7月 進捗確認会報告資料
Augumented　Autencoder を用いた姿勢推定の調査

安井　理

2020 年 8 月 9 日

1　はじめに

前回の進捗報告会までに調べてきた,AR マーカの座標推定のための SSD-6D 論文が, 実装を行うことが難しいという事が分かったため, 別の方法を考えることにした.
今回調べてきた論文も前回同様 6 次元の姿勢推定を行うための手法であるが, 方法としては前回調べてきた論文とは違い検出と推定を分けて考えられた手法である. 調べてきた論文では, 実際にコードがすべて Git-hub に記載されており, 実装することが可能なので, この論文を軸に引き継ぎをした鈴木さんの研究と合わせて自分の研究の目標を明確にし進めることにした. 最終的な研究目標としては鈴木さんの研究 (Faster-RCNN) を用いた検出と組み合わせて座標推定を実現できるようにしていこうと考えている.

2　進捗報告

2.1　調査内容

・implicit 3d orientation learning for 6d object detection from rgb images の論文調査

・Augumented　Autencoder の手法理解

・検出推定の手法理解

・実装に向けた環境作り

今回調査した手法及び論文は以下である.

・implicit 3d orientation learning for 6d object detection from rgb images

・Domain Randomization

2.2　今回扱う 6 次元推定の概要

implicit 3d orientation learning for 6d object detection from rgb images の論文は,ECCV2018 の Best paper に選ばれた 6 次元物体検出の論文.

6D 物体検出は,3 次元空間座標だけでなく 3 方向の向き姿勢情報も含んだ検出問題であり, 高速に推定を行えることに加えて,6D のラベル付き教師データがなくても学習可能な手法である.6D ラベル付き教師データの代わりに、検出対象となる物体の 3D CAD データが必要となる. 全体の処理の流れ図 1 としては, まず入力となる RGB 画像に対して SSD を用いて対象物体の Bounding Box を推定, その後, 推定された Bounding Box 領域から物体の姿勢情報を推定するという処理を行う, 後半の Bounding Box 領域から物体の姿勢情報を推定する部分が今回自分の研究に取り入れようと考えている方法で,Augmented AutoEncoder(以降 AAE) という手法が用いられている.

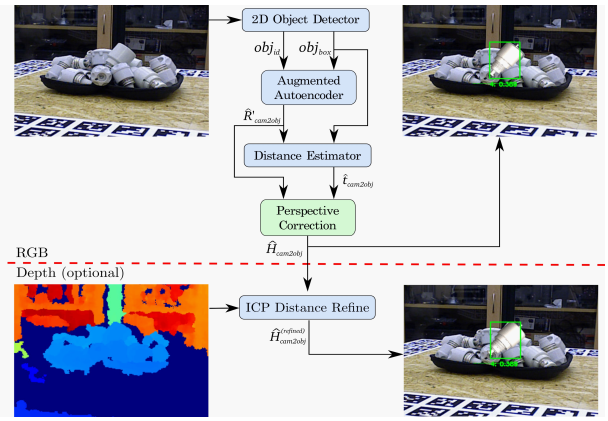

図 1: 全体の流れ.

2.3　トレーニング

あらかじめ Augumented　Autencoder（AAE）を使い検出したい物体の 3 D モデルを作成し様々な視点でトレーニングするトレーニングの流れを図 2 に示す. まず等間隔に 3 D モデル物体を回転させ, すべての角度の視点（2562 点）と物体の各回転座標のコード. 一つの視点につき 36 点を用意 ,AAE にかけ潜在変数 z ∈ R128 を生成して（ zi , Ri）を一つの物体 ID に 92232 個記録していく.

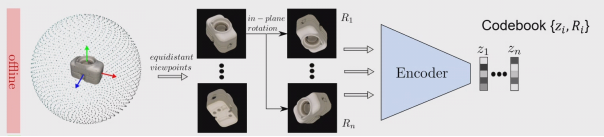

図 2: トレーニングの全体像.

2.3.1 Domain Randomization

今回扱う AAE には Domain Randomization という考え方が取り入れられている. この手法の目的は, テスト時にモデルが実世界のデータに一般化できるよう, 環境ノイズを加えた 3 D モデルをランダムに作成しトレーニングを行う. これにより図 3 に示すように物体と背景の対称性を明確にして, テスト時に現実でも推定可能になる.

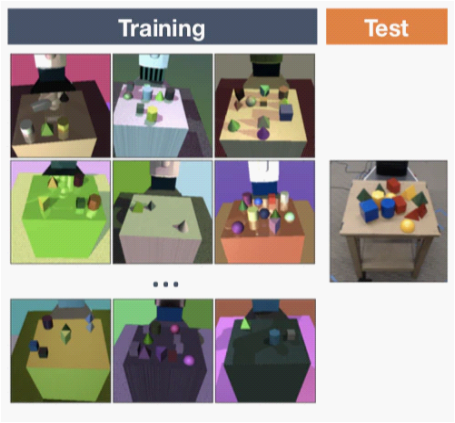

図 3: Domain Randomization.

2.3.2 Autencoder

今回学習に使われる手法の基礎となるのが Autencoder 図 4 である, 簡単な説明はオートエンコーダーとは、教師なし学習の一つであり, 学習時の入力データは訓練データのみで教師データは利用しない, データを表現する特徴を獲得するためのニューラルネットワークである.Autencoder を表

す数式を式 1 に示す. 要素はそれぞれ　x: 入力（インプット）　x ’: アウトプット　ϕ: エンコーダー　ψ: デコーダー z ：潜在変数　この式のロス関数は式 2 に示す.

$$(x') = (\psi * \phi)(x) = \psi(z) \tag{1}$$

$$l_2 = \sum_{i \in D} \parallel x - x' \parallel_2 \tag{2}$$

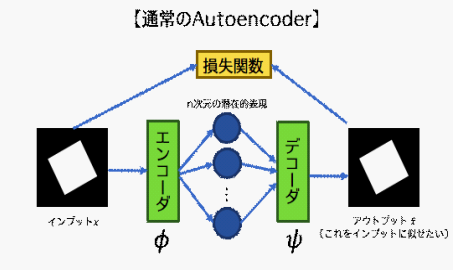

図 4: Autencoder.

2.3.3　Denoising Autoencoder

Autoencoder を発展させたノイズ除去方法が Denoising Autencoder である.Autoencoder にノイズありの画像を入力し、ノイズなしの画像が出力できるように学習させることで, ノイズによらない本質的な潜在表現を得ることができる方法である.

2.3.4　Augumented Autoencoder

今回姿勢推定を行う手法で,Denoising Autoencoder,Domain Randomization の手法を取り入れ応用した Autencoder. ノイズだけではなく背景や光、遮蔽物などの環境ノイズを追加した画像を入力し, 環境ノイズなしの画像を出力できるように学習させる手法. 図 5
Augumented Autencoder を表す数式は, 式 3 である. 要素はそれぞれ x:　入力（インプット）　x ’: アウトプット　　：変換（加工）の関数　x ’: インプット画像を変換（加工）したもの　ϕ : エンコーダー　ψ: デコーダー　z:潜在変数　この式のロス関数は式 2 と同じである.

$$x' = (\psi * \phi * f_{augm})(x) = (\psi * \phi)(x'') = \psi(z) \tag{3}$$

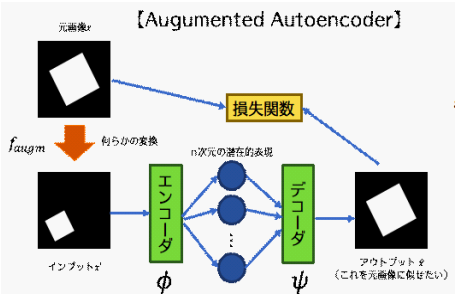

図 5: Augumented Autoencoder.

2.3.5 AE と AAE の比較

四角形の画像を用いた, 通常の Autencoder と Augumented Autencoder の精度の比較を図 5 に表す. （a）をベース画像として 1 : （a）→（a） 2: (d) →（d） 3: (d) → (a) の入出力で学習. 結果として通常の Autencoder を使用している 1 ・ 2 にはあまり応用が利かず若干のノイズが入ると潜在表現を反映できないが（ 3 ）の AAE を用いたものは高精度に潜在表現を表すことができている.

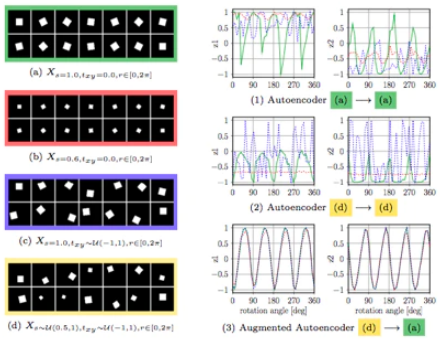

図 6: ,AE と AAE の比較.

2.4 テスト

方向が既知の潜在表現と、推定したい画像から新たに得た潜在表現を比較することで, 方向の推定を行う. まず, 事前に学習した 3D データから 92232 通りの向きの画像を作成し, それらをエンコーダに入力して学習し, 潜在表現を取得する. 推定時に,SSD などの物体検出器によって切り取られ

た物体画像をエンコーダに入力し潜在表現を取得, これを事前に取得した潜在表現とでコサイン類似度を測り, 最も類似していたものの方向が推定したい方向となる.n 次元のベクトル空間に画像を埋め込み, 最近傍法を使用.92232 通りというのは, 球面状に均一な 2562 箇所の点を考え, そこに物体を 36 通りの回転で配置した時の球の中心からの見え方を合わせた計 2562 × 36 の 92232 通りである. テストの流れを図 6 示す.

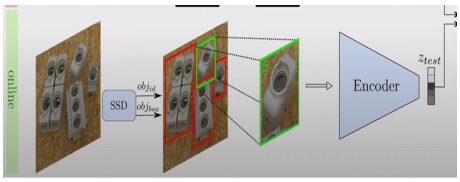

図 7: テスト.

2.4.1 コサイン類似度

コサイン類似度を求めるというのはトレーニングで取得し蓄積された潜在表現 z_i とテスト時に取得した潜在表現 z_{test} の類似度を求める計算である. 計算されたコサイン類似度の中から最も類似度の高い物（1 に近い）を k 近傍法によって決定する. コサイン類似度計算式は式 4 に示す.

$$cos_i = \frac{z_i * z_{test}}{|z_i||z_{test}|} \tag{4}$$

2.4.2 推定

推定はコサイン類似度によって求められた最も類似度の高かった訓練データの視点, 回転情報をそのままテスト時の物体の姿勢として値を返すことによって推定する.

2.5 動作確認

2.5.1 動作環境

・Linux, Python 2.7	・Tensorflow = 1.6	・OpenCV = 3.1	・tensorboard ==1.6.0
・PyOpenGL==3.1.4	・tensorflow-gpu==1.6.0	・tensorboard ==1.6.0	

2.5.2 セットアップ

・.bashrc にワークパスの設定の記述

```
AE_WORKSPACE_PATH=/path/to/autoencoder_ws
```

・ディレクトリの作成（ホームに作成）

```
mkdir    ae_workspace_path
```

・作成したディレクトリに移動しコマンドで入力

```
ae_init_workspace
```

・ディレクトリにファイルが生成される

2.5.3 トレーニング

・作成されたファイルに移動し ・ノイズを加えた画像の生成

```
ae_train exp_group /my_autoencoder -d
```

・別のターミナルでトレーニングを開始

```
ae_train exp_group / my_autoencoder
```

2.6 終わりに

今調べている手法をもとに卒業研究を進めていくことに決め, 方針が固まったのでゴールが明確になったという面でも先月の進捗報告会から進められていると思うので, 来月までにもっとペースを上げていきたいと思っている. 現在はまだ, トレーニングを行えておらずプログラムと PC と動作環境の互換性を見て GPU を使ったトレーニングを行えるようにしていかなければならない. 次の進捗報告会までには, プログラムを一通り動作確認を行いそのうえでどのように自分の卒業研究につなげていけるかを考えていけるところまで進めたい.

参考文献

[1] https://github.com/DLR-RM/AugmentedAutoencoder/

[2] https://arxiv.org/pdf/1902.01275v2.pdf/