Vol.2018-AAC-7 No.9
2018/8/25

視覚障害者の屋外移動支援に向けた物体検出データセットの基礎検討とプロトタイピング

石曽根 奏子[1,a)]　馬場 哲晃[1,b)]　渡邉 英徳[2]　釜江 常好[3]

概要：本稿では視覚障害者の屋外移動支援に向けた物体検出データセット開発に関する基礎検討，及びそのプロトタイピングについて述べる．現在様々な深層学習用データセットが開発されている中，自動運転技術の発展により屋外移動時におけるデータセットも充実しつつある一方，視覚障害者が単独歩行する際に必要なデータセットに関する議論はあまりされていない．そこで既存データセットからの応用可能性及び，その場合の問題点を調査議論しつつ，必要なデータについては独自に開発することとした．本稿では特に初期評価プロトタイプとして，深層学習を利用することで歩行時の主要な情報をどの程度検出が可能なのかをまずはプロトタイプから明らかにすることで，データセットの基本的な設計を行った．

キーワード：視覚障害，支援技術，アクセシビリティ，深層学習，物体検出，プロトタイピング，データセット

Kanako Ishisone[1,a)]　Tetsuaki Baba[1,b)]　Hidenori Watanave[2]　Tsuneyoshi Kamae[3]

1. 背景

本研究で対象とする物体検出に限らず，公開されているデータセットには多くの種類がある．github 上でまとめられている Awesome Public Datasets[*1]では，医療や政府機関などの公開データセットを幅広くまとめている．物体検出において，大規模データセットを公開しているのは，現時点（平成 30 年 8 月 1 日）では Image Net[*2]や Google Open Images Dataset v.4 であり，15,440,132 個のバウンディングボックスが 600 カテゴリに対して，30,113,078 画像が 19,794 のカテゴリに対してアノテーションされている．自動運転における評価用データセットとして有名な KITTI データセット [1] では，自動運転に特化した様々なデータを公開している．この他 32x32 などの極めて小さな画像データセットを 8,000 万枚集めた Tiny Images Dataset[*3]や，それをサブセット化した CIFAR-10[*4]も評価データとしてはよく知られている．いずれも対象を絞ったデータセットにおいて，本研究を対象とする当事者の屋外活動支援のためには十分なデータセットが提供されておらず，少なくともプロジェクトチームにてまずはそのデータセットを開発する必要がある．

検出物体を絞った支援アプリケーションに「てんじぶろっく」がある．スマートフォンアプリも公開されている[*5]．ユーザはカメラをかざすことで点字ブロックの方向及び，ユーザに向かって横向きまたは縦向きかを音声ガイドによってフィードバックする．このように個別の物体検出に関してはすでにいくつかのアプリケーションが存在しているが，それ以外の単独歩行支援に必要な様々な物体検出を網羅している事例がない．本研究はそれを深層学習により実装するものであるが，議論を初期評価プロトタイプから行うデザインプロセスをとるため，まずは実働するシステムが必要となる．そこで，まずはラベルリストを検討し，そのリストに基づきデータセットを開発することと

1 首都大学東京
Tokyo Metropolitan University, Asahigaoka, Hino, Tokyo 191–0065, Japan
2 東京大学
The University of Tokyo
3 東京大学/スタンフォード大学
The University of Tokyo/Stanford University
a) kanako.ishisone@gmail.com
b) baba@tmu.ac.jp

*1 https://github.com/awesomedata/awesome-public-datasets
*2 http://imagenet.stanford.edu
*3 http://horatio.cs.nyu.edu/mit/tiny/data/index.html
*4 http://www.cs.toronto.edu/%7Ekriz/cifar.html
*5 https://itunes.apple.com/jp/app/てんじぶろっく/id1172646239

した．

2. ラベルリストの検討

データセットを開発するにあたり，どのようなラベルが最低限必要になるのかを検討した．評価用データセットとして頻繁に用いられる VOC[*6]及び COCO[2] データセットのラベルセットには，aeroplane, bicycle, cird, boat, bottle, bus, car, chair 等の一般的なラベルが用意されている一方で，点字ブロックや横断歩道，歩行者用ボタン等の視覚障害者支援物体検出はできない．参考までに COCO データセットで歩道を歩いている際の認識状況を図 1 に示す．人物及び車等を検知している様子がわかる．まずはこの画像に表示されている交差点や信号機（赤/青）等の他，点字ブロックやガードレール，横断歩道や歩行者用信号機を検出できることを初期評価プロトタイプで実現することとした．

図 1　汎用データセット COCO による認識結果サンプル（撮影は著者によるものであり，一部モザイク処理）．

表 1 に今回の初期評価プロトタイプにて登録したラベル及び登録バウンディングボックスの数を載せる．

3. アノテーション作業

データセット開発にあたり，アノテーション作業が必要となる．すでにアノテーションソフトには imgLab[*7]や BBox-LabelTool[*8]，VoTT[*9]等が存在しており，これらを利用することが一般的である．一方で，開発グループ内での高速プロトタイプを考慮した結果，ヒューマンエラーを減らすために，なるべく機能は削減したものが好ましいと判断し，アノテーションツールは自作した．自作したアノテーションツールは github 上で公開している[*10]．

3.1 映像撮影方法

アノテーションの対象となる画像を撮影するにあたり，

[*6] http://host.robots.ox.ac.uk/pascal/VOC/
[*7] https://github.com/davisking/dlib/tree/master/tools/imglab
[*8] https://github.com/puzzledqs/BBox-Label-Tool
[*9] https://github.com/Microsoft/VoTT
[*10] ofxYolov2: https://github.com/TetsuakiBaba/ofxYolov2

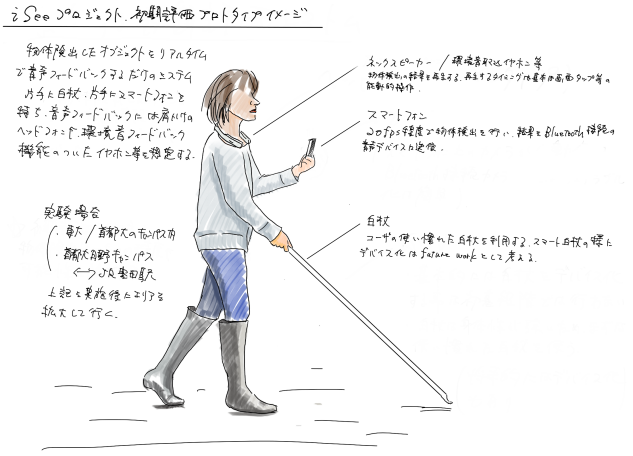

図 2　ユーザは片手にスマートフォンを持ち，カメラをかざして音声／触覚フィードバックを受ける

本研究ではすでにユーザの使用状況を明確にしている．図 2 となるべく同じ状況で取得した画像に対してアノテーションをすることが好ましいと考えた．まずは首都大学東京日野キャンパス及び JR 中央線豊田駅の往復をスマートフォンカメラにて動画撮影を行い，その後動画を 10 秒間隔で切り出し，アノテーション作業を行った．

スマートフォンを把持し，あるきながら撮影を行うとブレの多い映像となる．当事者が実際に使用する場合も同じ状況が考えられるが，利用シーンとしては，よく確認したい場合等は立ち止まったり，ゆっくりとカメラをむけることになる．そこで，学習用動画にはスタビライザを用いて撮影を行った．

撮影は平成 30 年 7 月 12 日，13 日にそれぞれ往路・復路を撮影した．JR 豊田駅から首都大学東京日野キャンパスまでの距離は約 1km 程度である．それぞれの動画から 10 秒おきに切り出した画像数は 940 枚，アノテーション数（バウンディングボックス数）は 4,417 個であった．

4. 学習

前節で用意したデータセットを元にネットワークの学習を行った．本研究では SSD[3] 及び YOLO[4] での実装を検討しているが，本稿では YOLO にて学習した結果を報告する．現在 YOLO は Version.3 であるが，今回利用したネットワークは Version.2 とした．ネットワークには yolov2-tiny 及び，yolov2 の 2 種類での学習を行った．詳細を表 2 に示す．いずれも 500,200 回のイテレーション及び，batch size, subdivision 数は初期設置のままとしている．トレーニング時のバッチサイズ及び，学習に使用した PC の主な仕様は ASUS All Series, Intel Xeon E5-1650 v4 4000 MHz (6 cores), GeForce GTX 1080 Ti x 4 である．

4.1 結果

学習したモデルを利用して，新たに録画した移動時の動

表 1　初期評価プロトタイプにて選定したラベル一覧．BBox 数は実際に登録作業をおこなった数．動画を撮影した後に，10 秒おきの画像に対してアノテーション作業をおこなったため，現時点ではバウンディングボックスの数に大きな偏りがある．

番号	クラス名	概要	BBox 数
0	person	人	589
1	bicycle	自転車	88
2	car	車	463
3	motorbike	オートバイ	21
4	bus	bus	25
5	train	train	1
6	truck	truck	70
7	boat	boat	0
8	traffic_light	traffic_light	89
9	bicycler	自転車に人が乗ってる（bicycler）	88
10	braille_block	点字ブロック (Braille block)	1001
11	guardrail	ガードレール (guardrail)	459
12	white_line	白線 (white line)	159
13	crosswalk	横断歩道 (crosswalk)	217
14	signal_button	歩行者ボタン	17
15	signal_red	歩行者信号機（赤）	43
16	signal_blue	歩行者信号機（青）	35
17	stairs	階段 (stairs)	16
18	handrail	手すり (handrail)	24
19	steps	段差 (steps)	40
20	faregates	改札機 (faregates)	9
21	train_ticket_machine	券売機	0
22	shrubs	植え込み (shrubs)	113
23	tree	街路樹 (tree)	153
24	vending_machine	自動販売機 (vending_machine)	16
25	bathroom	トイレマーク (bathroom)	1
26	door	ドア (door)	7
27	elevator	エレベータ (elevator)	1
28	escalator	エスカレータ (escalator)	0
29	bollard	車止め（bollar）	257
30	bus_stop_sign	バス停の看板	3
31	pole	電信柱	0

表 2　学習したモデルファイル．FPS はスマートフォン (iphone7) での参考値

Network	iteration[回]	model size[MB]	FPS
yolov2-tiny	500,200	44.4	15-20
yolov2	500,200	202.9	2-3

画に対して認識処理を行った．図 1 で示したフレームとほぼ同じ時間位置における，yolov2-tiny モデルの認識結果を図 3 に示す．

自転車搭乗者や横断歩道を二箇所検出できているものの，横断歩道の信号機，車用信号機は検出できていない．これは表 1 の BBox 数を参照すると，バウンディングボックスの絶対的な数量の少なさに起因していると考えられる．

図 4 では街路樹，ガードレール，点字ブロックを認識している．ガードレールや点字ブロック等は今回のようなバウンディングボックスではなく，Semantic Segmentation[5]

図 3　COCO のモデルと比較して，自転車搭乗者を bicycler，横断歩道箇所を Crossroad と正しく認識しているのがわかる．ただし，横断歩道の信号に関しては認識できていない．

によって提示されることが理想であるが，現時点では実時間実行速度の面で問題があり，将来的な議論としたい．図

5 では歩行者用信号機を青色として判別している様子がわかる．

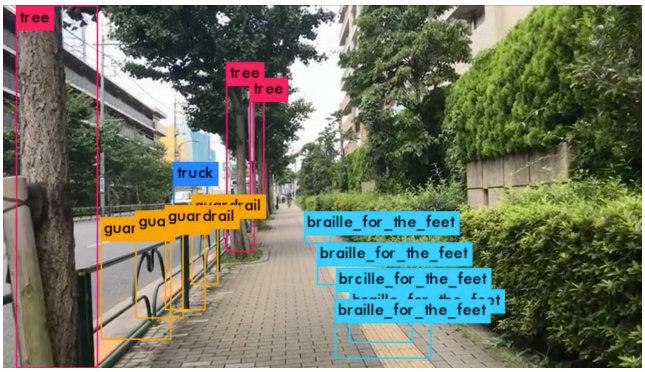

図 4　街路樹，ガードレール及び点字ブロック認識時の様子（yolov2-tiny モデル）

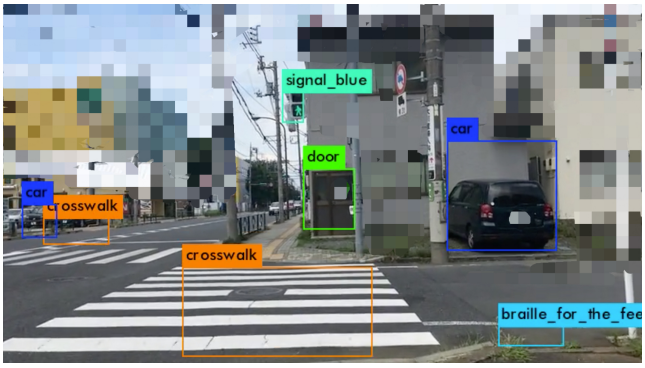

図 5　横断歩道，信号機，扉等の認識時の様子（yolov2-tiny モデル）

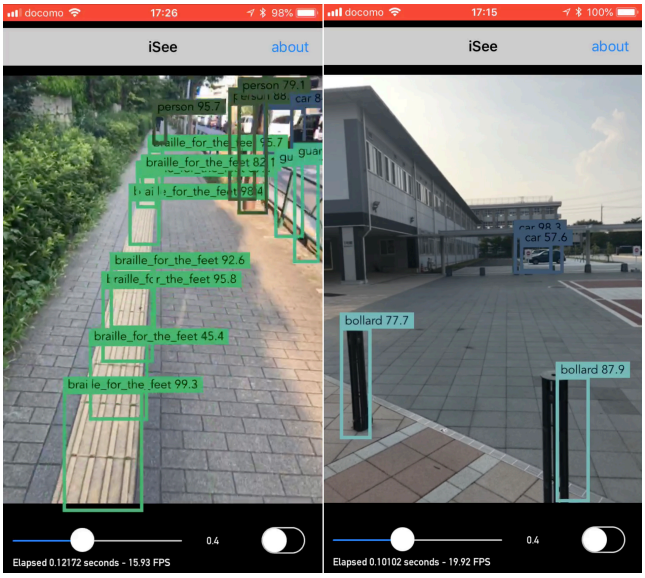

図 6　スマートフォン（iPhone7）上での動作の様子．画面下のスライダーは検出器の閾値設定であり，切り替えボタンは LED フラッシュ用

今回学習した学習済みネットワークを iOS 上に coreml ファイルとして移植することで，スマートフォン端末での動作も合わせて確認した．実行速度は表 2 を参照されたい．スマートフォン上で動作している様子を図 6 に示す．mAP の精確さと，実行速度のバランスに関して，現時点での両立は難しい．今回検証した yolov2 のネットワークモデルでは，yolov2 では 2-3fps 程度での実行速度であるため，12km/s の自転車走行に対して，3fps で処理した場合，1 フレーム検出時に自転車が 1.1[m] 進んでしまう．同様に歩行者に関しても，6[km/s] で歩く歩行者であれば，1 フレーム検出時に 0.65[m] 進んでしまう．これに対し，20fps で実行できている場合，自転車は 1 フレーム検出時に 0.055[m]，歩行者は 0.0235[m] となる．自動車のような移動速度程ではないが，即時性を考慮すると高い FPS を維持できることが好ましい．

5. まとめ

本稿では，視覚障害者の屋外移動のために必要な物体検出ラベル 31 種をまずは作成した後，実際に極めて小規模なデータセットにて学習させた結果を示した．物体検出に YOLO を利用し，その中でも実行速度が極めて早い yolov2-tiny ネットワークを用いて学習を行った．結果としてスマートフォン端末にて 20fps 程度の実行速度を得ることができ，歩行者用信号機，横断歩道，点字ブロック，ガードレール，車止め等の物体検出がリアルタイムに可能となった．しかしながらデータセットが小さいためまだその検出精度には問題があり，今後はデータセット拡充が必須である．また，音声フィードバックや触覚フィードバックに関しては今後の課題とし，まずは音声フィードバックを実装した後，ユーザフィードバックを集める．

謝辞　本研究は JSPS 科研費 JP18H03486 の助成を受けたものです。

参考文献

[1] Geiger, A., Lenz, P., Stiller, C. and Urtasun, R.: Vision meets Robotics: The KITTI Dataset, *International Journal of Robotics Research (IJRR)* (2013).
[2] Lin, T., Maire, M., Belongie, S. J., Bourdev, L. D., Girshick, R. B., Hays, J., Perona, P., Ramanan, D., Dollár, P. and Zitnick, C. L.: Microsoft COCO: Common Objects in Context, *CoRR*, Vol. abs/1405.0312 (online), available from ⟨http://arxiv.org/abs/1405.0312⟩ (2014).
[3] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y. and Berg, A. C.: SSD: Single Shot MultiBox Detector, *ArXiv e-prints* (2015).
[4] Redmon, J. and Farhadi, A.: YOLOv3: An Incremental Improvement, *arXiv* (2018).
[5] Long, J., Shelhamer, E. and Darrell, T.: Fully Convolutional Networks for Semantic Segmentation, *ArXiv e-prints* (2014).