |
| + NMS | 69.8 | 71.6 | 2,009 | 5,923 | 253 | 15.9 | 82.9 | 75.4 | 14,872 | 36,507 | 1,181 | 6.2 |
| TransCenter-Lite | 69.8 | 71.6 | 2,008 | 5,923 | 252 | 19.4 | 82.9 | 75.5 | 14,857 | 36,510 | 1,181 | 12.4 |
| + ex-ReID | 74.6 | 73.8 | 840 | 5,882 | 162 | 3.0 | 84.5 | 77.7 | 13,478 | 33,209 | 921 | 3.4 |
| + NMS | 74.6 | 76.5 | 891 | 5,879 | 127 | 5.1 | 84.6 | 77.9 | 13,413 | 33,203 | 921 | 3.4 |
| TransCenter-Dual | 74.6 | 76.5 | 892 | 5,879 | 128 | 5.6 | 84.6 | 78.0 | 13,415 | 33,202 | 921 | 4.8 |

---

They are evaluated on both MOT17 and MOT20 validation sets using TransCenter, TransCenter-Lite and TransCenter-Dual, respectively.

Table 6. Ablation Study of External Inference Overheads: The Results Are Expressed in MOTA, MOTP, FP, FN, IDS as Well as FPS.
impact on the computational efficiency and accuracy with different variants (Line 11-16). Precisely, comparing Line 11 and 12, we observe indeed that, with dual-decoder handling cross-attention for both detection and tracking, the performance is superior (+0.8% MOTA for MOT17 and +1.0% for MOT20) but the inference is slowed down by around 50%. Balancing the efficiency and accuracy, we argue that TransCenter (Line 12) is a better choice. Moreover, by removing the TQSA module (Line 13, 14), we obtain a slight inference speed up (+0.9 fps for MOT17 and MOT20) but at the cost of accuracy (-4.4% MOTA in MOT17 and -0.6% in MOT20). Finally, we also study the effect of sparse and dense tracking (Line 15-16), surprisingly, we find that using sparse tracking can help better associate object positions between frames (-20 IDS in MOT17 and -131 IDS in MOT20) and in a more efficient way (+3.4 fps in MOT17 and +2.6 in MOT20).

Efficient PVT-Encoder. Passing from DETR-Encoder (Line 7) to PVT-Encoder (Line 8) helps get rid of the ResNet-50 feature extractor, which partially speeds up the inference from 2.1 fps to 8.1 in MOT17, and 1.2 to 6.1 fps in MOT20. Moreover, the PVT-Encoder exhibits better results which may be due to the lighter structure that eases the training (+4.2% MOTA in MOT17 and +3.8% in MOT20). Similarly, with PVT-Lite, we can speed up +7 fps for MOT17 and +3.5 fps for MOT20, comparing Line 9 and 10 while keeping a competitive performance.

External Inference Overheads. MOT methods like [4] use an external Re-ID network to extract identity features so that we can recover the objects which are temporally suspended by the tracker through appearance similarities. The Re-ID network (paired with a light-weight optical flow estimation network LiteFlowNet [27] pre-trained on KITTI [19]) is often a ResNet-50 [4], pre-trained on object identities in MOT17 trainset. From Table 6, we observe that this external Re-ID does help reduce IDS, especially in crowded scenes but it slows down significantly the inference. To speed up, we replace the external Re-ID features extractor simply by sampled features from memories \(M_t\), with a feature sampler like in QLN\(_S\) (Fig. 3a) using positions from detections or tracks at \(t\), which is almost costless in terms of calculation and achieves comparable results to external Re-ID.

One of the benefits of using our image-related dense representations is the discard of the one-to-one assignment process during training. Intuitively, no NMS is needed during inference. However, recall from Eq. (2) that the heatmaps are Gaussian distributions centered at the object centers, a 3x3 max-pooling operation is somehow needed to select the maximum response from each distribution (i.e., object center). In practice, an NMS is only performed as in [4] within tracked objects. However, from the results shown in Table 6, the NMS operation between tracks has little impact on the accuracy but imports important overheads. For this reason, such NMS is discarded from TransCenter, TransCenter-Lite, and TransCenter-Dual.

In summary, TransCenter can efficiently perform MOT with sparse tracking queries and dense detection queries operating on the proposed QLN and TransCenter Decoder structures, leveraging the PVT-Encoder. The accuracy is
further enhanced with the finer multi-scale features from the PVT-Encoder, the sparsity of the tracking queries as well as the chosen designs of the QLN and TransCenter Decoder. Therefore, TransCenter shows a better efficiency-accuracy tradeoff compared to naive approaches and existing works.

4.6 Qualitative Visualizations in Crowded Scenes
We report in Fig. 6 the qualitative results evaluated on some crowded MOT20 sequences, to demonstrate the detection and tracking abilities of TransCenter in the context of crowded scenes. Concretely, we show in Fig. 6 the predicted center trajectories and the corresponding object sizes. Fig. 6a is extracted from MOT20-04, Fig. 6b from MOT20-07 and Fig. 6c from MOT20-06. We observe that TransCenter manages to keep high recall, even in the context of drastic mutual occlusions, and reliably associate detections across time.

5 Conclusion
In this paper, we introduce TransCenter, a powerful and efficient transformer-based architecture with dense image-related representations for multiple-object tracking. TransCenter explores dense pixel-level multi-scale detection queries and sparse tracking queries, produced with carefully-designed QLN and interacting in TransCenter Decoder with image features. Under the same training conditions, TransCenter outperforms its competitors in MOT17 and by a large margin in very crowded scenes in MOT20. It even exhibits better performance than some methods trained with much more data. More importantly, TransCenter maintains a reasonably high efficiency while exhibiting a state-of-the-art performance, thanks to its careful designs proven by a thorough ablation.

Current TransCenter explores the temporal information between only two frames. However, the long-term temporal information is crucial for identity consistency especially in situations when the objects are occluded for several frames, which is not well addressed in the work. Therefore, one of the future directions for making TransCenter more robust related to long-term MOT would be exploring multiple frames as input signals and conceiving an effective way of leveraging such information. However, this is not a trivial task since the model efficiency problem will be again challenged. Moreover, the long-term identity consistency is still an unsolved topic in the MOT community.

Acknowledgments
Yihong Xu and Yutong Ban contributed equally to this work.

References
[1] N. L. Baisa, “Online multi-object visual tracking using a GM-PHD filter with deep appearance learning,” in Proc. IEEE 22nd Int. Conf. Inf. Fusion, 2019, pp. 1–8.
[2] Y. Ban, X. Alameda-Pineda, L. Girin, and R. Horard, “Variational Bayesian inference for audio-visual tracking of multiple speakers,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 43, no. 5, pp. 1761–1776, May 2021.
[3] Y. Ban, S. Ba, X. Alameda-Pineda, and R. Horard, “Tracking multiple persons based on a variational Bayesian model,” in Proc. Eur. Conf. Comput. Vis., 2016, pp. 52–67.
[4] P. Bergmann, T. Meinhardt, and L. Leal-Taixe, “Tracking without bells and whistles,” in Proc. IEEE/CVF Int. Conf. Comput. Vis., 2019, pp. 941–951.
[5] K. Bernardin and R. Stiefelhagen, “Evaluating multiple object tracking performance: The CLEAR metricts,” EURASIP J. Image Video Process., vol. 2008, pp. 1–10, 2008.
[6] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, “Simple online and realtime tracking,” in Proc. IEEE Int. Conf. Image Process., 2016, pp. 3464–3468.
[7] G. Braso and L. Leal-Taixe, “Learning a neural solver for multiple object tracking,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2020, pp. 5247–5257.
[8] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, “End-to-end object detection with transformers,” in Proc. Eur. Conf. Comput. Vis., 2020, pp. 213–229.
[9] L. Chen, H. Ai, Z. Zhuang, and C. Shang, “Real-time multiple people tracking with deeply learned candidate selection and person re-identification,” in Proc. IEEE Int. Conf. Multimedia Expo, 2018, pp. 1–6.
[10] J. Dai et al., “Deformable convolutional networks,” in Proc. Int. Conf. Comput. Vis., 2017, pp. 764–773.
[11] P. Dendorfer et al., “MOT20: A benchmark for multi object tracking in crowded scenes,” 2020, arXiv:2003.09003.
[12] P. Dollár, C. Wojek, B. Schiele, and P. Perona, “Pedestrian detection: An evaluation of the state of the art,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 34, no. 4, pp. 743–761, Apr. 2012.
[13] A. Dosovitskiy et al., “An image is worth 16x16 words: Transformers for image recognition at scale,” 2020, arXiv:2010.11929.
[14] A. Ess, B. Leibe, K. Schindler, and L. van Gool, “A mobile vision system for robust multi-person tracking,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2008, pp. 1–8.
[15] C. Gan, D. Huang, P. Chen, J. B. Tenenbaum, and A. Torralba, “Foley music: Learning to generate music from videos,” in Proc. Eur. Conf. Comput. Vis., 2020, pp. 758–777.
[16] C. Gan, D. Huang, H. Zhao, J. B. Tenenbaum, and A. Torralba, “Music gesture for visual sound separation,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2020, pp. 10478–10487.
[17] C. Gan, H. Zhao, P. Chen, D. Cox, and A. Torralba, “Self-supervised moving vehicle tracking with stereo sound,” in Proc. IEEE/CVF Int. Conf. Comput. Vis., 2019, pp. 7053–7062.
[18] Z. Ge, S. Liu, F. Wang, Z. Li, and J. Sun, “YOLOX: Exceeding yolo series in 2021,” 2021, arXiv:2107.08430.
[19] A. Geiger, P. Lenz, and R. Urtasun, “Are we ready for autonomous driving? the kitti vision benchmark suite,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2012, pp. 3354–3361.
[20] N. F. Gonzalez, A. Ospina, and P. Calvez, “Smat: Smart multiple object tracking,” 2019, pp. 1–8.
[21] S. Guo, J. Wang, X. Wang, and D. Tao, “Online multiple object tracking with cross-task synergy,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2021, pp. 8136–8145.
[22] S. Han et al., “Mat: Motion-aware multi-object tracking,” 2020, arXiv:2009.04794.
[23] J. He, Z. Huang, N. Wang, and Z. Zhang, “Learnable graph matching: Incorporating graph partitioning with deep feature learning for multiple object tracking,” in Proc. Int. Conf. Comput. Vis. Pattern Recognit., 2021, pp. 5299–5309.
[24] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2016, pp. 770–778.
[25] S. He, H. Luo, P. Wang, F. Wang, H. Li, and W. Jiang, “TransReID: Transformer-based object re-identification,” 2021, arXiv:2102.04578.
[26] A. Hornakova, R. Henschel, B. Rosenhahn, and P. Swoboda, “Lifted disjoint paths with application in multiple object tracking,” in Proc. Inf. Conf. Mach. Learn., 2020, pp. 4364–4375.
[27] T.-W. Hu, X. Tang, and C. C. Loy, “LiteFlowNet: A lightweight convolutional neural network for optical flow estimation,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2018, pp. 8981–8989.
[28] D. Jia, D. Wei, S. Richard, L.-J. Le, L. Kai, and F.-F. Li, “ImageNet: A large-scale hierarchical image database,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2009, pp. 248–255.
[29] Y. Jiang, S. Chang, and Z. Wang, “TransGAN: Two transformers can make one strong GAN,” 2021, arXiv:2102.07074.
[30] S. Karthik, A. Prabhu, and V. Gandhi, “Simple unsupervised multi-object tracking,” 2020, arXiv:2006.02609.
[85] S. Zhang, R. Benenson, and B. Schiele, “Citypersons: A diverse dataset for pedestrian detection,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2017, pp. 3213–3221.

[86] Y. Zhang, H. Sheng, Y. Wu, S. Wang, W. Ke, and Z. Xiong, “Multiplex labeling graph for near-online tracking in crowded scenes,” IEEE Internet Things J., vol. 7, no. 9, pp. 7892–7902, Sep. 2020.

[87] Y. Zhang et al., “ByteTrack: Multi-object tracking by associating every detection box,” 2021, arXiv:2110.06864.

[88] Y. Zhang, C. Wang, X. Wang, W. Zeng, and W. Liu, “FairMOT: On the fairness of detection and re-identification in multiple object tracking,” 2020, arXiv:2004.01888.

[89] Z. Zhang, T. He, H. Zhang, Z. Zhang, J. Xie, and M. Li, “Bag of freebies for training object detection neural networks,” 2019, arXiv:1902.04103.

[90] L. Zheng, L. Shen, L. Tian, S. Wang, J. Wang, and Q. Tian, “Scalable person re-identification: A benchmark,” in Proc. IEEE Int. Conf. Comput. Vis, 2015, pp. 1116–1124.

[91] L. Zheng, H. Zhang, S. Sun, M. Chandraker, Y. Yang, and Q. Tian, “Person re-identification in the wild,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2017, pp. 1367–1376.

[92] L. Zheng, M. Tang, Y. Chen, G. Zhu, J. Wang, and H. Lu, “Improving multiple object tracking with single object tracking,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2021, pp. 2453–2462.

[93] X. Zhou, V. Koltun, and P. Krähenbühl, “Tracking objects as points,” in Proc. Eur. Conf. Comput. Vis., 2020, pp. 474–490.

[94] X. Zhou, D. Wang, and P. Krähenbühl, “Objects as points,” 2019, arXiv:1904.07850.

[95] T. Zhu, M. Hiller, M. Ehsanpour, R. Ma, T. Drummond, and H. Rezatofighi, “Looking beyond two frames: End-to-end multi-object tracking using spatial and temporal transformers,” 2021, arXiv:2103.14829.

[96] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai, “Deformable DETR: Deformable transformers for end-to-end object detection,” 2020, arXiv:2010.04159.

Yihong Xu received the master’s degree from IMT Atlantique (previously Télécom Bretagne) and the PhD degree from RobotLearn team with Inria Grenoble Rhône-Alpes, in June 2022. He is currently a research scientist with Valeo ai in Paris. His research interests lie in multiple-object tracking, domain adaptation as well as autonomous driving applications based on machine learning.

Yutong Ban received the BSc degree in telecommunications from Xidian University, China, the engineering degree in image processing and computer science from Telecom Saint-Etienne, France, and the PhD degree with PERCEPTION team with INRIA Grenoble Rhône-Alpes and University Grenoble Alpes. He is currently a postdoctoral research fellow with both Distributed Robotics Lab, CSAIL, MIT and Surgical Artificial Intelligence Innovation Lab, MGH. His research interests include surgical video analysis, audio-visual cross-modal learning, and multi-object tracking.

Guillaume Delorme received the PhD degree from RobotLearn team with Inria Grenoble Rhône-Alpes, in 2021 and the engineering degree in applied mathematics and computer science from the Ensimag Institut Polytechnique de Grenoble, in 2017. He is especially interested in deep learning, appearance modeling, person re-identification, and visual tracking.

Chuang Gan is a principal research staff member with MIT-IBM Watson AI Lab. He is also an affiliated researcher with MIT EECS. His research interests focus on computer vision and machine learning. His research works have been recognized by Microsoft Fellowship, Baidu Fellowship, and media coverage from CNN, BBC, The New York Times, WIRED, Forbes, and MIT Tech Review. He has also served as an area chair of CVPR, ICCV, AICL, ICLR, and an associate editor of IEEE Transactions on Image Processing and IEEE Transactions on Circuits and Systems for Video Technology.

Daniela Rus (Fellow, IEEE) received the PhD degree in computer science from Cornell University, Ithaca, NY, USA, in 1993. She is the Andrew (1956) and Erna Viterbi professor of Electrical Engineering and Computer Science and the Director of the Computer Science and Artificial Intelligence Laboratory with Massachusetts Institute of Technology (MIT), Cambridge, MA, USA. Prior to joining MIT, she was a professor with the Department of Computer Science, Dartmouth College, Hanover, NH, USA. Her research interests include robotics, mobile computing, and Big Data. The key focus of her research is to develop the science of networked/distributed/collaborative robotics. She is a Class of 2002 MacArthur Fellow, a Fellow of ACM and AAAI, and a member of the National Academy of Engineering.

Xavier Alameda-Pineda (Senior Member, IEEE) received the MSc degree (equivalent) in mathematics, in 2008, in Telecommunications, in 2009 from BarcelonaTech, and in Computer Science from Université Grenoble-Alpes (UGA), in 2010. He is a tenured Research Scientist with Inria, and the Leader of the RobotLearn Team. He is currently working toward the PhD degree in mathematics and computer science, and obtained it in 2013, from UGA. After a two-year post-doc period at the Multimodal Human Understanding Group, with the University of Trento, he was appointed with his current position. Xavier is an active member of SIGMM, and a member of ELLIS. He is the Coordinator of the H2020 Project SPRING: Socially Pertinent Robots in Gerontological Healthcare and is co-leading the “Audio-visual machine perception and interaction for companion robots” chair of the Multidisciplinary Institute of Artificial Intelligence. Xavier’s research interests are in combining machine learning, computer vision, and audio processing for scene and behavior analysis and human-robot interaction.

For more information on this or any other computing topic, please visit our Digital Library at www.computer.org/csdl.