6D-VNet: End-to-end 6DoF Vehicle Pose Estimation from Monocular RGB Images

Di Wu, Zhaoyong Zhuang, Canqun Xiang, Wenbin Zou and Xia Li
The Guangdong Key Laboratory of Intelligent Information Processing
College of Information Engineering, Shenzhen University

Abstract

We present a conceptually simple framework for 6DoF object pose estimation, especially for autonomous driving scenario. Our approach efficiently detects traffic participants in a monocular RGB image while simultaneously regressing their 3D translation and rotation vectors. The method, called 6D-VNet, extends Mask R-CNN by adding customised heads for predicting vehicle's finer class, rotation and translation. The proposed 6D-VNet is trained end-to-end compared to previous methods. Furthermore, we show that the inclusion of translational regression in the joint losses is crucial for the 6DoF pose estimation task, where object translation distance along longitudinal axis varies significantly, e.g.*, in autonomous driving scenarios. Additionally, we incorporate the mutual information between traffic participants via a modified non-local block. As opposed to the original non-local block implementation, the proposed weighting modification takes the spatial neighbouring information into consideration whilst counteracting the effect of extreme gradient values. Our 6D-VNet reaches the 1 st place in ApolloScape challenge 3D Car Instance task*[1] *[21]. Code has been made available at: https://github.com/stevenwudi/6DVNET.*

1. Introduction

For self-driving cars, it is important to detect surrounding vehicles, pedestrians, riders, *etc*. The system must understand the 3D relationships between traffic participants in its field of view. One of the crucial components is to detect, estimate and reconstruct the 3D shape of vehicles in a given video (Fig. 1).

Current state-of-the-art RGB-based 6DoF pose estimation systems [32, 6, 22] are two-staged: the first stage is to detect the object with 3D rotation via a trained network, and the second stage is to estimate the full 3D translation via projective distance estimation. The aforementioned two-staged systems primarily focus on the industry-relevant bin-picking tasks. Typically, a robot needs to grasp a single arbitrary instance of the required object, *e.g.*, a component such as a bolt or nut, and operate with it. In such scenario, the surface alignment in the Z dimension, *i.e.*, the optical axis of the camera, is less important than the alignment in the X and Y dimensions. Such industrial setting requires accurate estimation of rotation, whereas the translation tolerance can be relaxed. However, in autonomous driving, translation distance of traffic participants along longitudinal axis varies significantly. Consequently, the translation estimation is more challenging. In the meanwhile, the estimation of vehicle's translation is more critical than that of orientation.

Figure 1: 6D-VNet is trained end-to-end to estimate vehicles' six degree of freedom poses from a single monocular image. The output will precisely estimates the vehicle's voxels in 3D occupancy.

Traditional methods leave the translational estimation as a separate procedure after the object class prediction and rotation estimation by using a geometric projection method. However, the geometric projection method assumes that: (i) the object centre in 3D will be projected to the object bounding box centre in the 2D image; (ii) the predicted object class and rotation vector is correctly estimated. Therefore,

[1] http://apolloscape.auto/car_instance.html

by using geometric projection as the post-processing step, the error from object class estimation and rotation regression will be aggregated in the following projective distance estimation.

To accommodate the requirement for accurate translation estimation in autonomous driving, we propose a framework, called 6D-VNet, aiming at regressing the vehicle's rotation and translation simultaneously (Fig. 2). 6D-VNet streamlines the vehicle's 6DoF via the intermediate outputs from the Region Proposal Network (RPN) [10]. The detection part of the network is the canonical 2D object detection network (Mask R-CNN). The 6DoF estimation part of the network takes the intermediate output from the detection head. The challenging aspect of learning 6DoF vehicle pose is to design a loss function which is able to learn both rotation and translation. The model learns a complementary representation when supervised by both translation and orientation signals. Moreover, traffic participants exert mutual influence among their neighbours. Therefore, we introduce a weighted non-local block, which is a modified version of [41], to capture the collective information between traffic participants with interpretable self-attention map.

Specifically, the network is trained end-to-end by joint losses designed with solid geometric ground. Experimental results show that the proposed method outperforms the state-of-the-art two-staged systems. We list our contributions as follows:

- To our best knowledge, this is the first work which successfully regresses the rotation and translation simultaneously for deep learning-based object 6DoF pose estimation. And we showcase the effectiveness of the translation head inclusion into end-to-end training scheme (Sec. 3.1).
- With a grounding in geometry, we investigate several joint losses that function synergistically (Sec. 3.2).
- We capture the densely spatial dependencies by introducing a weighted non-local operation with interpretable self-attention map (Sec. 3.3).

2. Related Work

Monocular-based 3D object detection were helped by early work on face detection [39] to popularise bounding box object detection. Later, pedestrian detection [7], PASCAL VOC [8], MS-COCO [29] pushed the detection towards a more diverse, challenging task. Detectron [11] is based on a line of works [27, 14, 41, 12, 28, 20, 31, 13, 43] to enable extensive research projects based on 2D object detection. KITTI dataset [9] propelled the research for traffic participants under autonomous driving scenario. However, the 3D object detection task in the KITTI dataset primarily focuses on using point clouds data from Velodyne laser scanner, which is an expensive apparatus. In addition, the KITTI 3D object detection task only has half degree of freedom for rotation overlooking vehicle's heading direction.

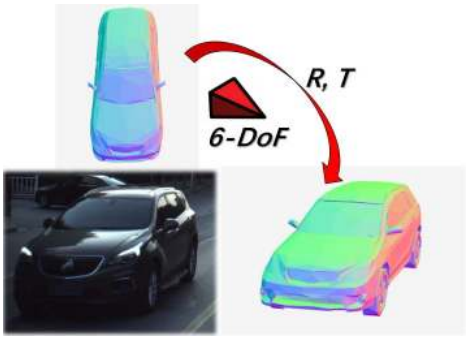

Figure 2: The core component of the proposed network: following the detection head in 2D space, the 6D-VNet regresses the 6DoF rotation and translation vector simultaneously in 3D space.

Camera pose estimation is the problem of localisation, that is, to infer where you are, and is crucial for mobile robotics, navigation and augmented reality. PoseNet [25] trains a convolutional neural network to regress the 6DoF camera pose from a single RGB image in an end-to-end manner with no need of additional engineering or graph optimisation. A more fundamental theoretical treatment is given in [24] by exploring a number of loss functions based on geometry and scene reprojection error. [40] proposes a unified framework to tackle self-localisation and camera pose estimation simultaneously. The problem of camera pose estimation is egocentric, in other words, a single vector of 6 dimensions will suffice to relocalise the camera pose.

6DoF object detection is essential for mobile robotic manipulation and augmented reality. The BOP benchmark [18] comprises of eight datasets in a unified format that cover different practical scenarios and shows that methods based on point-pair features [38] currently outperform methods based on template matching [19], learning-based [2, 3, 33, 23] and 3D local features [5]. There are very encouraging recent results [22, 17, 42, 38, 15, 32, 6, 34] using either RGB or RGB-D images for detecting 3D model instances and estimating their 6DoF poses. A two-staged 6DoF object detection pipeline is proposed in [32]: firstly, a Single Shot Multibox Detector (SSD) [30] is applied to provide object bounding boxes and identifiers. Then an Augmented Autoencoder (AAE) is applied to estimate the object rotation using a Domain Randomisation [35] Strategy. However, the aforementioned methods focused on the industry-relevant objects with neither significant texture nor discriminative colour or reflectance properties. Moreover, objects of interests are lying on a uniform ground plane (*e.g.*, in T-LESS dataset [17], the range of object distance is from 650mm to 940mm). Hence, the tolerance of rotation needs to stay low, whereas translation can be relaxed.

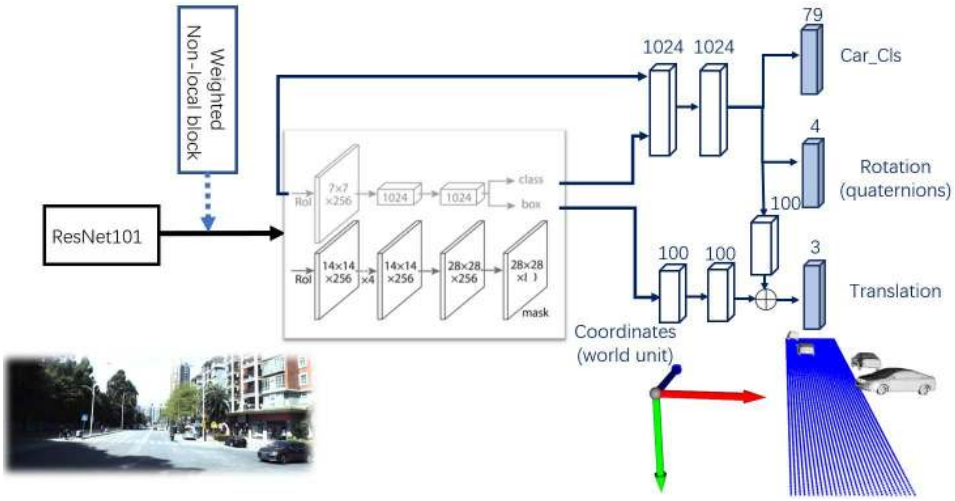

Figure 3: System pipeline: 6D-VNet takes a monocular image as input and performs the vehicles' 6DoF estimation. The grey box represents a canonical instance segmentation network and the dark blue branch is for estimating object 6DoF pose and its sub-category.

3. Model

6D-VNet is conceptually intuitive and structurally hereditary: Faster R-CNN has two outputs for each candidate object, a class label and a bounding-box offset, Mask R-CNN adds a third branch that outputs the object mask. Likewise, 6D-VNet streamlines the objects 6DoF prediction via the intermediate outputs from the Region Proposal Network (RPN). However, in order not to break already learnt functionality of the pre-trained network, careful design choices must be customised for the end-to-end training. Next, we present the overall architecture in Sec. 3.1. Particularly, we introduce the end-to-end training paradigm with translation estimation integration which greatly outperforms other two staged frameworks in terms of translation estimation accuracy. The design choices for the joint losses are presented in Sec. 3.2. Lastly we show the spatial relationship between traffic participants can be incorporated via a modified weighted non-local block in Sec. 3.3.

3.1. Network Architecture

6D-VNet is built upon the canonical object detection network as shown in Fig. 3. The system is a two-staged network trained end-to-end to estimate the 6DoF pose information for object of interest. The first stage of the network is a typical 2D object detection network (Mask R-CNN). The second stage of the network is the customised heads to estimate the object 6DoF pose information.

The 6DoF pose estimation branch is the novelty of the model and is split into two parts: the first part only takes RoIAlign [14] from each candidate box if the candidate is of the vehicle class and performs sub-class categorisation and rotation estimation. Due to in-plane rotation is unique for a given vehicle class, all vehicles share similar rotational features for the same yaw, pitch, and roll angles. Therefore, the fixed-size visual cue from RoIAlign layer is sufficient for estimating the candidate sub-category and rotation.

The second part takes both RoIAlign feature and bounding box information (in world unit as described in Sec. 3.2) via a concatenation operation to estimate the 3 dimensional translational vector. To our knowledge, this novel formulation is the first of its kind to regress the translational vector directly. The joint feature combination scheme implicitly encodes the object class and rotation information via the concatenation operation ($\oplus$ in Fig. 3). The translation regression head functions in synergy when it is combined with the joint loss from sub-category classification and quaternion regression. We show in the experiment that our novel formulation for translation regression produces much more accurate position estimation comparing to the methods that treat the translation estimation as a post-processing step. This accurate estimation of translational vector is particularly crucial for the applications where the distance of the objects are of primary importance (*e.g.*, in the autonomous driving scenario).

3.2. Joint Losses

We minimise the following loss $\mathcal{L}$ to train our network in an end-to-end fashion: $\mathcal{L} = \mathcal{L}_{det} + \mathcal{L}_{inst}$, where $\mathcal{L}_{det}$ denotes the multi-task loss as in a canonical detection network: $\mathcal{L}_{det} = \mathcal{L}_{cls} + \mathcal{L}_{box} + \mathcal{L}_{mask}$. The classification loss $\mathcal{L}_{cls}$, 2D bounding box loss $\mathcal{L}_{box}$ and 2D mask loss $\mathcal{L}_{mask}$ are identical as those defined in [14]. In order to accelerate the network training and keep the functionality of the pre-trained multi-task module (*e.g.*, mask head for instance segmentation), we can freeze these heads and their correspond-

ing child nodes, *i.e.*, the convolutional backbone and set the $\mathcal{L}_{det}$ to zero during back propagation phase. $\mathcal{L}_{inst}$ denotes the individual instance loss for 6DoF estimation with sub-class categorisation. Specifically, it is defined as a triple-loss: $\mathcal{L}_{inst} = \lambda_{sub\_cls}\mathcal{L}_{sub\_cls} + \lambda_{rot}\mathcal{L}_{rot} + \lambda_{trans}\mathcal{L}_{trans}$, where $\lambda_{sub\_cls}, \lambda_{rot}, \lambda_{trans}$ are hyper-parameters used to balance their corresponding loss. Next we explain the design choices for the above triple losses.

Sub-category classification loss $\mathcal{L}_{sub\_cls}$. Sub-category denotes the finer class of the vehicle corpus: *e.g., Audi-A6, BMW-530, Benze-ML500, etc*. In order to balance the rare cases for infrequently appearing cars in the training images, weighted cross entropy is used for sub-category classification loss.

Rotation loss $\mathcal{L}_{rot}$. There are generally three representations for providing orientation information: Euler angles, $SO(3)$ rotation matrices and Quaternions. Euler angles are easily understandable and interpretable parametrisation of 3D rotation. However, there are two issues when directly regressing the Euler angles: (1) non-injectivity: the same angle could be represented by multiple values due to the wrapping around 2π radians, which make the regression a non uni-modal task; (2) Gimbal lock: possible loss of one degree of freedom does not make Euler angles invalid but makes them unsuited for practical applications. Given 3D models of the objects, one way to work around the problem is to rotate each view at fixed intervals to cover the whole $SO(3)$ and then find the nearest neighbour [32] or closest viewpoint [22], which treat the rotation estimation problem as a classification problem. But this requires a complete CAD model of the object and a discretisation step of orientation angles. To estimate rotation matrix directly, [6] propose a LieNet to regress a Lie algebra based on rotation representation. However, a 3×3 orthogonal matrix is over-parameterised and enforcing the orthogonality is non-trivial.

Quaternions are favourable due to the universality mapping from 4 dimensional values to legitimate rotations. This is a simpler process than the orthonormalisation of rotation matrices. Quaternions are continuous and smooth, lying in a unit manifold, which can be easily enforced through back-propagation. Therefore, the rotation head in our network focuses on the regression of quaternion representation. However, the main problem with quaternions is that they are not injective: the quaternion $\mathbf{q}$ and $-\mathbf{q}$ represent the same rotation because two unique values (from each hemisphere) map to a single rotation. To address the issue, we constrain all quaternions to one hemisphere such that there is a unique value for each rotation [2]. Hence, for the rotation head, given the ground truth unique quaternion $\mathbf{q}$ and the predicted $\hat{\mathbf{q}}$, the rotation loss is defined as:

$$\mathcal{L}_{rot}(\mathbf{q}, \hat{\mathbf{q}}) = \|\mathbf{q} - \frac{\hat{\mathbf{q}}}{\|\hat{\mathbf{q}}\|}\|_{\gamma} \tag{1}$$

An important choice for regressing in Euclidean space is the regression norm $\|\|_{\gamma}$. Typically, deep learning models use $L_1 = \|\|_1$ or $L_2 = \|\|_2$. With the datasets used in this paper, we found the L_1 norm performs better: the error does not increase quadratically with magnitude nor over-attenuate large residuals.

Translation loss $\mathcal{L}_{trans}$. Regressing translation vector in world unit instead of pixel unit stabilises the loss. The transformation of the detected object takes 2D bounding box centre, height and width u_p, v_p, h_p, w_p in pixel space and then outputs their corresponding u_w, v_w, h_w, w_w in world unit as:

$$u_w = \frac{(u_p - c_x)z_s}{f_x}, v_w = \frac{(v_p - c_x)z_s}{f_y}, h_w = \frac{h_p}{f_x}, w_w = \frac{w_p}{f_y}$$

where the matrix $[f_x, 0, c_x; 0, f_y, c_y; 0, 0, 1]$ is the camera intrinsic calibration matrix.

Huber loss is adopted to describe the penalty in translation estimation: give ground truth 3 dimensional translation vector $\mathbf{t}$ and the prediction $\hat{\mathbf{t}}$, the translation loss is:

$$\mathcal{L}_{trans}(\mathbf{t}, \hat{\mathbf{t}}) = \begin{cases} \frac{1}{2}(\mathbf{t} - \hat{\mathbf{t}})^2/\delta & \text{if } |\mathbf{t} - \hat{\mathbf{t}}| < \delta, \\ |\mathbf{t} - \hat{\mathbf{t}}| - \frac{1}{2}\delta & \text{otherwise.} \end{cases} \tag{2}$$

where the hyperparamter δ controls the boundary of outliers. If δ is set to 1, then it becomes the smooth-L1 loss used in [10]. In this paper, δ is set as 2.8 which is the cut off threshold for translational evaluation as described in Sec. 4.1.

3.3. Weighted Non-local neighbour embedding

In order to capture spatial dependencies among detected objects of interest, we introduce a non-local block with a weighted operation. We reason that the dependencies among neighbouring objects will assist the network to regularise the 6DoF pose estimation collectively better than treating them individually. For example, neighbouring cars on the same lane will follow almost the same orientation and maintain certain distance. There are several advantages of using a weighted non-local operations comparing with other social embedding schemes [37, 1]: (i) non-local operations capture long-range dependencies directly by computing interactions between any two positions, regardless of their positional distance; (ii) non-local operations maintain the variable input sizes and can be easily combined with other operations; (iii) our proposed weighted operation renders it possible to associate the output maps with self-attention mechanisms for better interpretability.

[2] We enforce the uniqueness as in Appendix.

The non-local means (NL-means) is first introduced in [4], based on a non-local averaging of all pixels in the image. Later [41] introduced the *non-local* operations as an efficient and generic component for capturing long-range dependencies with deep neural networks. The non-local operations maintain the variable input sizes. Intuitively, a non-local operation computed the response at a position as a weighted sum of the features at all positions in the input feature maps. The generic non-local operation in deep neural network is defined as:

$$y_i = \frac{1}{\mathcal{C}(x)} \sum_{\forall j} f(x_i, x_j) g(x_j) \tag{3}$$

where i is the index of an output position (here in space) whose response is to be computed and j is the index that enumerates all possible positions. x is the input signal and y is the output signal of the same size as x. A pairwise function f computes a scalar (representing relationship such as affinity) between i and j. The unary function g computes a representation of the input signal at the position j. The response is normalised by a factor $\mathcal{C}(x)$. The non-local models are not sensitive to the design choices of f and g. For simplicity and fast computation, we consider g in the form of a linear embedding: $g(x_j) = W_g x_j$, where W_g is a weight matrix to be learnt. The pairwise function f is implemented in the form of *embedding Gaussian* as: $f(x_i, x_j) = e^{\theta(x_i)^T \phi(x_j)}$, where $\theta(x_i) = W_\theta x_i$ and $\phi(x_j) = W_\phi x_j$ are two embeddings. And $\mathcal{C}(x) = \sum_{\forall j} f(x_i, x_j)$. The recently proposed self-attention module [36] is a special case of non-local operations in the embedding Gaussian version: when, for a given i, $\frac{1}{\mathcal{C}(x)} \sum f(x_i, x_j)$ becomes *softmax* computation along the dimension j. So we have $y = softmax(x^T W_\theta^T W_\phi x) g(x)$.

However, we found out that when the input dimension d (for a feature map of $H \times W \times C$ where C is the channel number, $d = H \times W$) gets large, the dot products grow large in magnitude, pushing the softmax function into regions where it has extreme gradients. As a result, y will have extreme value as well.[3] To counteract the effect, we propose to use a *weighted non-local operation* for calculating the self-attention map $\mathcal{A}$ as:

$$\mathcal{A} = softmax(\frac{x^T W_\theta^T W_\phi x}{\sqrt{d}}) \tag{4}$$

So that $y = \mathcal{A} \cdot g(x)$. The weighted non-local operation scales the dot-product attention variance to unit 1, which consequently does not push the softmax operation to extreme, saturated values. Similar technique is also adopted in [16], where temperature of the final softmax is raised so as to obtain soft targets. Intuitively, the weighted operation has the same form of expression. However, the suitable temperature in softmax is finicky to determine. Alternatively, we scale the dot-product input variance to unit 1. Consequently, the output map after softmax operation will provide a justifiable interpretation in the form of a self-attention formulation.

[3] To illustrate why the dot products get large, assume that the components of $\mathbf{x}$ are random variables with mean 0 and variance 1. Then its self dot product, $\mathbf{x}^T \cdot \mathbf{x} = \sum_i^d x_i^2$, has mean 0 and variance d (d is the input dimension).

4. Experiments on Apolloscape Dataset

We perform comprehensive studies on the challenging Apolloscape dataset. The Apolloscape 3D Car Instance challenge contains a diverse set of stereo video sequences recorded in the street scenes from different cities. There are 3941/208/1041 high quality annotated images in the training/validation/test set.[4] The monocular RGB images are of pixel size 2710×3384. It is worth noticing the high resolution of the images: the total number of pixels of a single image is 100 times than those of other canonical image datasets (*e.g.*, MS-COCO, Mapillary Vistas, ImageNet). The camera intrinsic parameters are provided in the form of camera focal lengths (f_x, f_y) and optical centres expressed in pixels coordinates (c_x, c_y). Car models are provided in the form of triangle meshes. The mesh models have around 4000 vertices and 5000 triangle faces. One example mesh model is shown as in Fig. 2. There are total 79 car models in three categories (sedan1, sedan2, SUV) with only 34 car models appearing in the training set. In addition, ignored marks are provided as unlabelled regions and we only use the ignored masks to filter out detected regions during test.

4.1. Evaluation Metrics

The evaluation metrics follow similar instance mean AP as the MS-COCO [29]. However, due to 3D nature, the 3D car instance evaluation has its own idiosyncrasies: instead of using 2D mask IoU to judge a true positive, the 3D metric used in this dataset contains the perspective of shape (s), 3D translation (t) and 3D rotation (r). The shape similarity score is provided by an $N_{car} * N_{car}$ matrix where N_{car} denotes the number of car models. For 3D translation and 3D rotation, the Euclidean distance and arccos distance are used for measuring the position and orientation difference respectively.

Specifically, given an estimated 3D car model in an image $C_i = \{s_i, t_i, r_i\}$ and ground truth model $C_i^* = \{s_i^*, t_i^*, r_i^*\}$, the evaluation for these three estimates are as follows: for 3D shape, reprojection similarity is considered by putting the model at a fix location and rendering 10 views (v) by rotating the object. Mean IoU is computed between the two poses (P) rendered from

[4] After manual examination, we have deleted visually distinguishable wrongly labelled images, leaving us with 3888/206 images for training/validation.

each view. Formally, the metric is defined as: $c_{shape} = \frac{1}{|V|}\sum_{v\in V} IoU(P(s_i), P(s_i^*))_v$, where V is a set of camera views. For 3D translation and rotation, the evaluation metric follows that of the canonical self-localisation: $c_{trans} = \| t_i - t_i^* \|^2$ and $c_{rot} = \arccos(|q(r_i) \centerdot q(r_i^*)|)$. Then, a set of 10 thresholds from loose criterion to strict criterion $(c_0, c_1, \ldots, c_9)$ is defined as:

$$shapeThrs - [.5 : .05 : .95]$$
$$rotThrs - [50 : 5 : 5]$$
$$transThrs - [2.8 : .3 : 0.1]$$

where the most loose metric c_0: $0.5, 50, 2.8$ means shape similarity > 0.5, rotation distance $< 50^\circ$ and translation distance < 2.8 metres, and stricter metrics can be interpreted correspondingly: all three criterion must be satisfied simultaneously so as to be counted as a true positive.

It is worth noting the strict translation distance threshold of 2.8 metres: it requires that the detected vehicle's distance from the camera centre needs to be correctly estimated within a 2.8 metres threshold even if the vehicle is hundreds metres away from the camera, otherwise the detection will be counted as a false positive. The precise translational estimation requirement is the major factor for the network to produce incorrect false positive, which is a challenging task from a human perspective as well.

4.2. Implementation Details

ResNet-101 is adopted as the convolutional body with Feature Pyramid Networks (FPN) as the detection backbone. The instance segmentation head is pre-trained using the Apolloscape scene dataset[5] with "car, motorcycle, bicycle, pedestrian, truck, bus, and tricycle" as 8 instance-level annotations.

The hyperparameters $\lambda_{sub\_cls}, \lambda_{rot}, \lambda_{trans}$ in Eqn. 3.2 are set to $1.0, 1.0, 0.1$ to scale the loss accordingly. To decrease the translational outlier penalty and stabilise the network training, the hyperparameter δ in Eqn.3.2 is set to 2.8 metres as the loose end of the translational metric in Sec. 4.1. The base learning rate starts from 0.01 with warm start-up scheme and the models are trained up to 5×10^4 iterations with learning rate divided by 10 at $1.5 \times 10^4 th$ and $3 \times 10^4 th$ iterations. We use a weight decay of 0.0001 and a momentum of 0.9. The RoIAlign takes a feature map of 7×7 from each RoI. The weighted non-local block is plugged into the last layer (5th) of the convolutional body with a receptive field of 32×32. During training, the images are resized with the largest side randomly in [2000, 2300]. Due to memory limitation, batch size is set as one image per single GPU. The incorporation of non-local block will increase the memory requirement, hence the training images are resized with the largest side randomly in [1500, 2000] when non-local block is plugged in. Top 1000 regions are chosen as per FPN level with 100 batch size per image. During testing phase, 0.1 is chosen as the detection threshold in the Faster R-CNN head with multi-scale augmentation.

[5]http://apolloscape.auto/scene.html

	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	AP_{XL}
Single Scale	0.57	0.87	0.62	0.34	0.50	0.65	0.73
Multiple Scale	0.59	0.89	0.64	0.34	0.51	0.68	0.84

Table 1: 2D bounding box detection mAP from the Faster R-CNN head. It is the upperbound for 3D detection result. Subscripts of AP represent squared object one side size in pixels as: S:28-56, M:56-112, L: 112-512, XL:512+.

4.3. Main Results & Ablation Studies

We present results of 6D-VNet that reaches the 1st place in Apolloscape challenge 3D Car Instance task. Furthermore, we perform comprehensive evaluations to analyse the "bells and whistles" in 6D-VNet, which further improve state-of-the-art (Tab. 4). Our ablation studies gradually incorporated all components and are detailed as follows.

Effect of End-to-End Training. We first provide the 2D bounding box mAP from Faster R-CNN head to serve as the upper bound by 2D object detection as in Tab. 1. We can find that small objects are more challenging to detect. Small object also indicates the object longitudinal axis distance is typically far away from the camera. The accurate estimation of large translational distance value is thus more challenging.

In Tab. 2 we show that the translational head is crucial for improving the mAP in our end-to-end training scheme. The projective distance estimation was the defacto approach in previous state-of-the-art methods [32, 22, 6] as a second stage to measure the translational distance. The depth component estimation is treated as an entirely independent process given the rotation pose estimation [6]. Hence, the geometric post-processing method achieves around only 3.8%mAP due to the crude translation estimation. We show that using the same bounding box information (in world unit) to train a translation regression head improves the mAP to 8.8%.

Effect of Joint Losses. We then investigate the synergistic training of using visual information to regress translational vector as the concatenation operation $\oplus$ in Fig. 3. The w/o $\oplus$ row in Tab. 2 represents the translation head using only normalised world unit bounding box information, the $\oplus$ row represents the translation head that combines the intermediate visual information from RoIAlign with the bounding box information. The overall mAP is further improved by 2% using the RoIAlign intermediate feature. It is worth noting the synergy of the joint losses reflecting through the

[6]See Appendix for a more detailed description for projective distance estimation.

Method	Intermediate training value			val mAP	test mAP
	shape sim	rot dist	trans dist		
projective distance	0.84	13.5°	-	0.038	0.0371
w/o $\oplus$	0.86	11.9°	4.6	0.088	0.0882
$\oplus$	0.87	10.5°	2.3	0.108	0.1049
$\oplus$ (fine-tune)	0.90	8.8°	1.3	0.128	0.1223

Table 2: Effect of end-to-end training and joint losses. The projective distance is the most commonly adopted approach in previous state-of-the-art methods [32, 22, 6] as a second stage to measure the translational distance. w/o $\oplus$ denotes the network without the concatenation operation in Fig. 3. $\oplus$ represents the branch trained with joint losses. $\oplus$ (fine-tune) denotes, when training the network, the learnt parameters in convolutional body and Faster R-CNN head are unfreezed.

Method	Inference Time (second)				val mAP	test mAP
	Det Head	Triple Head	Misc	Total		
w/o Weighted Non-local (Single Scale)	0.966	1.322	0.002	2.40	0.128	0.1223
w/o Weighted Non-local (Multi-Scale)	5.535	1.381	0.004	6.86	0.143	0.1412
Weighted Non-local (Multi-Scale)	5.810	1.473	0.003	7.29	0.146	0.1435

Table 3: Effect of Weighted Non-local Block and runtime analysis.

intermediate training value columns in Tab. 2: the shape similarity, rotation and translational score improve by 0.01, 1.4° and 2.3 metres respectively. By connecting the translation head with intermediate RoIAlign branch, the translational losses are jointly back propagated with the losses from the sub-categorisation and rotation. The improvement of translational estimation synergistically boost the accuracy for shape and rotation estimation. It shows that the network is able to learn the implicit information that is shared amongst the object class, rotation and translation.

Effect of Weighted Non-local Block. The weighed non-local block is inserted into the last layer of the ResNet to encode dense spatial dependencies. Fig. 4 shows that by plugging in the weighted non-local block, both the rotational and translational training losses further decrease comparing with the previously converged model. Tab. 3 shows that the incorporation of weighted non-local block consistently improves mAP on both validation and test set with marginally increased inference runtime.

We visualise the self-attention map $\mathcal{A}$ of Eqn. 4 in the weighted non-local block in Fig. 5. The heat maps on the right exhibit meaningful attention positions (without the weighted operation, the heat map will exhibit extreme high temperature with hard to interpret positions on the attention map), which demonstrates that the weighted non-local block can learn to find meaningful relational clues regardless of the distance in space. In Fig. 6, we visualise the predicted vehicles in 3D space[7]. When incorporated with weighted non-local block, the model is able to capture the spatial dependencies so that it adjusts the predictions according to the distance and orientation of neighbouring vehicles.

[7]We use Open3D [45] to render the triangle mesh.

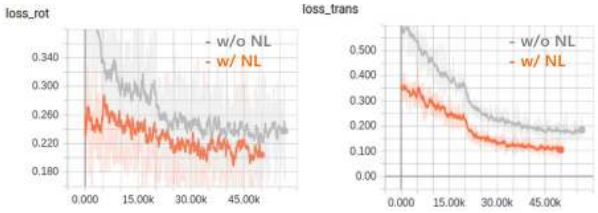

Figure 4: Losses w/o & w/ weighted non-local block (NL).

5. Conclusions

We have presented 6D-VNet as an end-to-end network for 6DoF pose estimation of vehicles from monocular RGB images. To the best of our knowledge, the incorporation of translation regression into the network is the first of its kind. In addition, we design the joint losses according to solid grounding in geometry, which are crucial to achieve accurate pose estimation. Particularly, a large improvement for position estimation is observed when the translation head is trained from both visual clues and bounding box information. Furthermore, we demonstrate the spatial dependencies among neighbouring vehicles can be incorporated via a weighted non-local block with interpretable self-attention map. Our future work will include further refinement of the estimated 6DoF pose using post-processing techniques such as iterative closet point based algorithms [44] or iterative refinement network in [26].

Acknowledgement. The authors gratefully acknowledge the support of by the National Natural Science Foundation of China (NSFC) No.61871273, China Postdoctoral Science Foundation No. 174112.

PD	TH	TV	FT	MS	NL	IM	QS	mAP	c_0	c_5	AP^S	AP^M	AP^L	AR^1	AR^{10}	AR^{100}	AR^S	AR^M	AR^L
✔								0.037	0.125	0.042	0.072	0.042	0.076	0.016	0.091	0.128	0.072	0.133	0.245
	✔							0.067	0.190	0.089	0.069	0.057	0.194	0.026	0.116	0.139	0.069	0.138	0.305
	✔	✔						0.088	0.237	0.122	0.100	0.077	0.246	0.030	0.141	0.174	0.100	0.169	0.360
	✔	✔	✔					0.121	0.331	0.164	0.126	0.115	0.297	0.035	0.162	0.231	0.126	0.246	0.424
	✔	✔	✔	✔				0.141	0.351	0.198	0.128	0.131	0.357	0.040	0.179	0.243	0.128	0.253	0.477
	✔	✔	✔	✔	✔			0.144	0.352	0.199	0.131	0.133	0.360	0.040	0.187	0.249	0.131	0.261	0.481
	✔	✔	✔	✔	✔	✔		0.147	0.357	0.206	0.117	0.137	0.366	0.041	0.190	0.246	0.117	0.262	0.489
	✔	✔	✔	✔	✔	✔	✔	0.148	0.353	0.209	0.115	0.138	0.371	0.042	0.191	0.244	0.115	0.259	0.490

Table 4: Performance on test set in terms of mAP. c_0 is the most loose criterion for evaluating AP and c_5 is in the middle of criterion. Superscript S, M, L of average precision (APs) and average recall (ARs) represent the object sizes. Superscript number $1, 10, 100$ represent the total number of detections for calculating recalls. Projective distance (PD) [6] estimation is adopted in the state-of-the-art methods [6, 22, 32]. Triple head (TH) is the baseline 6D-VNet. Translation head is then concatenated with visual branch (TV), represented by $\oplus$ operation in Fig. 3. Fine-tuning (FT) the convolutional body and detection head gives the task specific network a 3% boost in mAP. Multi-scale testing (MS) further increases the accuracy. The incorporation of weighted non-local block (NL) improves both precision and recall. Using ignore mask (IM) to filter 2D detection bounding box with 0.5 IoU as threshold improves the precision, however, slightly degrades the recall. Finally, enforcing the quaternions to one hemisphere (QS) achieves the current state-of-the-art.

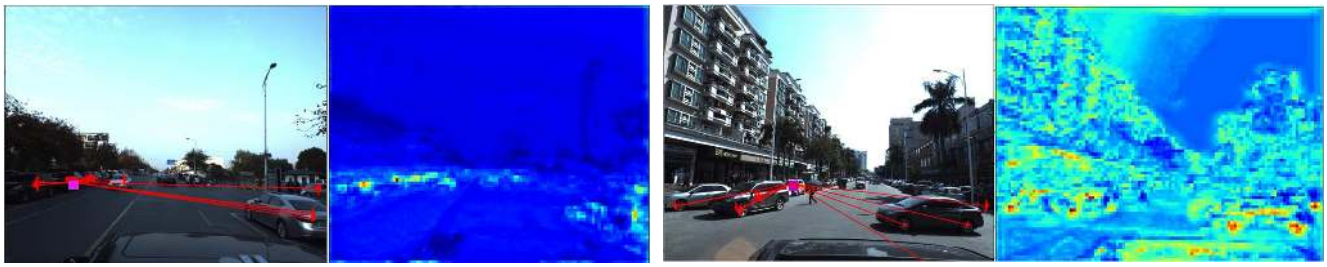

Figure 5: Examples of the behaviours of a weighted non-local block. These two examples are from held-out validation images. On the left images, the starting point (pink square) represents one x_i. Since we insert the weighted non-local block in res5 layer, one pink square denotes a receptive field of pixel size 32×32. We position the starting points on one of the vehicles. The end points of arrows represent x_j. The 10 highest weighted arrows for each x_i are visualised. The arrows clearly indicate that the weighted non-local block attends to the neighbouring traffic participants. Note that in the second illustration, end points are able to locate neighbouring vehicles' wheels and rear-view mirrors which are critical clues to position and orient the vehicle. The right images are the visualisations of self-attention map $\mathcal{A}$ of x_i from Eqn. 4. These visualisations show how the weighted non-local block finds interpretable, relevant indications of neighbouring vehicles to adjust the pose estimation.

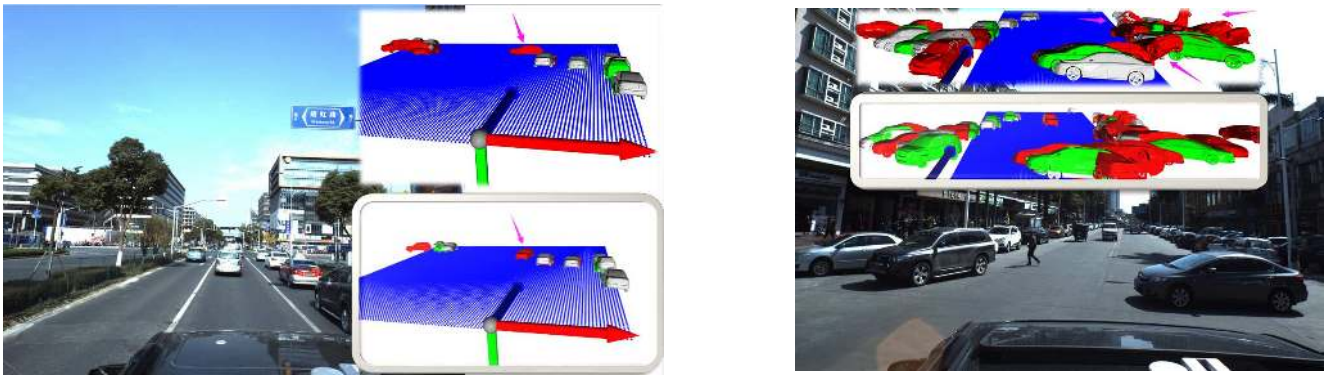

Figure 6: 3D renderings of predicted vehicles with camera coordinate axis. The bottom renderings framed by silver plate represent the results from the model with weighted non-local block. The vehicles in colour silver, green, red represent ground truth, true positive and false positive respectively. Due to the strictly fixed 2.8-metre translation criterion, vehicles that are farther way from the cameras are more difficult to measure. The pink arrows highlight the predictions that have been adjusted according to their neighbouring vehicles when weighted non-local block is plugged in.

References

[1] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese. Social-lstm: Human trajectory prediction in crowded spaces. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[2] E. Brachmann, A. Krull, F. Michel, S. Gumhold, J. Shotton, and C. Rother. Learning 6d object pose estimation using 3d object coordinates. In *European conference on computer vision (ECCV)*, 2014.

[3] E. Brachmann, F. Michel, A. Krull, M. Ying Yang, S. Gumhold, et al. Uncertainty-driven 6d pose estimation of objects and scenes from a single rgb image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[4] A. Buades, B. Coll, and J.-M. Morel. A non-local algorithm for image denoising. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2005.

[5] A. G. Buch, L. Kiforenko, and D. Kraft. Rotational subgroup voting and pose clustering for robust 3d object recognition. In *IEEE International Conference on Computer Vision (ICCV)*, 2017.

[6] M.-T. Do, T. Pham, M. Cai, and I. Reid. Real-time monocular object instance 6d pose estimation. In *British Machine Vision Conference (BMVC)*, 2018.

[7] P. Dollar, C. Wojek, B. Schiele, and P. Perona. Pedestrian detection: An evaluation of the state of the art. *IEEE transactions on pattern analysis and machine intelligence*, 2012.

[8] M. Everingham, S. A. Eslami, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The pascal visual object classes challenge: A retrospective. *International journal of computer vision*, 2015.

[9] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[10] R. Girshick. Fast r-cnn. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[11] R. Girshick, I. Radosavovic, G. Gkioxari, P. Dollár, and K. He. Detectron. `https://github.com/facebookresearch/detectron`, 2018.

[12] G. Gkioxari, R. Girshick, P. Dollár, and K. He. Detecting and recognizing human-object interactions. *arXiv preprint arXiv:1704.07333*, 2017.

[13] R. A. Güler, N. Neverova, and I. Kokkinos. Densepose: Dense human pose estimation in the wild. *arXiv preprint arXiv:1802.00434*, 2018.

[14] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2017.

[15] S. Hinterstoisser, V. Lepetit, N. Rajkumar, and K. Konolige. Going further with point pair features. In *European Conference on Computer Vision (ECCV)*, 2016.

[16] G. Hinton, O. Vinyals, and J. Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[17] T. Hodan, P. Haluza, Š. Obdržálek, J. Matas, M. Lourakis, and X. Zabulis. T-less: An rgb-d dataset for 6d pose estimation of texture-less objects. In *Applications of Computer Vision (WACV)*, 2017.

[18] T. Hodan, F. Michel, E. Brachmann, W. Kehl, A. GlentBuch, D. Kraft, B. Drost, J. Vidal, S. Ihrke, X. Zabulis, C. Sahin, F. Manhardt, F. Tombari, T.-K. Kim, J. Matas, and C. Rother. Bop: Benchmark for 6d object pose estimation. In *The European Conference on Computer Vision (ECCV)*, September 2018.

[19] T. Hodaň, X. Zabulis, M. Lourakis, Š. Obdržálek, and J. Matas. Detection and fine 3d pose estimation of texture-less objects in rgb-d images. In *IEEE International Conference on Intelligent Robots and Systems (IROS)*, 2015.

[20] R. Hu, P. Dollár, K. He, T. Darrell, and R. Girshick. Learning to segment every thing. *arXiv preprint*, 2018.

[21] X. Huang, X. Cheng, Q. Geng, B. Cao, D. Zhou, P. Wang, Y. Lin, and R. Yang. The apolloscape dataset for autonomous driving. *arXiv: 1803.06184*, 2018.

[22] W. Kehl, F. Manhardt, F. Tombari, S. Ilic, and N. Navab. Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2017.

[23] W. Kehl, F. Milletari, F. Tombari, S. Ilic, and N. Navab. Deep learning of local rgb-d patches for 3d object detection and 6d pose estimation. In *European Conference on Computer Vision (ECCV)*. Springer, 2016.

[24] A. Kendall, R. Cipolla, et al. Geometric loss functions for camera pose regression with deep learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[25] A. Kendall, M. Grimes, and R. Cipolla. Posenet: A convolutional network for real-time 6-dof camera relocalization. In *Proceedings of the IEEE international conference on computer vision (ICCV)*, 2015.

[26] Y. Li, G. Wang, X. Ji, Y. Xiang, and D. Fox. Deepim: Deep iterative matching for 6d pose estimation. In *The European Conference on Computer Vision (ECCV)*, 2018.

[27] T.-Y. Lin, P. Dollár, R. B. Girshick, K. He, B. Hariharan, and S. J. Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[28] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. Focal loss for dense object detection. *arXiv preprint arXiv:1708.02002*, 2017.

[29] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision (ECCV)*, 2014.

[30] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. Ssd: Single shot multibox detector. In *European conference on computer vision (ECCV)*, 2016.

[31] I. Radosavovic, P. Dollár, R. Girshick, G. Gkioxari, and K. He. Data distillation: Towards omni-supervised learning. *arXiv preprint arXiv:1712.04440*, 2017.

[32] M. Sundermeyer, Z. Marton, M. Durner, and R. Triebel. Implicit 3d orientation learning for 6d object detection from rgb

images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

[33] A. Tejani, D. Tang, R. Kouskouridas, and T.-K. Kim. Latent-class hough forests for 3d object detection and pose estimation. In *European Conference on Computer Vision (ECCV)*. Springer, 2014.

[34] B. Tekin, S. N. Sinha, and P. Fua. Real-time seamless single shot 6d object pose prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[35] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *Intelligent Robots and Systems (IROS)*, 2017.

[36] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems(NIPS)*, 2017.

[37] A. Vemula, K. Muelling, and J. Oh. Social attention: Modeling attention in human crowds. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[38] J. Vidal, C.-Y. Lin, and R. Martí. 6d pose estimation using an improved method based on point pair features. In *International Conference on Control, Automation and Robotics (ICCAR)*, 2018.

[39] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2001.

[40] P. Wang, R. Yang, B. Cao, W. Xu, and Y. Lin. Dels-3d: Deep localization and segmentation with a 3d semantic map. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[41] X. Wang, R. Girshick, A. Gupta, and K. He. Non-local neural networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[42] P. Wohlhart and V. Lepetit. Learning descriptors for object recognition and 3d pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[43] Y. Wu and K. He. Group normalization. In *European Conference on Computer Vision (ECCV)*, 2018.

[44] Z. Zhang. Iterative point matching for registration of free-form curves and surfaces. *International journal of computer vision*.

[45] Q.-Y. Zhou, J. Park, and V. Koltun. Open3D: A modern library for 3D data processing. *arXiv:1801.09847*, 2018.