自作 AI 物体検出による教室内距離測定

林 祐大　　坂田 萌泰
兵庫県立神戸高等学校総合理学科 2 年

コロナの三密対策を避けるために、ある程度の距離をとらなければならない。しかし、人間がそれを測るのには限界がある。それを自動的に人物間の距離を測定できるシステムを私たちは yolo v 5 を使って、頭部を検出する重みを作成し、人物間の距離を測定するシステムの完成を目指した。

1.初めに

1.1 動機

現在 AI 機器が様々な場で作動しており、生活の一部と言っても過言ではないだろう。私たちは、AI により、仕事を簡略化する恩恵を受けた。さらに、コロナ禍ということで、店頭に置かれている体温測定装置のように AI 機器を見る機会が増え、コロナ対策の手助けになっている。コロナ対策の一環で三密対策というものがあり、ある一定の距離を保たなければならず、その距離を人事的に測るには限界がある。そこで、カメラの画像を読み込み、画像内で座標系を作ることで、座標としての距離を表すことができる。さらに、その端点となる人間を物体検出する AI 機能を使うことで、人物間距離測定システムが人事作業を大幅に減らすことができると考え、このシステム製作を行った。

1.2.システム手順

1.天井に設置したカメラで上部から教室内の写真を撮影する。

2.1.で撮影した画像から頭部を予め学習させた重みで物体検出する。

3.2. で検出した頭部の中心を端点とし、各頭部の座標を割り出す。

4.すべての頭部の座標から 2 点を取り出し、三平方の定理で座標距離を求める。これをすべての 2 点を取る組み合わせを行う。

5.予め、単位座標当たりの長さをもとめ、それに 4.の座標距離を掛け合わせ、人物間距離を出力する。

1.3 研究の決定事項

今回の研究で使用した物体検出の[1]アルゴリズムと研究の定義を説明する。

1.今回の研究で使用したアルゴリズムは[2]yolo v 5 である。

2.カメラを天井に設置し、教室内で上部から距離の測定を行う。使用したカメラは IO DATA の ts-wrFE である。画角は水平に 180°、垂直に 95.5°。

3.全員直立していること、身長は 170 ㎝である。

4.人間の身長を 170 ㎝、人間の頭部の大きさはないものと定義する。

1.4 本研究の目的

人間の頭部を学習させた重みを使い、静止画に写っている人を検出し、正確な距離を測る自動システムの製作を心掛けた。

2.実験 I

2.1 目的

今回、距離を測るために端点である頭部を検出する必要がある。そこで、物体検出アルゴリズムである yolov5 を使用した。また、サンプルの yolov5 は頭部の検出をすることができないため、自分で学習させた。そして、自作した yolov5 を使って、物体の正確さを表す指標[3]recall,precision,mAP の値を出し、自作した重みが適確に頭部を検出できるかを調べる。

2.2 頭部検出を行う yolov5 の学習の方法

方法は次の通りである。また、コード、パスの指定のテキストは 8.で説明する。

1. 天井に設置したカメラで人間の写真を 172 枚撮影する。
2. 写真 172 枚ずつに、[4]labelImgwo を使って、頭部に四角でラベルを付け、テキストデータとして保存する。。
3. [5]Googlecolab を開き、yolo v 5 に 1,2.を学習させる。

※教室内の画像を今回は class を 1 つにすると、エラーが起こったため、class に’head’と全く関 係のない’a’ をいれて学習させた。画像に’a’とラベルをした写真は 1 つもない。

2.3 実験方法

172 枚のうち、22 枚の写真をテストさせ、[6]tensorboad を使い、AP,mAP の計算を行う。また、

テスト用だけでなく、無作為な地点にいる人間に画像22 枚ほどの検出を行った。

2.4 結果

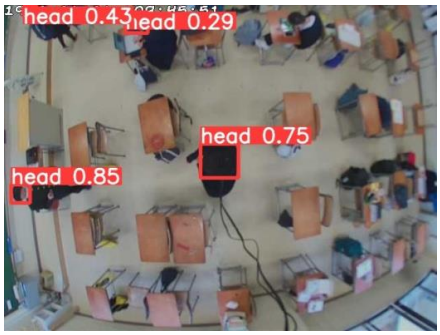


図 1 検出が比較的高い写真

図 2 FN である写真

図 3 FP である写真

図 4 'a'と認識してもの

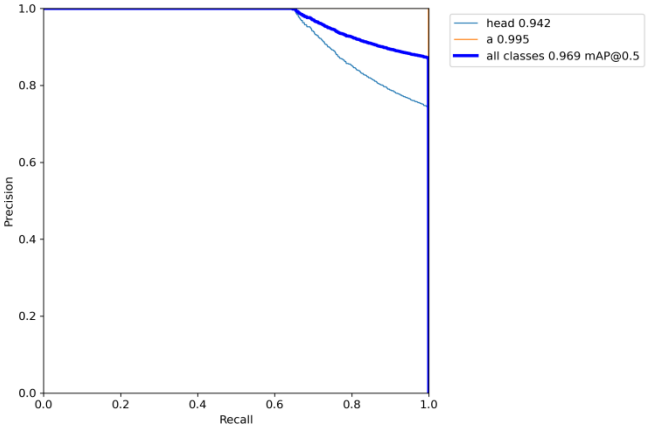
表 1 すべての画像の TP,FP,FN の合計

	合計
映っている人	89
TP	26
FP	3
TP+FP	29
FN	63
TP+FN	89

表 2 TP,FP,FN の合計に対する recall と precision

recall	0.292
precision	0.897

表 3 tensorboad が出した racall、precision の相関と mPA の算出結果



2.5 考察

AP は head0.942 と高い。'a'が 0.995 であり、mAP は 0.969 であった。検出した図 1,2,3 である。'a'と指定したラベルを付けていないのにも関わらず、'a'と認識している。しかし、図 4 より、'a'と認識しているものは全て人間の頭部であったことより、'a'は人間の頭部を認識している。図 3 より、頭部を誤検出している写真が 3 つあるが、表 2 より、precision の値から正確にとらえていることが分かる。しかし、表 2 の recall が低いように、写真図 1 と図 2 を比べると、同じようにカメラの真下にいるような写真でも、検出するものとしないものがあった。検出しなかった要因としては、髪と服がほぼ同一色であり、検出しにくい可能性が高い。

3.実験Ⅱ

3.1 目的

この実験は、ts-wrFE カメラで撮影した写真からどれだけ人物間距離を正確に測れるかを調べる。3.1 より頭部の検出する重みは完成したが、表 2 の recall が頭部をとらえきれていないことが多い。また、3.で完成した重みと yolo の画像内のボックスの座標を出力するプログラム出来なかった。そのため、labelImg を使い、手作業で、画像内の座標を計測した。そして、実際の長さと算出結果,誤差を調べた。

3.2 実験方法

1.図 6 のように、教室内の地面に名前を指定する。点 O は地面と垂直にとったカメラを通る直線と教室の地面との交点である。実際の長さ,は次の通りである。

OA=OF=OB=OI=1.00m,OH＝OK=3,00m

,CJ//HK//DL,AB⊥HK

2.labelImg を使い、目的としている頭部の中心点の座標を出す。原点は画像の左上の端,。

3.2.ででた 2 物体間の x 座標、y 座標の座標距離をし、単位当たりの座標を掛け合わせる。

4.3.の座標から三平方の定理を使って、距離を求める。あらかじめ AB の延長線上と HK の延長線上のカメラが映る最大の距離を端点とし、単位 1 座標あたりの長さを求めた。x 座標は 1 座標あたり 0.877m,y 座標は 1 座標あたり 0.547m である。つまり 2 点間の距離はこのようにして求める。

$$d = \sqrt{(0.877(\mathrm{Xa} - \mathrm{Xb}))^2 + (0.547(\mathrm{Ya} - \mathrm{Yb}))^2}$$

※2 つの物体を A,B とし、A の x 座標,y 座標,B の x 座標,y 座標を Xa,Ya,Xb,Yb とし、物体 AB の距離を D とする。

算出する地点間距離は縦方向である AB,EG,CD 横方向である IF,FH,斜め方向である OG,OC,JD を調べた。

図 5OC 計測時の画像

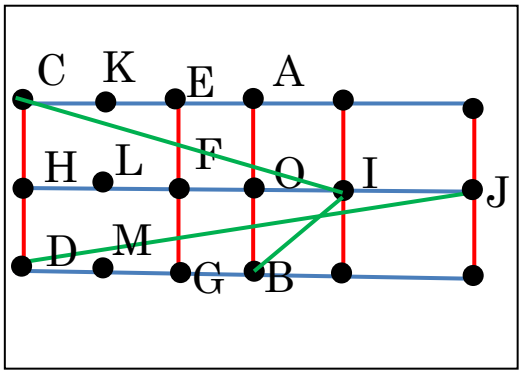


図 6 カメラから見た地点の図

3.3 結果

表 3　実際の距離と算出結果の値と実際の距離算出結果との誤差

方向の分類	地点	実際の距離(m)	算出結果(m)	誤差(m)
縦方向	AB	2.00	2.02	+0.02
	EG	2.00	1.96	−0.04
	CD	2.00	1.42	−0.68
横方向	IF	2.00	3.08	+1.08
	FH	2.00	1.07	−0.93
斜め方向	OG	1.41	1.65	+0.24
	OC	3.16	2.65	−0.51
	JD	6.08	4.98	−1.10

3.4 考察

画角が小さい縦方向では、AB,EG の誤差の値は小さかったため、AB から KM の地点まで 1 座標あたりの長さは変えずに正確に測ることが可能。しかし、KM から CD までは 1 座標あたり長くするべきと考えた。画角が大きい横方向では、誤差が大きく、IF の地点では 1 座標あたりの長さは小さく、FH の地点では、長くするべきである。

4 距離測定コード

4.1 目的

距離を測る画像内での距離を自動で測るコードの作成を行った。自作の重みを使って、距離を測定するコードはできなかったため、今回は、[8]torchhub のサンプルの重みを使ったコードにしてある。

4.2 作成コードの動作説明

1.物体検出後、その物体の[9]box の x 座標,y 座標の最大値、最小値ををを torchhub で求める。

2.1.で求めた座標から物体の中心座標を求め,a のリストに x 座標, b に y 座標を追加する。

3.2.リストの中から 2 つの物体の座標を取り出し、距離を測定する。これをすべての組み合わせになるように測定する。

4. 距離を表示する。

5. a,b のリストを空にする。空にすることで、次使う時に前のデータの影響なしで計測可能になる。

4.3 改善点

重みがサンプルデータのままなので、実験 I の重みを使えるようにする。どの物体との距離を測定しているかを出力していない。4.2 の 5.で前のデータの影響な

しだが、前のデータを保存することができない。そのため、前のデータをためることが可能なプログラムを作る必要がある。

6. 実験Ⅰ,Ⅱ,距離測定コードのまとめ,展望

実験Ⅰより、recall の値を上げるために 150 枚の画像学習では、最低でも枚数を今後増やす方針である。また。実験Ⅱで、正確に距離を求めるために、歪みが放物線に似ているため、積分で長さを求める可能性を考えている。測定コードは、画像内の距離測定の自動化を実現させた。その後、画像内に表示するシステムやある一定の距離以下になると警告するシステム製作する余地がある。そして、この 3 つを使い、リアルタイムの撮影するシステムを目標とする。

7 謝辞

本研究に関わったサイエンスアドバイザーの方,今回の研究に携わった橋本先生に感謝を申し上げる。

8 距離測定コード

```
import torch
model = torch.hub.load("ultralytics/yolov5", "yolov5s", pretraine
d=True)
print(model.names)
results = model("/content/yolov5/data/images/bus.jpg")
objects = results.pandas().xyxy[0]
a=[]
b=[]
for i in range(len(objects)):#len(object)は検出した物体の数
    name = objects.name[i]
    xmin = objects.xmin[i]
    ymin = objects.ymin[i]
    xmax = objects.xmax[i]
    ymax = objects.ymax[i]
    xcenter = (xmax+xmin)/2
    ycenter = (ymax+ymin)/2
    width = objects.xmax[i] - objects.xmin[i]
    height = objects.ymax[i] - objects.ymin[i]
    print(f"{i}, 種類:{name},中心:{xcenter,ycenter}")
    a.append(xcenter)
    b.append(ycenter)
A=0
B=0
n=len(objects)
while  B<n+1 and A<n :
  if B<n-1:
    B=B+1
    d=(((887/1000)*(a[A]-a[B]))**2+(((547/1000)*(b[A]-b[B]))**2))
**(1/2)
    print(d)
  else :
    A=A+1
    B=A
a=[]
b=[]
```

9.語句定義

[1].アルゴリズム

コンピューターが計算を行うときの計算、学習方法。

[2].yolov5

リアルタイムオブジェクト検出アルゴリズム。

[3].TP,FP,FN,recall,precision,AP,mAP,

TP

head と認識して、実際は head と認識した数。

FP

head と認識したが、head ではない数。

FN

head と認識しなかったが、実際 head である数。

precision(適合率)

予測がどれだけ正確化を表す値。

$$precision = \frac{TP}{TP+FP}$$

recall(再現率)

結果として出るもののうち、実際出てきたものの割合。

$$\text{recall} = \frac{TP}{TP+FN}$$

AP

p を precision の変数、r を recall の変数としたとき

$$AP = \int_0^1 p(r)dr$$

mAP

[4].labeling

物体や人を認識し、座標を出力する物体検出モデルに学習させるためのデータ作成を行うソフト。

各物体の AP の平均。

[5]googlecolab

Google のブラウザから python を使用できるサービ

[6]tensorboad

tensorboad は、機械学習の実験に必要な菓子か機能とツールを提供するキット。

[7]torchhub

学習済みのモデルを取り込むライブラリー

[8]box

物体検出した時に出力される囲い。

[9]リスト(list)

複数の値を管理するためのコレクション。

10.参考文献

[1]Qitta【物体検出】mAP (mean Average Precision) の算出方法
https://qiita.com/cv_carnavi/items/08e11426e2fac8433fed
[2]Qitta mAP(mean Average Precision)のまとめ
https://qiita.com/cv_carnavi/items/08e11426e2fac8433fed
[3]YOLO:Real-Time Object Detection-Joseph Redmon
https://pireddie.com/darknet/yolo/
[4]Qitta [YOLO V5]AI でじゃんけん検出
https://qitta.com/PoodleMaster/items/5f2cc3248c03821b8
[5][Python]画像から物体の種類,座標,幅,高さを求める https://kazuya-pg.com/object-detection-picture/
[6]スッキリわかる Python 入門
著:国本大吾/須藤秋良 監修:株式会社 フレアリンク