

# 6DoF Pose Estimation with DETR-based transformer model

## Group 4

r11922a18 資工碩一 洪彥揚

r11922a13 資工碩一 王安

r11922a10 資工碩一 林昱辰

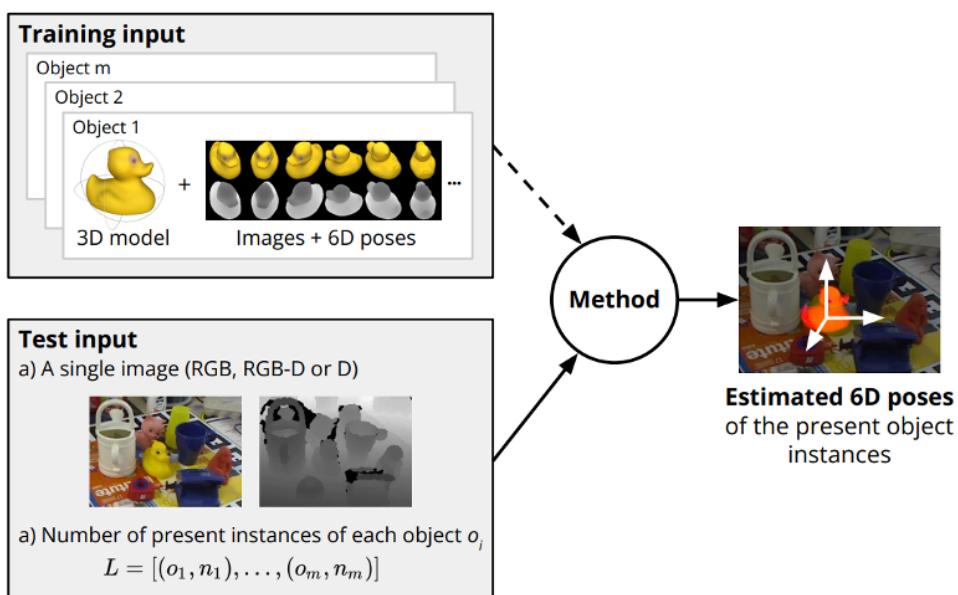
## I. Motivation

物體位姿估測(Object Pose Estimation)一直是許多應用任務的先決條件，主要常見任務有自駕車、自動化工廠機械手臂和智慧醫療手術等等，在自駕車任務中，估測行人與鄰近車輛的距離與位置是首要任務，進而達成閃避與安全移動等操作。

而在自動化工廠機械手臂中，如何定位產線上的零件以使機械手臂精準抓取是首要目標，其中手臂與零件的相對距離、零件的姿態抑或是重疊的零件如何被精準預估，皆是應用中影響效能的因素，也因此，如何透過模型精準預估待測物體的旋轉與平移，便是目前姿態預估中的重要課題。

## II. Problem Definition

本次專案的問題定義為如何透過訓練讓模型正確輸出物體的旋轉與平移。我們將會採用 BOP Challenge 2022 中所提供的資料集形式，其資料集分為訓練以及測試資料集，訓練集部分會提供 3D 物品模型、合成訓練影像、相機內部參數以及 ground truth 標註，其中 3D 物體模型是手動創建或使用類似 KinectFusion 的系統進行 3D 表面重建，合成訓練影像則由 3D 物體模型渲染得到。測試集部分則提供統一格式的圖片以供評估，並會依照場景複雜度進行分級，場景內通常有雜物和遮擋。



### III. Methodology

#### A. Model Introduction

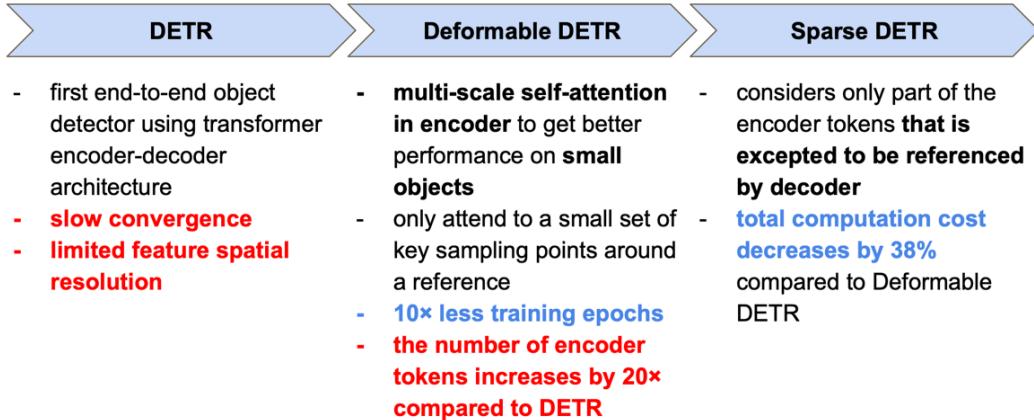


Fig.1: Advantages and disadvantages between different DETR-based model.

首先 **DETR**, 是第一個 transformer-based 的 object detector, 由於 backbone 為 Vision-transformer, 其優缺點也就繼承了 transformer-based model, 雖然精確度能比起 CNN-based model 更高，但訓練時間也較為冗長，因此最致命的缺點就是收斂時間過長；除此之外，因為 DETR 的 self-attention 都採用單一且固定的 scale，因此在處理小物體的偵測上會較差。

為了解決以上兩個缺點，後續出現了 **Deformable DETR**，其最大的改進在於改採用 **”Multi-scale self-attention”**，使其在遇到小物體的偵測任務時也能順利偵測到。此外，在收斂時間這項致命缺點中，也提出了一項改進，變成只會去參考當下關注的重點周遭的 key sampling point，而不會像最原始的 ViT 一樣去參考全局的資訊，如此一來也能省去一些不必要的資訊與學習時間，並維持一定的 performance。這樣的改進使 Deformable DETR 比起 DETR 在訓練時間上只需要 1/10 倍的 epochs 就能收斂。

但是 Deformable DETR 在 multi-scale 的改進後為了觀察到不同 scale 的資訊，反而讓 encoder token 的數目增加了 20 倍，因此 ”Sparse DETR” 特別針對這點去做了改善。

**Sparse DETR** 是以上三種方法中最新發表的 model，其主要改進就是在 token 數量上的精簡，他們發現其實在做 object detection 時並不是所有 tokens 都是必須的，因此設計出這樣的機制，讓 model 在 self-attention 的部分只會去參考到有可能在 decoder 會被利用到的 encoder tokens，以此大幅降低了整個訓練過程中用到的 token 數量，並節省了 38% 的計算成本。

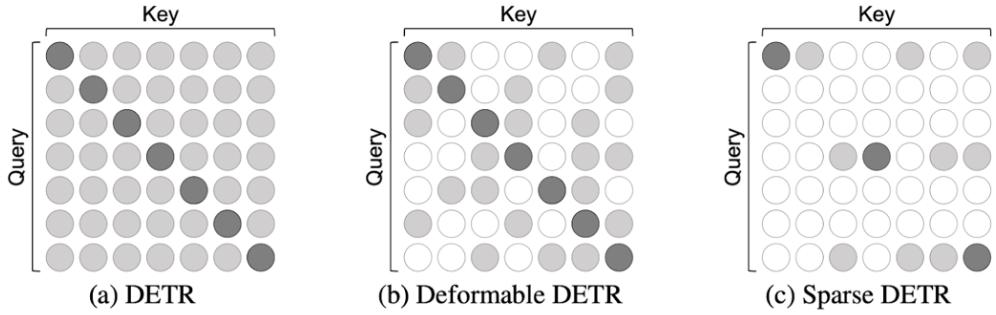


Fig.2: Attention complexity. The circles in the square matrix represent the attention between keys and queries. The gray/white circles correspond to preserved/removed connection respectively, and darker gray on the diagonal positions means where the token attends to itself.

因此三種 DETR 的差別就在於 self-attention 的 scale 問題，以及 encoder reference 到的 token 數量差異，如 Fig.2 為 Sparse DETR 論文中比較這三種方法 attention 到的 token 數量之間的差異，其中深色的圈圈代表關注程度較高的 tokens，可見 Sparse DETR 所需的 token 數量是大幅減少的。

而 Table.1 的實驗數據可見 Sparse DETR 依然能達到高水準甚至更好的精確度，但最大的優點是收斂速度的大幅上升，以避免冗長的訓練過程。

Method	Epochs	AP	AP_50	AP_75
<b>DETR</b>	500	42.0	62.4	44.2
<b>Deformable-DETR</b>	50	46.0	65.2	49.8
<b>Sparse-DETR</b>	50	46.3	66.0	50.1

Table.1: Performance on Object Detection task of 3 models on COCO 2017 validation set.

## B. Modified Architecture: Pose Estimation

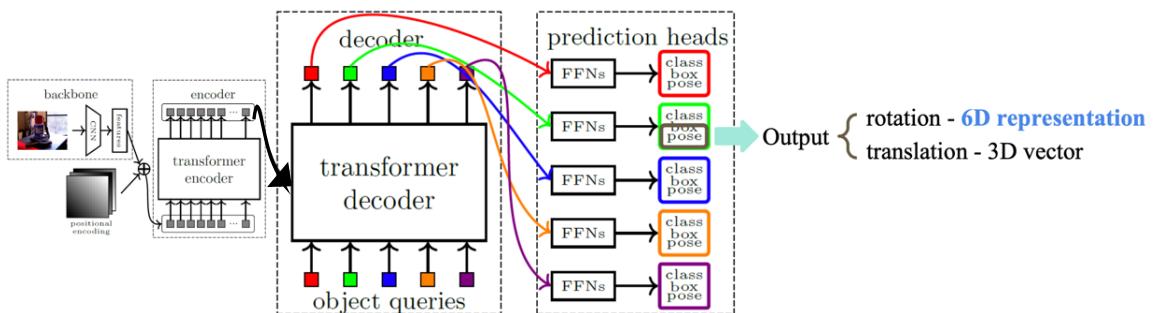


Fig.3: Modified architecture of our method.

對於此類 Single-stage 的 6D 位姿估計方法，可以通過直接修改 Object Detection 的模型，除了預測影像上面的檢測框外也一併輸出物體的類別與位姿。為了由相片自動達成 single-stage 的姿態預估，我們將基於 Transformer 之影像物體檢測網路 DETR 進行改良，並透過上述 CNN features extractor，對

於輸入的影像做特徵提取。當特徵提取完成後，即輸出關於此影像的描述，接著我們將設計 Transformer 中的 encoder 與 decoder，encoder 的輸出會提供給 decoder 作為輸入，並配合位置編碼一起使用。我們希望設計出與使用固定正弦位置編碼的編碼器不同的方法，並向 decoder 提供學習後的位置編碼。Decoder 的每個輸出都由不同的 shared prediction heads 以平行方式獨立處理，最後生成一組包含類別概率、Bbox 和 6D 物體姿態的平移和旋轉的 N 元組。本次任務將針對三種 DETR-based 模型架構做修改。

### C. Losses

$$L_t = \|\mathbf{t} - \tilde{\mathbf{t}}\|_2 . \quad \text{L2 loss}$$

$$L_{rot} = \arccos \frac{1}{2} \left( Tr \left( R \tilde{R}^T \right) - 1 \right) \quad \rightarrow \text{total loss: } w1 * \text{class loss} + w2 * \text{bbox loss} \\ + w3 * \text{rotation loss} + w4 * \text{translation loss}$$

Geodesic Distance between the predicted and target rotation

Fig.4: Loss of our method.

### D. 6D Rotation Representation

針對 rotation 的表示，我們參考了 GDR-Net 這篇論文，裡面提到根據實驗，rotation matrix 與模型所預測的旋轉表示間，應該要存在可微分且連續的 mapping 方式，而以往將 rotation matrix 轉換成 quaternion 的函式在某些地方是不可微分的，會影響模型學習，因此在 single-stage 的姿態預估中，6D 是比較好的表示方式。

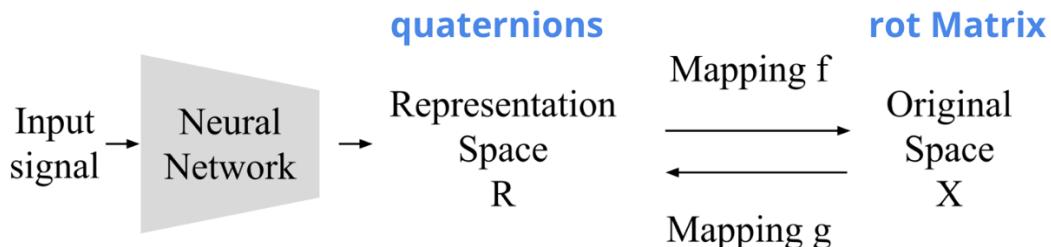


Fig.5: Modified architecture of our method.

## IV. Dataset: YCB-Video

YCB-Video dataset 包含了 92 部影片，共 133,827 個 frames，影片中共計會出現以下 21 種物體，分別是 master\_chef\_can, cracker\_box, sugar\_box, tomato\_soup\_can, mustard\_bottle, tuna\_fish\_can, pudding\_box, gelatin\_box, potted\_meat\_can, banana, pitcher\_base, bleach\_cleanser, bowl, mug, power\_drill, wood\_block,

`scissors`, `large_marker`, `large_clamp`, `extra_large_clamp`, `foam_brick`，其中 `bowl`, `wood_block`, `large_clamp`, `extra_large_clamp`, `foam_brick` 為「對稱」物體，其餘為「非對稱」物體，此差異會影響計算 loss 的方式。

我們最終使用的為 BOP challenge 提供的 ycbv BOP dataset，其中，我們使用 BlenderProc4BOP 進行模擬資料的合成。具體過程如下：

1. 在空房間中安排選定的 BOP 數據集中的物體，並使用其他 BOP 數據集中的物體作為干擾物。
2. 為了豐富生成影像的材質，BlenderProc4BOP 將 CC0 Textures 庫中的隨機 PBR 材料分配給房間的牆壁，並從房間的天花板和隨機定位的照明源發出隨機強度和顏色的光。
3. 物體材質不是為了完美的模擬真實情況，而是隨機化的。通過在 Blender 中集成的 PyBullet 物理引擎將物體放在地面上，可以實現真實的物體姿勢，並有輕微遮擋的效果。

## V. Performance

### A. Metrics

#### a). ADD (average distance)

此指標的計算方法為針對 3D 點雲集合中的每一個點，分別利用 ground truth 的 R, T 和預測出來的 R, T 將其旋轉至空間中正確及預測的位置，並且計算這兩個位置的距離。具體公式如下：

$$\text{ADD} = \frac{1}{|\mathcal{M}|} \sum_{x \in \mathcal{M}} \|(Rx + t) - (\hat{R}x + \hat{t})\|$$

#### b). ADD-S (average closest pairwise distance)

此指標為 ADD metric 的變形，針對同一堆點雲中的每個點，我們去計算其與 ground truth 點雲中最近點的距離，具體公式如下：

$$\text{ADD-S} = \frac{1}{|\mathcal{M}|} \sum_{x_1 \in \mathcal{M}} \min_{x_2 \in \mathcal{M}} \|(Rx_1 + t) - (\hat{R}x_2 + \hat{t})\|$$

此 metric 相對於 ADD 來說較為寬鬆，主要是為了讓對稱物體的 ground truth 可以有多種不同的角度，例如一個對稱的碗水平放置桌面，無論預測出來的點雲水平旋轉幾度都應被視為是預測正確。

## B. Comparison

- ◆ ADD(-S)：針對「非對稱」物體計算其 ADD；  
針對「對稱」物體計算其 ADD-S。
- ◆ ADD-S：針對所有物體計算其 ADD-S。
- ◆ AUC of ADD-S/ADD(-S)：根據計算出來的 distance，在 0.1m 的距離內，設定不同的 threshold 來計算 precision 和 recall，畫出 ROC curve 並計算其線下面積。
- ◆ Parameters Setting：  
weight of loss：  
    class loss: 1  
    Bbox loss: 1  
    rotation loss: 1  
    translation loss: 1  
optimizer: AdamW  
learning rate: 2e-4  
batch size: 12 (T6D-Direct) / 8 (Deformable-DETR) / 16 (Sparse-DETR)  
由於模型大小及電腦差異，model batch size 的大小略有差異，主要以 memory 的容量為考量並盡量設大。

下表為各模型的結果：

Method	AUC of ADD-S	AUC of ADD(-S)	Training time (per epoch)	Epoch	num of epoch that paper claims to converge	num of parameters
T6D-Direct (DETR)	86.2	74.6	-	-	300	-
T6D-Direct (DETR) (reproduce)	71.83	43.64	20 min. (RTX-3090)	300	300	41526819
Deformable DETR	69.61	41.94	1hr 5min. (RTX-2080Ti)	100	50	39946533
Sparse DETR	<b>76.45</b>	<b>55.48</b>	25 min. (RTX-3090)	80	50	39946533

Table.2: Comparison between different models.

下表中我們再針對稱及非對稱物體分別計算其 ADD-S 和 ADD(-S) 的平均 AUC，表格中斜線前面為非對稱物體，後面為對稱物體。

Method	AUC of ADD-S	AUC of ADD(-S)
T6D-Direct (DETR)	87.81/81.04	72.59/81.04
T6D-Direct (DETR) (reproduce)	72.12/70.88	35.13/70.88
Deformable DETR	69.84/68.87	33.52/68.87
Sparse DETR	76.61/75.94	49.09/75.94

Table.3: Comparison between symmetric and asymmetric object on different models.

## VI. Discussion And Observation

### ◆ 針對 Table.2 :

#### 1. model-wise comparison

可以看到 T6D 的 reproduce 結果距離 paper 所稱還有一段距離，其中 ADD-S 此項指標又比 ADD(-S) 好一點。而 Deformable 的表現不如預期，甚至比 T6D reproduce 的結果還差一點。實作下來表現最好的是 Sparse DETR 這個 model，不僅收斂速度最快，最終表現也最好，可見將原始 DETR 的 backbone 換成 Sparse DETR 後，對於訓練 pose estimation 任務是有幫助的。

#### 2. training time

這邊可以看到我們 Deformable 和 Sparse DETR 都 train 超過了 paper 所說會 converge 的 epoch 數量，並且我們認為這兩個 model 都還在持續的收斂，因此可以觀察到加上 pose estimation 的任務，原本 DETR 的 model 要學習的目標更多，訓練時間也相應的拉長。

#### 3. number of parameters

在我們的實做下，可以看到 T6D 相較於另外兩個 model 的參數量是更多的，某種程度上也可以說明為什麼 T6D 收斂所需的 epoch 數最多。

#### 4. converge speed of rotation & translation

在訓練過程中我們觀察到，translation 相較於 rotation 更容易收斂，收斂速度也較快，推測可能原因為預測 Bbox 座標時也會輔助 translation 的預測。

- ◆ 針對 Table.3 :

可以看到無論是哪一個模型，對稱物體的訓練都是較佳的，而相對的非對稱物體的 ADD 都比其 ADD-S 低上許多，表示非對稱物體可能僅有 translation 的部分有學好，而 rotation 並沒有被良好的學習到，因此往後也可以針對非對稱物體的旋轉做更進一步的訓練與嘗試。

## VII. Visualization

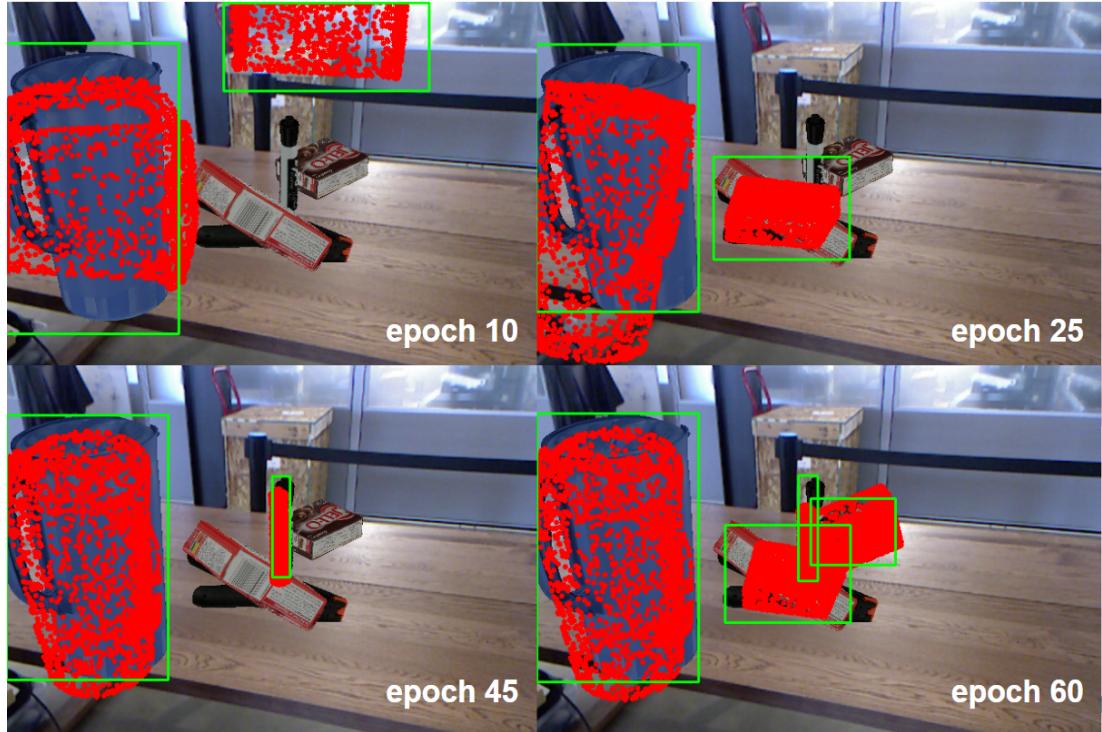
A. 我們使用這張 **test image** 針對三個模型下去做視覺化的呈現如下：

Test Image	T6D
Deformable DETR	Sparse DETR

圖中綠色的框線為 **bbox**，超過一定信心水準的 **bbox** 才會被繪製出來；紅色部分則為點雲，由這三張圖比較可以看到，很明顯 Deformable rotation 的表現是最差的，Sparse DETR 表現最好，符合數據上的結果，同時 **bbox** 偵測的部分也是 Sparse 表現最佳。

B. 針對 **Sparse DETR** 我們挑選了訓練過程中的四個 epoch，將其結果視覺化：

可以看到藍色水壺很明顯從水平的方向漸漸轉向垂直方向，過程中 Bbox 也從原本預測錯誤到後面越來越多正確的物品被偵測出來。



## VIII. Other Experiments

### i. rotation representation :

一開始我們嘗試讓模型輸出 quaternion 形式的 rotation，轉換成 rotation matrix 後再與 ground truth 計算 loss，但實作下來可以發現，在訓練很初期 rotation loss 即不再下降，後來參考 "GDR-Net" 了解 quaternion 和 rotation matrix 在轉換過程中可能產生不連續的情況，因此決定遵照 paper 指示將模型輸出換成 6D representation 的形式，成功讓 rotation loss 穩定收斂。

### ii. rotation weight :

有鑑於訓練過程中觀察到 rotation 相較於其他 loss 更難被訓練，我們嘗試過一開始就將 rotation weight 調高，但我們發現若在訓練初期就把 rotation weight 設太高有可能會使得 Bbox 和 class 的部分沒有學好，因此權衡之下我們針對 Sparse-DETR 設計兩種 setting：

- (1). setting1 是讓 class / Bbox / rotation / translation weight 皆維持 1。
- (2). setting2 是在 training 至 45 個 epoch 後將 rotation weight 調高至 2，並且比較這兩種 setting 訓練 80 個 epoch 後的效果，如 Table4。

Method	AUC of ADD-S	AUC of ADD(-S)	Epoch
Setting1	76.45	55.48	80
Setting2	75.46	54.36	80

Table.4: Comparison between 2 settings on Sparse-DETR.

可以看到無論是在哪個 metrics 下，setting1 表現更佳，可見直接的加大 rotation loss 的 weight，對於 pose estimation task 的表現不一定有直接的幫助，還有待更進一步的實驗。

## IX. Difficulties

### I. 訓練時長過長

雖然本次實作 Deformable 及 Sparse DETR 模型本身就是為了解決原始 DETR 訓練時長過長的問題，但實作的過程中發現加上 pose estimation 問題，它訓練時間就又被拉長，因此過程中做的各種嘗試都需要花費以天為計算的訓練時間才可以看出差別或是否有效果，是本次遇到的最大挑戰。

### II. 模型需要 optimize 的方向過多

Object detection 模型本身需要優化的任務就包含了物體類別的辨認、物體 Bbox 座標，加上 pose estimation 的 rotation 和 translation，總計就有四項 loss 需要設定各自的 weight，在訓練過程中我們也有發現若是沒有設定好，可能造成 translation error 收斂非常快，但 rotation loss 卡在一個點上下擺盪，又或者是過度調高 rotation & translation loss，造成 class 辨認不佳，從而影響後續點雲的優化，這部分也有待更進一步的實驗及嘗試來找出最佳組合。

## X. Conclusion

本次 final project 不同於以往 two-stage 的傳統方法，我們嘗試了三種 DETR-based 的 transformer model，讓模型 end-to-end 的輸出 rotation 跟 translation，實做 single-stage 的 pose estimation。

實做下來發現 Deformable DETR 和 Sparse DETR 的收斂速度的確較原始 T6D 快，但其中 Deformable 的表現較差，而 Sparse DETR 表現最好，同時我們也在過程中驗證了 6D rotation representation 相較於 quaternion 的表示方式會有更好的效果。

## XI. Work Division

- ◆ model implementation
  - T6D : 洪彥揚
  - Deformable DETR : 王安
  - Sparse DETR : 林昱辰
- ◆ oral presentation : 洪彥揚、林昱辰
- ◆ video editing : 王安
- ◆ report, slides : 洪彥揚、王安、林昱辰

## XII. Reference

- (i). T6D-Direct: Transformers for Multi-Object 6D Pose Direct Regression  
<https://arxiv.org/abs/2109.10948>
- (ii). DETR: End-to-End Object Detection with Transformers  
<https://arxiv.org/abs/2005.12872>
- (iii). Deformable DETR: Deformable Transformers for End-to-End Object Detection  
<https://arxiv.org/abs/2010.04159>
- (iv). Sparse DETR: Efficient End-to-End Object Detection with Learnable Sparsity  
<https://arxiv.org/abs/2111.14330>
- (v). GDR-Net: Geometry-Guided Direct Regression Network for Monocular 6D Object Pose Estimation  
<https://arxiv.org/abs/2102.12145>