arXiv:2210.13540v1 [cs.CV] 24 Oct 2022

Video based Object 6D Pose Estimation using Transformers

Apoorva Beedu
Georgia Institute of Technology
abeedu3@gatech.edu

Huda Alamri
Georgia Institute of Technology
halamri@gatech.edu

Irfan Essa
Georgia Institute of Technology
irfan@gatech.edu

Abstract

We introduce a Transformer based 6D Object Pose Estimation framework *VideoPose*, comprising an end-to-end attention based modelling architecture, that attends to previous frames in order to estimate accurate 6D Object Poses in videos. Our approach leverages the temporal information from a video sequence for pose refinement, along with being computationally efficient and robust. Compared to existing methods, our architecture is able to capture and reason from long-range dependencies efficiently, thus iteratively refining over video sequences. Experimental evaluation on the YCB-Video dataset shows that our approach is on par with the state-of-the-art Transformer methods, and performs significantly better relative to CNN based approaches. Further, with a speed of 33 fps, it is also more efficient and therefore applicable to a variety of applications that require real-time object pose estimation. Training code and pretrained models are available at https://github.com/ApoorvaBeedu/VideoPose.

1 Introduction

Estimating the 3D translation and 3D rotation for every object in an image is a core building block for many applications in robotics [44, 48, 14] and augmented reality [35]. The classical solution for such 6-DOF pose estimation problems utilises a feature point matching mechanism, followed by Perspective-n-Point (PnP) to correct the estimated pose [41, 47, 40, 21]. However, such approaches fail when objects are texture-less or heavily occluded. Typical ways of refining the 6DOF estimation involves using additional depth data [51, 7, 18, 25] or post-processing methods like Iterative Closest Point (ICP) or other deep learning based rendering methods [56, 23, 30, 45], which increase computational costs. Other approaches treat it as a classification problem [49, 23], resulting in reduced performance as the output space is not continuous.

In robotics, augmented reality, and mobile applications, the input signals are typically videos rather than single images, thus, giving opportunity for a multi-view framework. Li *et al.* [28] utilize multiple frames from different viewing angles to estimate single object poses. Wen *et al.* [55] and Deng *et al.* [13] use tracking methods to estimate the poses, however these methods do not explicitly exploit the temporal information in the videos. The idea of using more than one frame to estimate object poses has seen limited exploration. As the object poses in a video sequence are implicitly related to camera transformations and do not change abruptly between frames, and as different viewpoints of the objects aid in the pose estimation [27, 12], we believe that modelling temporal relationship can only aid in effective perfomance on the task.

NeurIPS 2022 Workshop on Vision Transformers: Theory and applications.

Motivated by this, we introduce a video based object $6D$ pose estimation framework, that uses past estimations to bound the $6D$ pose in the current frame. Specifically, we leverage the popular Transformer architecture [50, 42] with causal masked attention, where each input frame is only allowed to attend to frames that precede it. We train the model to jointly predict the 6D poses while also learning to accurately predict future features to match the true features. Such a setup has been employed in [15], which shows that predicting future features is an effective self-supervised pretext task for learning visual representations.

While the temporal architecture described above can be applied on top of any visual feature encoder (as discussed in ablations), we propose a purely transformer-based model that uses a Swin transformer [32] as the backbone. This enables our network to not only attend temporally to frames in the video, but also spatially within the frame.

In summary, the contributions of our paper are:

- We introduce a video based 6D Object pose estimation framework that is purely attention based.
- We incorporate self supervision via a predictive loss for learning better visual representations.
- We perform evaluation on the challenging YCB-Video dataset [56], where our algorithm achieves improved performance over state-of-the-art single frame methods such as PoseCNN [56] and PoseRBPF [13] with a real-time performance at 33fps, and transformer based method such as T6D-Dicrect [1].

2 Related Work

Estimating the 6-DOF pose of objects in the scene is a widely studied task. The classical methods either use template-based or feature-based approaches. In template-based methods, a template is matched to different locations in the image, and a similarity score is computed [18, 17]. However, these template matching methods could fail to make predictions for textureless objects and cluttered environments. In feature based methods, local features are extracted, and correspondence between known 3D objects and local 2D features is established using PnP to recover 6D poses [43, 39]. However, these methods also require sufficient textures on the object to compute local features and face difficulty in generalising well to new environments as they are often trained on small datasets.

Convolutional Neural Networks (CNNs) have proven to be an effective tool in many computer vision tasks. However, they rely heavily on the availability of large-scale annotated datasets. Due to this limitation, the YCB-Video [56], T-LESS [19], and OccludedLINEMOD datasets [26, 40] were introduced. They have enabled the emergence of novel network designs such as PoseCNN [56], DPOD [59], PVNet [40], and others [8, 52, 16, 10]. In this paper, we use the challenging YCB-Video dataset, as it is a popular dataset that serves as a testbed for many recent methods [1, 2, 13].

Building on those datasets, various CNN architectures have been introduced to learn effective representations of objects and to estimate accurate 6D poses. Kehl *et al.* [23] extend SSD [31] by adding an additional viewpoint classification branch to the network whereas [41, 47] predict 2D projections from 3D bounding box estimations. Other methods involve a hybrid approach where the model learns to perform multiple tasks, e.g., Song *et al.* [45] enforce consistencies among keypoints, edges, and object symmetries, and Billings *et al.* [6] predict silhouettes of objects along with object poses. There is also a growing trend of designing model agnostic features [53] that can handle novel objects. Finallly, few shot, one shot, and category level pose estimation has also seen increased interest recently [11, 54, 46]

To refine the predicted poses, several works use additional depth information and perform the standard ICP algorithm [56, 23], directly learn from RGB-D inputs [51, 30, 59], or through neural rendering [57, 22, 30, 33]. We argue that since the input signals to robots and/or mobile devices are typically video sequences, instead of heavily relying on post processing refinement using additional depth information and rendering, estimating poses in videos by exploiting the temporal data could already refine the single pose estimations. Recently, several tracking algorithms are utilising videos to estimate object poses. A notable work from Deng *et al.* [13] introduces the PoseRBPF algorithm that uses particle filters to track objects in video sequences. However, this state-of-the-art algorithm provides accurate estimations at a high computational cost. Wen *et al.* [55] also perform tracking, but use synthetic rendering of the object at the previous time-step.

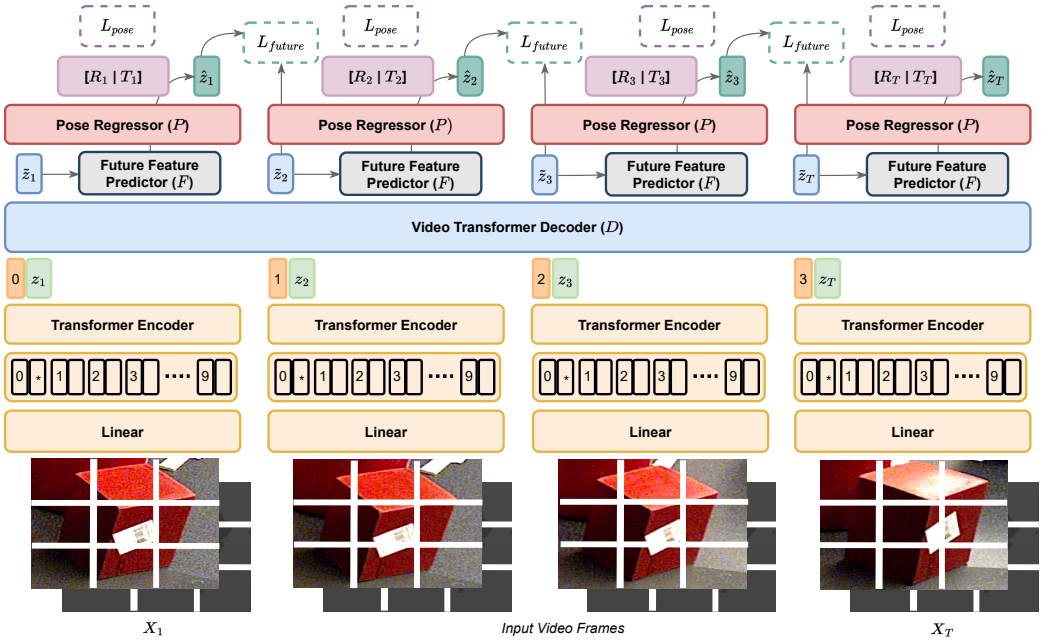

Figure 1: Overview of our framework for 6D object pose estimation. We use Swin transformer [32] as Transformer Encoder, and GPT2 [42] as Video Transformer Decoder. Future Feature Predictor and Pose Regressor consists of a 2 layer MLP, further described in Figure 2

With the rise in self-attention models and Transformer architectures [50, 42, 58], we also saw an increased interest in vision based transformers [3, 32, 5]. This has resulted in the application of Transformers to other applications like object detection [9, 62] and human pose estimation [60, 61, 29, 36, 34], and object pose estimation [37, 2, 1]. TD6-Direct [1] builds on the Detection Transformer (DETR) [9] architecture to directly regress to the pose, while [2] uses transformers to predict keypoints, and subsequently does keypoint regression. In contrast to these works, we use Transformer models to attend over a set of frames in a video, and directly regress to the 6D pose. As transformers use a self-attention mechanism, our framework is capable of learning and refining 6D poses from previous frames, without needing additional post process refinement.

3 Approach

Given an RGB-D video stream, our goal is to estimate the 3D rotation and 3D translation of all the objects in every frame of the video. We assume the system has access to the 3D model of the object. In the following sections, $\boldsymbol{R}$ denotes the rotation matrix with respect to the annotated canonical object pose, and $\boldsymbol{T}$ is the translation from the object to the camera.

3.1 Overview of the network

Our pipeline, as shown in Figure 6, is a two stage network. The first stage comprises of a feature extraction module; We use a Swin transformer [32] to learn the visual features for every frame in the video. For a given video sequence and its corresponding depth, the transformer encoder gives us a feature vector of shape $b \times t \times n \times 768$ where **b** corresponds to the batch size, **t** corresponds to the temporal length and **n** corresponds to the number of objects in the image.

Pose Estimation relies on accurate object detection, which derives the class-id and Region-Of-Interest (ROI). During training, we use the ground truth bounding box whereas during testing, we use the predictions and bounding box from the PoseCNN model. This can potentially be replaced with any lightweight feature extraction model such as MobileNet [20] to make the inference faster, or DETR [9] - a transformer based object detection module. We also use depth as an additional input.

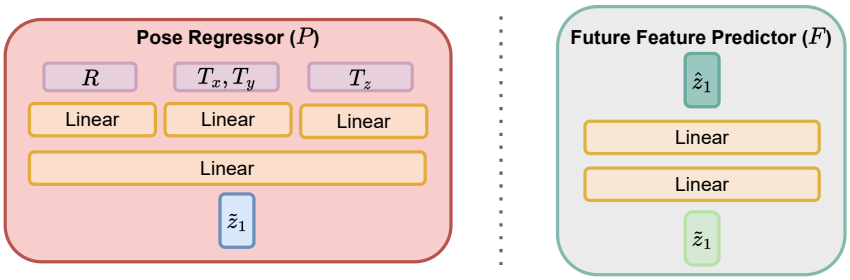

Figure 2: (*Left*) **Pose Regressor (P)**: Features from the temporal decoder is passed through a single linear layer, which is then passed through 3 separate linear layers for the estimation of $\boldsymbol{R}$, $(\boldsymbol{T_x}, \boldsymbol{T_y})$ and $\boldsymbol{T_z}$. (*Right*) **Future Feature Predictor (F)**: Features from decoder are passed through a 2 layer MLP to predict the future features.

In this paper, we use ground truth depth images; however, they can be replaced with other depth estimation modules.

3.2 Video Transformer Decoder

Given the features extracted by the encoder, the decoder attends to the previous features and predicts the 6D poses using the Pose Regressor, and the future frame features using Future Feature Predictor networks. We denote $\tilde{z}_T$ for the decoded features, and $\hat{z}_T$ for the predicted future features. Specifically,

$$\hat{z}_1, \cdots, \hat{z}_T = F(\tilde{z}_1, \cdots, \tilde{z}_T) \tag{1}$$

And,

$$[R_T|T_T] = P(\tilde{z}_T) \tag{2}$$

3.3 6D Pose Regression

The Pose Regressor (P) architecture can be seen in Figure 2. Rotation $\boldsymbol{R}$ and Translation $\boldsymbol{T}$ have one common linear layer, which is then branched to three different linear layers. As $\boldsymbol{R}$, $(\boldsymbol{T_x}, \boldsymbol{T_y})$ and $\boldsymbol{T_z}$ occupy different latent spaces, we believe it is best to learn them separately, instead of a single vector of 7 values as $[\boldsymbol{R}|\boldsymbol{T}]$.

Translation

The translation vector $\boldsymbol{T}$ is the object location in the camera coordinate system. A naive way of estimating $\boldsymbol{T}$ is to directly regress to it. However, doing so cannot handle multiple object instances or generalise well to new objects. To tackle this problem, Xiang *et al.* [56] estimate $\boldsymbol{T}$ by localising the 2D object center in the image and estimating object distance from the camera. Suppose $\mathbf{c} = (c_x, c_y)^T$ are the centers of the object in the frame and T_z is either learnt or estimated from the depth image, then T_x and T_y can be estimated as:

$$\begin{bmatrix} c_x \\ c_y \end{bmatrix} = \begin{bmatrix} f_x \dfrac{T_x}{T_z} + p_x \\ f_y \dfrac{T_y}{T_z} + p_y \end{bmatrix}, \tag{3}$$

where f_x and f_y are focal lengths and $(p_x, p_y)^T$ are principal points. Since we have rough estimates of object locations from the noisy object detection inputs, we train our model to estimate $\Delta c_x, \Delta c_y$, and T_z. We then estimate T_x and T_y using the following equation:

$$\begin{bmatrix} c_x + \Delta c_x \\ c_y + \Delta c_y \end{bmatrix} = \begin{bmatrix} f_x \dfrac{T_x}{T_z} + p_x \\ f_y \dfrac{T_y}{T_z} + p_y \end{bmatrix}. \tag{4}$$

Rotation

Similar to [56], we represent the rotation $\boldsymbol{R}$ using quaternions: $\boldsymbol{R} = \{w, x, y, z\}$. However, we only predict the x, y, z values, and infer w, the real value as:

$$w = 1 - norm(x, y, z) \tag{5}$$

We noticed that doing this helps the training process, and the quaternions learnt are more bounded.

3.4 Training Strategy

The pose estimation loss is obtained by projecting the 3D points using the estimated and ground truth pose, and then computing their L_2 distance:

$$L_{\text{pose}}(\widetilde{\mathbf{q}}, \mathbf{q}) = \frac{1}{m} \sum_{x \in M} ||(R(\widetilde{\mathbf{q}})x + \widetilde{\mathbf{t}}) - (R(\mathbf{q})x + \mathbf{t})||^2, \tag{6}$$

where M denotes the set of 3D points and m is total number of points. $R(\widetilde{\mathbf{q}})$ and $R(\mathbf{q})$ indicate the rotation matrix computed from the quaternion representation as in [56]. In addition, we also add a cosine loss on the quaternions, and a regularisation loss to force the norm of the quaternion to be 1. Quaternions that represent rotations are unit norm, and forcing the norm to be bounded by 1 helps in the learning process by eliminating all non-plausible vector combinations.

$$L_{\text{reg}} = ||1 - \text{norm}(\widetilde{\mathbf{q}})||, \quad L_{\text{inner\_prod}} = 1 - \langle \widetilde{\mathbf{q}}, \mathbf{q} \rangle. \tag{7}$$

In addition to these losses, inspired by [15], we add a future feature loss that is defined as:

$$L_{future} = \sum_{t=1}^{T-1} ||\hat{\mathbf{z}}_{\mathbf{t}} - \tilde{\mathbf{z}}_{\mathbf{t+1}}||_2^2 \tag{8}$$

The total loss is defined as

$$L(\widetilde{\mathbf{q}}, \mathbf{q}, \widetilde{\mathbf{t}}, \mathbf{t}, \hat{\mathbf{z}}, \tilde{\mathbf{z}}) = L_{\text{future}} + L_{\text{pose}} + L_{\text{reg}} + L_{\text{inner\_prod}}. \tag{9}$$

3.5 Implementation

VideoPose is implemented using the PyTorch [38] framework. We use a learning rate of $1e^{-4}$ and the Adam optimiser [24] with a weight decay of $1e^{-6}$. We use the ReduceOnPlateau scheduler that decreases the learning rate by a factor of 0.9 with a patience of 3 epochs. We create video samples of 5 frames and train our model for 20 epochs with the learning schedule described above. The training was done on 4 A40 GPUs for 1 week or 20 epochs, whichever occurred first.

During training, we augment the input images with colour-jitter and noise, and the bounding box by extending the height and width randomly between 0 and 10% of the height and width of the object. While training the temporal block, we create videos with random time jumps in between. For instance, given a large video sequence, we create video samples $1 : n : 10 * n$, where n is a random number between 1 and 10, thus forcing the model to account for small and large jumps between consecutive frames.

4 Experiments

We compare our framework with PoseCNN[56], PoseRBPF[13] and a recent image based transformer model T6D-Direct [1]. To the best of our knowledge, this work presents the first foray towards predicting pose directly from videos. Hence, we also create a simpler video baseline using the PoseCNN architecture for feature extraction, and ConvGRU to model the temporal information, as can be seen in Figure 3. Further details about the baseline are provided in the Appendix A.

4.1 Dataset

We evaluate the proposed method on the YCB-Video dataset [56] (see Sec. 3.5 for reference). It contains 92 RGB-D video sequences of 21 objects, and contains both textured and textureless objects of varying shapes, and different levels of occlusion where about 15% of objects are heavily occluded. Objects are annotated with 6D poses, segmentation masks and depth images.

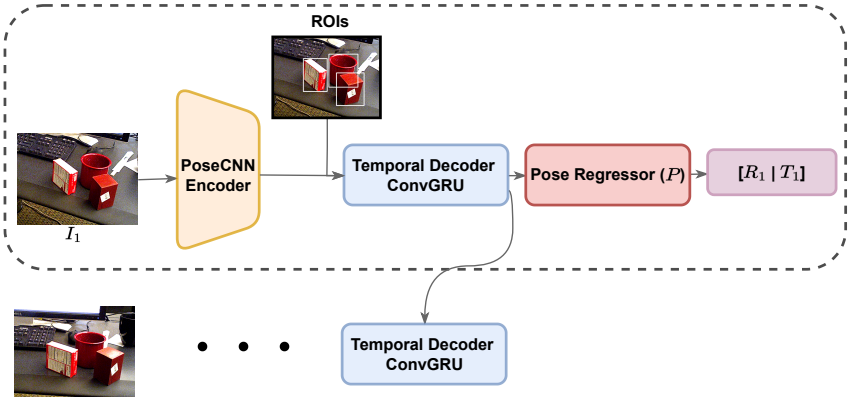

Figure 3: Overview of our baseline framework for 6D object pose estimation. We use the PoseCNN encoder [56] as feature extractor, and a ConvGRU [4] as a temporal decoder. The regression module is similar to Figure 2

Table 1: Comparison of performance between different architectures for single frame methods and video based baseline and our method, VideoPose. **Bold Green** values represent the best method across the tracking/video methods (The corresponding methods are highlighted with gray columns). Underlined values compare between **ALL** the methods in the table.

	PoseCNN		DeepIM		PoseRBPF (200 particles)		PoseRBPF (50 particles)		VideoPose Baseline (ConvGRU)		VideoPose (Transformer)	
	ADD	ADD-S	ADD	ADD-S	ADD	ADD-S	ADD	ADD-S	ADD	ADD-S	ADD	ADD-S
002_master_chef_can	50.9	84.0	71.2	93.1	58.0	77.1	56.1	75.6	36.3	80.9	**70.2**	**93.3**
003_cracker_box	51.7	76.9	83.6	91.0	**76.8**	**87.0**	73.4	85.2	22.5	62.1	43.3	78.2
004_sugar_box	68.6	84.3	83.6	91.0	**75.9**	**87.6**	73.9	86.5	40.7	68.8	58.1	82.5
005_tomato_soup_can	66.0	80.9	86.1	82.4	74.9	84.4	71.1	82.0	67.0	83.5	**83.3**	**91.1**
006_mustard_bottle	79.9	90.2	91.5	95.1	**82.5**	91.0	80.0	90.0	75.3	89.0	75.2	**91.8**
007_tuna_fish_can	70.4	87.9	87.7	96.1	59.0	79.0	56.1	73.8	60.6	86.4	**65.1**	**94.0**
008_pudding_box	62.9	79.0	82.7	90.7	57.2	72.1	54.8	69.2	49.6	75.8	**77.9**	**90.3**
009_gelatin_box	75.2	87.1	91.9	94.3	**88.8**	93.1	83.1	89.7	81.1	89.3	88.1	**93.1**
010_potted_meat_can	59.6	78.5	76.2	86.4	49.3	62.0	47.0	61.3	61.5	83.6	**71.1**	**89.3**
011_banana	72.3	85.9	81.2	91.3	24.8	61.5	22.8	64.2	22.3	69.6	**54.3**	**81.3**
019_pitcher_base	52.5	76.8	90.1	94.6	75.3	88.4	74.0	87.5	70.5	85.9	**78.0**	**90.6**
021_bleach_cleanser	50.5	71.9	81.2	90.3	54.5	69.3	51.6	66.7	46.9	62.1	**67.4**	**88.4**
024_bowl	6.5	69.7	8.6	81.4	**36.1**	86.0	26.4	**88.2**	12.8	80.1	13.0	78.8
025_mug	57.7	78.0	81.4	91.3	**70.9**	85.4	67.3	83.7	67.6	88.8	54.1	**91.7**
035_power_drill	55.1	75.8	85.5	92.3	**70.9**	**85.0**	64.4	80.6	36.0	71.2	58.8	82.7
036_wood_block	31.8	65.8	60.0	81.9	2.8	33.3	0.0	0.0	0.0	28.7	**7.0**	**68.6**
037_scissors	35.8	56.2	60.9	75.4	21.7	33.0	20.6	30.9	**50.1**	**73.4**	21.2	60.8
040_large_marker	58.0	71.4	75.6	86.2	48.7	59.3	45.7	54.1	36.8	53.9	**53.0**	**84.2**
051_large_clamp	25.0	49.9	48.4	74.3	**47.3**	76.9	27.0	73.3	20.7	69.3	34.0	**81.8**
052_extra_large_clamp	15.8	47.0	31.0	73.3	**26.5**	**69.5**	50.4	68.7	5.9	55.4	8.0	60.6
061_foam_brick	40.4	87.8	35.9	81.9	**78.2**	**89.7**	75.8	88.4	44.8	86.1	45.7	92.7
ALL	53.7	75.9	71.7	88.1	**59.9**	77.5	57.1	74.8	44.1	73.9	57.4	**85.3**

4.2 Metrics

We report the performance using two metrics: *(i)* ADD, which is the average distance between the corresponding points of the 3D object at the ground truth and predicted poses; and, *(ii)* ADD-S, which is designed for symmetric objects and calculates the mean distance from each 3D point to a closest point on the target model.

4.3 Evaluation

We compare our results with PoseCNN [56] for single frame prediction and PoseRBPF [13] for videos in Table 1. We also compare our results against DeepIM [30]. However, it is worth noting

Table 2: Comparison of performance between transformer based object pose estimations and convolution based frameworks. **Bold Green** values represent the best method across Transformer based methods (The corresponding methods are highlighted with gray columns). Underlined values compare between **ALL** the methods in the table. T6D-Direct [1] and VideoPose Transformer frameworks are the Transformer based methods that have been compared and DeepIM, PoseRBPF and PoseCNN are CNN based methods. Blue represents higher performance when bounding boxes from PoseCNN architecture is used.

	PoseCNN		DeepIM		PoseRBPF (200 particles)		T6D Direct		VideoPose (Transformer)		VideoPose (Transformer) (PoseCNN BBox	
	ADD	ADD-S	ADD	ADD-S	ADD	ADD-S	ADD	ADD-S	ADD	ADD-S	ADD	ADD-S
002_master_chef_can	50.9	84.0	71.2	93.1	58.0	77.1	61.5	91.9	**70.2**	**93.3**	68.7	92.0
003_cracker_box	51.7	76.9	83.6	91.0	76.8	87.0	**76.3**	**86.6**	43.3	78.2	31.6	75.1
004_sugar_box	68.6	84.3	83.6	91.0	75.9	87.6	**81.8**	**90.3**	58.1	82.5	52.2	80.7
005_tomato_soup_can	66.0	80.9	86.1	82.4	74.9	84.4	72.0	88.9	**83.3**	**91.1**	79.6	89.7
006_mustard_bottle	79.9	90.2	91.5	95.1	82.5	91.0	**85.7**	**94.7**	75.2	91.8	70.8	89.6
007_tuna_fish_can	70.4	87.9	87.7	96.1	59.0	79.0	59.0	92.2	**65.1**	**94.0**	65.9	94.0
008_pudding_box	62.9	79.0	82.7	90.7	57.2	72.1	72.7	85.1	**77.9**	**90.3**	68.8	87.9
009_gelatin_box	75.2	87.1	91.9	94.3	88.8	93.1	74.4	86.9	**88.1**	**93.1**	87.1	93.0
010_potted_meat_can	59.6	78.5	76.2	86.4	49.3	62.0	67.8	83.5	**71.1**	**89.3**	70.2	88.4
011_banana	72.3	85.9	81.2	91.3	24.8	61.5	**87.4**	**93.8**	54.3	81.3	50.6	81.5
019_pitcher_base	52.5	76.8	90.1	94.6	75.3	88.4	**84.5**	**92.3**	78.0	90.6	73.0	88.8
021_bleach_cleanser	50.5	71.9	81.2	90.3	54.5	69.3	65.0	83.0	**67.4**	**88.4**	56.0	81.3
024_bowl	6.5	69.7	81.4	81.4	36.1	86.0	**91.6**	**91.6**	78.8	78.8	77.8	77.8
025_mug	57.7	78.0	81.4	91.3	70.9	85.4	**72.1**	89.8	54.1	**91.7**	48.2	91.5
035_power_drill	55.1	75.8	85.5	92.3	70.9	85.0	**77.7**	**88.8**	58.8	82.7	49.8	81.7
036_wood_block	31.8	65.8	81.9	81.9	2.8	33.3	**90.7**	**90.7**	68.6	68.6	66.4	66.4
037_scissors	35.8	56.2	60.9	75.4	21.7	33.0	**59.7**	**83.0**	21.2	60.8	43.6	74.3
040_large_marker	58.0	71.4	75.6	86.2	48.7	59.3	**63.9**	74.9	53.0	**84.2**	51.5	82.1
051_large_clamp	25.0	49.9	74.3	74.3	47.3	76.9	78.3	78.3	**81.8**	**81.8**	67.9	67.9
052_extra_large_clamp	15.8	47.0	73.3	73.3	26.5	69.5	54.7	54.7	**60.6**	**60.6**	54.4	54.4
061_foam_brick	40.4	87.8	81.9	81.9	78.2	89.7	89.9	89.9	**92.7**	**92.7**	92.7	92.7
ALL	53.7	75.9	71.7	88.1	59.9	77.5	**74.6**	**86.2**	66.4	85.3	61.5	82.9

that the DeepIM is an iterative refinement framework that works on top of the PoseCNN results. PoseRBPF does not directly utilise video information, but uses a tracking pipeline to iteratively refine their poses. Several newer image based pose estimation methods have redefined the state-of-the-art, but for the scope of this paper, and for cleaner comparisons, we compare our framework with the aforementioned works. To the best of our knowledge, we believe that our work is the first to use videos directly in the estimation of 6D Object Pose. Therefore, we also compare against a simple CNN and ConvGRU based framework as shown in Figure 3.

From Table 1, we see that when compared to PoseRBPF, the simpler VideoPose (with the ConvGRU) shows improvements for several objects (e.g 037_scissors, 007_tuna_fish_can, 010_potted_mean_can etc.), whereas the Transformer based VideoPose outperforms PoseRBPF for most objects in ADD metric, and significantly outperforms for the ADD-S metric. These results indicate the effectiveness of using videos, and also the effectiveness of using attention in learning to estimate object poses. In Table 2, we compare T6D-Direct against our method, and we see that our framework is able to perform better for most of the objects, and comparably to the rest. VideoPose outperforms T6D-Direct for 12 of the 21 (57%) objects for the ADD-S metric, and 10 of the 21 objects for ADD metric. T6D-Direct, in addition to pose, also predicts the bounding boxes and class probabilities. We believe that these additional auxiliary losses have aided the model's performance, and as a future work, we would like to explore this space. In Table 2, we also provide results from using PoseCNN bounding boxes as opposed to GT bounding boxes, and we see that the performance drop is very minimal suggesting that the augmentation during the training has helped in the model learning to fix these predictions. AUC in Figure 4 shows that even for a difficult object such as a foam brick, our model performs significantly better than PoseCNN, and when the bounding boxes are not accurate, our method shows very slight drop in performance.

Effect of the number of previous frames used Table 4 shows the effect of number of previous frames used. We see that we get the best performance when 10 frames are used, indicating that the model performance is influenced by previous frames. It is worth noting that the model is trained for a video length of 5 frames. However, the model is still capable of extracting relevant information

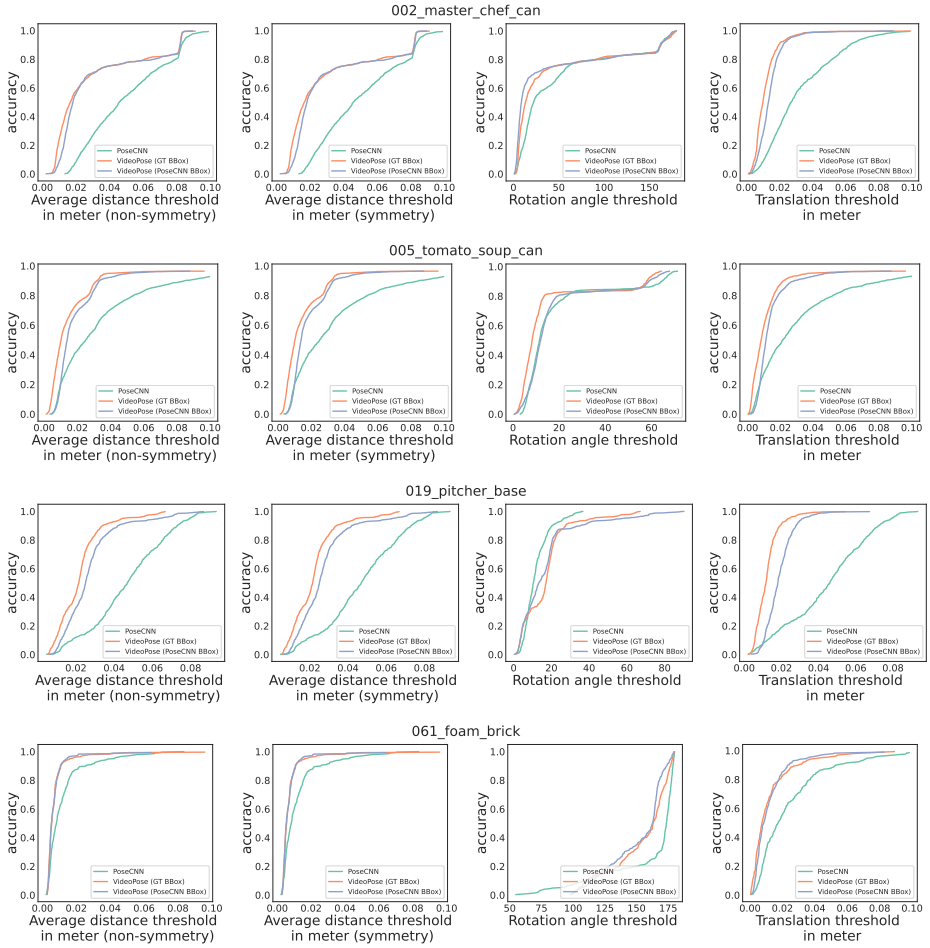

Figure 4: Accuracy-threshold curve for rotation and translation, and reprojection errors for few objects in the YCB-Video dataset. Emerald is used to plot PoseCNN curve, Orange for VideoPose Transformer with GT bounding boxes, and Purple for VideoPose Transformer with bounding boxes from PoseCNN architectures.

for longer-range videos. We believe that the augmentation described in Section 3.5 has helped the model in learning the long-range dependencies.

Time efficiency We compare the time efficiency of our model with several models. The run time for PoseCNN is taken from Wang *et al.* [51]. From Table 3, we see that VideoPose with the Beit Transformer [5] backbone performs the fastest, as opposed to the Swin Transformer used in our framework results. We saw a significant improvement in accuracy when using Swin Transformer compared to Beit Transformer, at a small decrease in the speed, and hence reported results for Swin backbone throughout the paper. However, this also indicates that the our framework's speed is affected by the backbone framework, and learning from video adds a very small time cost. In the future, when faster, and more accurate attention based image feature extractors are developed, our framework can be easily adapted to it. Our model is tested on GeForce GTX 1080 Ti.

Qualitative Analysis of the 6D predictions We show three examples of the predictions by VideoPose Transformer, PoseCNN, and ground truth poses in Table 5. The columns represent the 2D projections of predictions using VideoPose Transformer, PoseCNN, and the ground truth poses. We observe that the poses estimated using the VideoPose Transformer framework looks qualitatively more accurate than PoseCNN.

Table 3: Comparison of frame rates for different methods

Method	Time (fps)
PoseCNN [56]	5.88
PoseRBPF(50) [13]	20
PoseRBPF(200) [13]	5
CosyPose [27]	3
DeepIM [30]	12
T6D-Direct [1]	59
Ours(Swin backbone)	33
Ours(Beit backbone)	**67**

Table 4: Effect of video length on accuracies.

Video Length	ADD	ADD-S
2	63.9	84
5	64.3	84.4
7	64.1	84.2
10	**64.8**	**84.8**

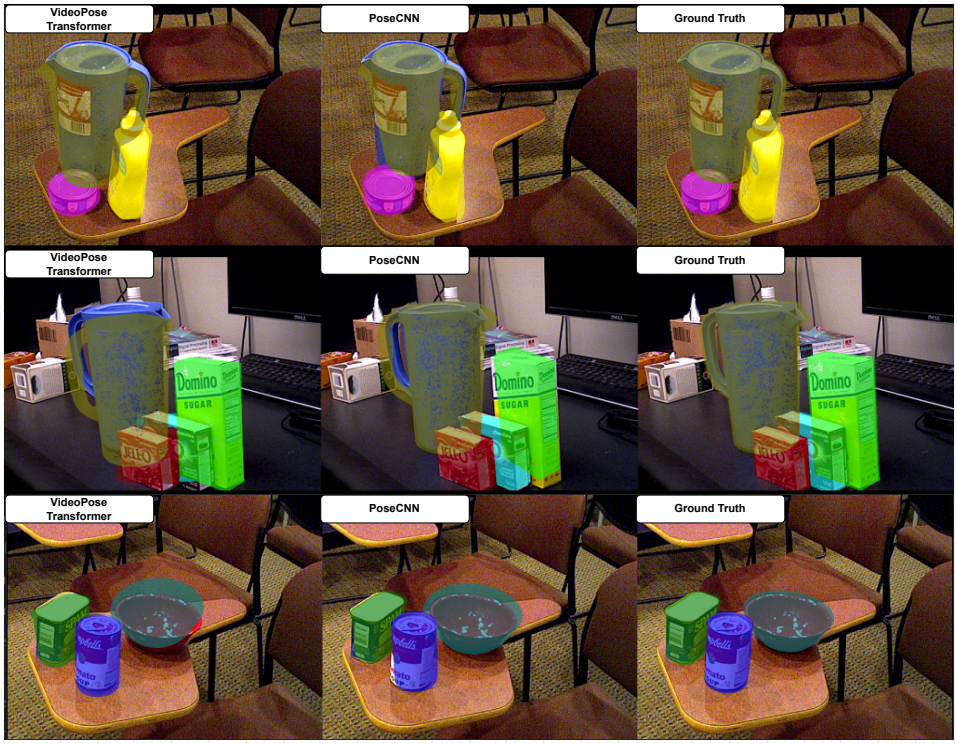

Figure 5: Visualisations of the estimated poses on YCB-Dataset: each row represents results at different time-steps. The columns consist of VideoPose, PoseCNN and Ground truth visualisations respectively.

5 Conclusion

In this work, we introduced an end-to-end self-attention based Transformer network for the estimation of 6D Object Pose called *VideoPose*. We demonstrate that by using the 6D predictions from the previous frames, we can significantly improve 6D predictions in the subsequent frames. We also conducted an extensive ablation study on different design choices of the network, and show that our model is able to learn and utilise the features from previous predictions regardless of the network choices. Finally, the proposed network performs in real-time at 33fps, thereby improving the time efficiency over previous approaches. As a future work, we would like to further improve our architecture with a better temporal module and model the relationship with the camera transformation and the objects. As T6D-Direct showed improvement with a single-stage network, detecting objects along with their poses, we would like to explore this idea in our future work as well. Our method successfully maintains consistency in pose estimation between frames, however, still depends on the initial frame estimation, and accurate bounding box prediction. We would like to investigate further on improving this, while maintaining/improving the computational efficiency.

References

[1] Arash Amini, Arul Selvam Periyasamy, and Sven Behnke. T6d-direct: Transformers for multi-object 6d pose direct regression. In *DAGM German Conference on Pattern Recognition*, pages 530–544. Springer, 2021.

[2] Arash Amini, Arul Selvam Periyasamy, and Sven Behnke. Yolopose: Transformer-based multi-object 6d pose estimation using keypoint regression. *arXiv preprint arXiv:2205.02536*, 2022.

[3] Anurag Arnab, Mostafa Dehghani, Georg Heigold, Chen Sun, Mario Lučić, and Cordelia Schmid. Vivit: A video vision transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6836–6846, 2021.

[4] Nicolas Ballas, Li Yao, Chris Pal, and Aaron Courville. Delving deeper into convolutional networks for learning video representations. *arXiv preprint arXiv:1511.06432*, 2015.

[5] Hangbo Bao, Li Dong, and Furu Wei. Beit: Bert pre-training of image transformers. *arXiv preprint arXiv:2106.08254*, 2021.

[6] Gideon Billings and Matthew Johnson-Roberson. Silhonet: An rgb method for 6d object pose estimation. *IEEE Robotics and Automation Letters*, 4(4):3727–3734, 2019.

[7] Eric Brachmann, Alexander Krull, Frank Michel, Stefan Gumhold, Jamie Shotton, and Carsten Rother. Learning 6D object pose estimation using 3d object coordinates. In *European conference on computer vision*, pages 536–551. Springer, 2014.

[8] Yannick Bukschat and Marcus Vetter. Efficientpose: An efficient, accurate and scalable end-to-end 6d multi object pose estimation approach. *arXiv preprint arXiv:2011.04307*, 2020.

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

[10] Hansheng Chen, Pichao Wang, Fan Wang, Wei Tian, Lu Xiong, and Hao Li. Epro-pnp: Generalized end-to-end probabilistic perspective-n-points for monocular object pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2781–2790, 2022.

[11] Wei Chen, Xi Jia, Hyung Jin Chang, Jinming Duan, Linlin Shen, and Ales Leonardis. Fs-net: Fast shape-based network for category-level 6d object pose estimation with decoupled rotation mechanism. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1581–1590, 2021.

[12] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. Multi-view 3D object detection network for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1907–1915, 2017.

[13] Xinke Deng, Arsalan Mousavian, Yu Xiang, Fei Xia, Timothy Bretl, and Dieter Fox. Poserbpf: A rao-blackwellized particle filter for 6d object pose tracking. *Robotics: Science and Systems (RSS)*, 2019.

[14] Clemens Eppner, Arsalan Mousavian, and Dieter Fox. A billion ways to grasp: An evaluation of grasp sampling schemes on a dense, physics-based grasp data set. *arXiv preprint arXiv:1912.05604*, 2019.

[15] Rohit Girdhar and Kristen Grauman. Anticipative Video Transformer. In *ICCV*, 2021.

[16] Yisheng He, Haibin Huang, Haoqiang Fan, Qifeng Chen, and Jian Sun. Ffb6d: A full flow bidirectional fusion network for 6d pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3003–3013, 2021.

[17] Stefan Hinterstoisser, Cedric Cagniart, Slobodan Ilic, Peter Sturm, Nassir Navab, Pascal Fua, and Vincent Lepetit. Gradient response maps for real-time detection of textureless objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 34(5):876–888, 2011.

[18] Stefan Hinterstoisser, Vincent Lepetit, Slobodan Ilic, Stefan Holzer, Gary Bradski, Kurt Konolige, and Nassir Navab. Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes. In *Proceedings of the Asian Conference on Computer Vision (ACCV)*, pages 548–562. Springer, 2012.

[19] Tomás Hodan, Pavel Haluza, Stepán Obdrzálek, Jiri Matas, Manolis I. A. Lourakis, and Xenophon Zabulis. T-less: An rgb-d dataset for 6d pose estimation of texture-less objects. *IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 880–888, 2017.

[20] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. MobileNets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[21] Yinlin Hu, Joachim Hugonot, Pascal Fua, and Mathieu Salzmann. Segmentation-driven 6d object pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3385–3394, 2019.

[22] Shun Iwase, Xingyu Liu, Rawal Khirodkar, Rio Yokota, and Kris M Kitani. Repose: Fast 6d object pose refinement via deep texture rendering. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3303–3312, 2021.

[23] Wadim Kehl, Fabian Manhardt, Federico Tombari, Slobodan Ilic, and Nassir Navab. Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again. *Proc. of the IEEE International Conference on Computer Vision (ICCV)*, pages 1530–1538, 2017.

[24] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[25] Yoshinori Konishi, Kosuke Hattori, and Manabu Hashimoto. Real-time 6D object pose estimation on CPU. *arXiv preprint arXiv:1811.08588*, 2018.

[26] Alexander Krull, Eric Brachmann, Frank Michel, Michael Ying Yang, Stefan Gumhold, and Carsten Rother. Learning analysis-by-synthesis for 6d pose estimation in rgb-d images. In *Proc. of the IEEE International Conference on Computer Vision (ICCV)*, pages 954–962, 2015.

[27] Yann Labbé, Justin Carpentier, Mathieu Aubry, and Josef Sivic. Cosypose: Consistent multi-view multi-object 6d pose estimation. In *European Conference on Computer Vision*, pages 574–591. Springer, 2020.

[28] Chi Li, Jin Bai, and Gregory D Hager. A unified framework for multi-view multi-class object pose estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 254–269, 2018.

[29] Wenhao Li, Hong Liu, Hao Tang, Pichao Wang, and Luc Van Gool. Mhformer: Multi-hypothesis transformer for 3d human pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13147–13156, 2022.

[30] Yi Li, Gu Wang, Xiangyang Ji, Yu Xiang, and Dieter Fox. DeepIm: Deep iterative matching for 6d pose estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 683–698, 2018.

[31] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.

[32] Ze Liu, Jia Ning, Yue Cao, Yixuan Wei, Zheng Zhang, Stephen Lin, and Han Hu. Video swin transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3202–3211, 2022.

[33] Wufei Ma, Angtian Wang, Alan Yuille, and Adam Kortylewski. Robust category-level 6d pose estimation with coarse-to-fine rendering of neural features. *arXiv preprint arXiv:2209.05624*, 2022.

[34] Weian Mao, Yongtao Ge, Chunhua Shen, Zhi Tian, Xinlong Wang, and Zhibin Wang. Tfpose: Direct human pose estimation with transformers. *arXiv preprint arXiv:2103.15320*, 2021.

[35] Eric Marchand, Hideaki Uchiyama, and Fabien Spindler. Pose estimation for augmented reality: a hands-on survey. *IEEE transactions on visualization and computer graphics*, 22(12):2633–2651, 2015.

[36] Paschalis Panteleris and Antonis Argyros. Pe-former: Pose estimation transformer. In *International Conference on Pattern Recognition and Artificial Intelligence*, pages 3–14. Springer, 2022.

[37] Jaewoo Park and Nam Ik Cho. Dprost: 6-dof object pose estimation using space carving and dynamic projective spatial transformer. *arXiv preprint arXiv:2112.08775*, 2021.

[38] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in PyTorch. In *NIPS Autodiff Workshop*, 2017.

[39] Georgios Pavlakos, Xiaowei Zhou, Aaron Chan, Konstantinos G Derpanis, and Kostas Daniilidis. 6-dof object pose from semantic keypoints. In *2017 IEEE international conference on robotics and automation (ICRA)*, pages 2011–2018. IEEE, 2017.

[40] Sida Peng, Yuan Liu, Qixing Huang, Xiaowei Zhou, and Hujun Bao. Pvnet: Pixel-wise voting network for 6dof pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4561–4570, 2019.

[41] Mahdi Rad and Vincent Lepetit. Bb8: A scalable, accurate, robust to partial occlusion method for predicting the 3d poses of challenging objects without using depth. *Proc. of the IEEE International Conference on Computer Vision (ICCV)*, pages 3848–3856, 2017.

[42] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[43] Fred Rothganger, Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. 3d object modeling and recognition using local affine-invariant image descriptors and multi-view spatial constraints. *International journal of computer vision*, 66(3):231–259, 2006.

[44] Ashutosh Saxena, Justin Driemeyer, and Andrew Y Ng. Robotic grasping of novel objects using vision. *The International Journal of Robotics Research (IJRR)*, 27(2):157–173, 2008.

[45] Chen Song, Jiaru Song, and Qixing Huang. Hybridpose: 6d object pose estimation under hybrid representations, 2020.

[46] Jiaming Sun, Zihao Wang, Siyu Zhang, Xingyi He, Hongcheng Zhao, Guofeng Zhang, and Xiaowei Zhou. Onepose: One-shot object pose estimation without cad models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6825–6834, 2022.

[47] Bugra Tekin, Sudipta N Sinha, and Pascal Fua. Real-time seamless single shot 6d object pose prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 292–301, 2018.

[48] Jonathan Tremblay, Thang To, Balakumar Sundaralingam, Yu Xiang, Dieter Fox, and Stanley T. Birchfield. Deep object pose estimation for semantic robotic grasping of household objects. In *Conference on Robot Learning (CoRL)*, 2018.

[49] Shubham Tulsiani and Jitendra Malik. Viewpoints and keypoints. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1510–1519, 2015.

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

[51] Chen Wang, Danfei Xu, Yuke Zhu, Roberto Martín-Martín, Cewu Lu, Li Fei-Fei, and Silvio Savarese. Densefusion: 6d object pose estimation by iterative dense fusion. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[52] Gu Wang, Fabian Manhardt, Federico Tombari, and Xiangyang Ji. Gdr-net: Geometry-guided direct regression network for monocular 6d object pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16611–16621, 2021.

[53] He Wang, Srinath Sridhar, Jingwei Huang, Julien Valentin, Shuran Song, and Leonidas J Guibas. Normalized object coordinate space for category-level 6d object pose and size estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2642–2651, 2019.

[54] Bowen Wen and Kostas Bekris. Bundletrack: 6d pose tracking for novel objects without instance or category-level 3d models. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8067–8074. IEEE, 2021.

[55] Bowen Wen, Chaitanya Mitash, Baozhang Ren, and Kostas E Bekris. se (3)-tracknet: Data-driven 6d pose tracking by calibrating image residuals in synthetic domains. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 10367–10373. IEEE, 2020.

[56] Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

[57] Lin Yen-Chen, Pete Florence, Jonathan T Barron, Alberto Rodriguez, Phillip Isola, and Tsung-Yi Lin. inerf: Inverting neural radiance fields for pose estimation. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1323–1330. IEEE, 2021.

[58] Fei Yu, Jiji Tang, Weichong Yin, Yu Sun, Hao Tian, Hua Wu, and Haifeng Wang. Ernie-vil: Knowledge enhanced vision-language representations through scene graph. *arXiv preprint arXiv:2006.16934*, 2020.

[59] Sergey Zakharov, Ivan Shugurov, and Slobodan Ilic. Dpod: 6d pose object detector and refiner. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1941–1950, 2019.

[60] Shuaitao Zhao, Kun Liu, Yuhang Huang, Qian Bao, Dan Zeng, and Wu Liu. Dpit: Dual-pipeline integrated transformer for human pose estimation. *arXiv preprint arXiv:2209.02431*, 2022.

[61] Ce Zheng, Sijie Zhu, Matias Mendieta, Taojiannan Yang, Chen Chen, and Zhengming Ding. 3d human pose estimation with spatial and temporal transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11656–11665, 2021.

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

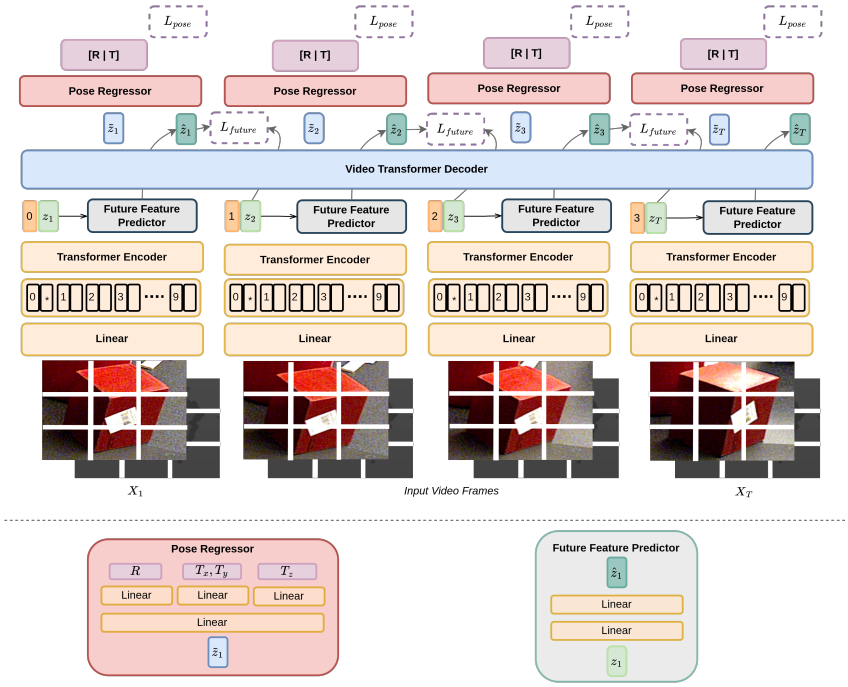

Figure 6: Overview of our VideoPose framework for 6D object pose estimation baseline. We use the same encoder as in [56].

A Appendix

A.1 Code Release:

Training code and pretrained models are available at https://github.com/ApoorvaBeedu/VideoPose.

A.2 Baseline implementation:

As mentioned in Section 4, to establish a baseline for video related pose estimation, we implemented a ConvGRU based temporal encoder as our baseline. In order to improve performance and match the pipeline of PoseCNN, several modifications were made to the baseline framework that does not exist in our Transformer based model. The overview diagram for ConvGRU based implementation can be seen in Figure 6. For the baseline, we train a depth decoder, and concatenate the features with the image features before feeding it into the ConvGRU. The $[\boldsymbol{R}|\boldsymbol{T}]$ regressor is the same as in the main framework.

A.2.1 Baseline Training Statergy:

In addition to the losses defined in Section 3.4, two additional losses were used in the training of ConvGRU baseline. We use the L_1 loss to learn depth (L_{depth}), and cross entropy loss for semantic segmentation (L_{label}). The total loss can be defined as:

$$L(\widetilde{\mathbf{q}}, \mathbf{q}, \widetilde{\mathbf{t}}, \mathbf{t}) = L_{\text{depth}} + L_{\text{label}} + L_{\text{pose}} + L_{\text{reg}} + L_{\text{inner\_prod}}. \tag{10}$$