CF-011

# ブロッコリー選別自動収穫機における深層学習を使った花蕾検出

# Flower Buds Detection Using Deep Learning in Broccoli Sorting Automatic Harvester

渡部 進一[†] 西田 宗司[‡] 中田 健信[‡] 吉田 信明[†] 横田 吏司[†] 畝村 暢一[§]
Shinichi Watanabe Takashi Nishida Takenobu Nakata Nobuaki Yoshida Satoshi Yokota Nobuichi Unemura

## 1. はじめに

農業人口減少に伴い，圃場の大区画化・大規模化・高効率化が急務である．現状のブロッコリー栽培の全労働時間の 70-80 % が収穫・調整作業であるが，選別・収穫を行う農業機械は市販されていない．そこで，マイコム株式会社では，一連の収穫作業を全て自動で行う「ブロッコリー選別自動収穫機」[1] 図 1 を開発している．収穫機の試作機では，画像処理手法により，人の手で設計された特徴量をもとに花蕾を検出している．撮影環境や識別対象の状態は一定ではなく，ブロッコリーの品種・生育状況・地面・撮影時の光量・花蕾とカメラとの距離といった様々な要素があるが，画像処理手法では，そのような様々な要素に対応する適切なアルゴリズム・パラメータの選定に多大な労力が必要となる．また，花蕾の一部が葉に隠れ，見えている花蕾部分のみを検出した場合，収穫適期判断や葉の除去に悪影響を与えるため，葉に隠れた部分を含めた花蕾全体を検出することが求められる．これらの課題を解決するため，深層学習による物体検出手法を適用し，ブロッコリー選別自動収穫機に組み込むシステム化を行った．

## 2. ブロッコリー収穫機

ブロッコリー収穫機の例は少なく，国内ではブロッコリー収穫機ヤンマー HB1250 がモニター販売を行っている．こちらはブロッコリー各株の生育状況に関わらず，一斉に収穫を行う．ブロッコリーは生育にバラツキがあるため，一斉収穫したブロッコリーのうち，一定以下の大きさのものは加工用として販売されるため，販売単価の低下が懸念される．ブロッコリーを選別して収穫するため，ブロッコリーの生育状況を画像処理によって判断する手法 [2] や，Mask R–CNN を使ったデータ拡張とネットワーク簡素化によってブロッコリーの花蕾検出に与える効果が研究されている [3]．

ブロッコリーの生育のバラツキ対応に加え，人手による調整作業を少なくするため，マイコム株式会社では，選択収穫が可能なブロッコリー選別自動収穫機を開発している．外葉の掻き分け，花蕾撮影，画像処理による収穫適期判断，外葉を根本まで切断，茎の切断，ベルトコンベアによる収穫物の回収まで，一連の収穫作業を自動で行う．連続収穫・ほ場からの搬出作業は含まない条件で，手収穫 4 個 / 分に対し，機械収穫 30 個 / 分を目標としている．人に代わり，画像から花蕾の大きさ・位置の検出，葉や周囲の状況判断，茎の位置を推定する必要があるため，画像から状況を判断する処理が極めて重要となる．


図 1: ブロッコリー選別自動収穫機

## 3. 物体検出

画像から特定の物体の位置と範囲を選定し，その物体が何かを分類するタスクを物体検出と呼ぶ．YOLO 以前の物体検出では，一度物体の候補となる領域を選定し，選定された領域をもとに SVM 等のアルゴリズムで分類を行っていた．Faster R–CNN [4] では，分類に加え，選定部分も深層学習で行う，言い換えると，End–to–End な深層学習による実装に初めて成功している．領域選定に深層学習のチューニングの少なさ，実行速度の向上，および高い検出性能が得られる一方，選定と分類で異なるネットワークを使う事による速度が十分に早くならない課題が残る．YOLO [5] では，選定と分類を一つのニューラルネットワークで実現し，高速な実行動作となった．直接位置予測，細かい特徴量，マルチスケールトレーニングを加えて，再現率と領域分離の精度を上げる事を目標としている．YOLOv2 [6] では，YOLO にバッチノーマライゼーション，高解像度の分類器，領域分類，YOLOv3 [7] では，YOLOv2 にクラス予測とスケール間予測に工夫を加えている．YOLOv4 [8] では，YOLOv3 からモデルアーキテクチャの変更，学習上の工夫，精度改善上の工夫を行っている．バージョンが上がるほど様々な工夫に加えてネットワークが大きくなるため，検出性能は向上する一方，実行速度は低下する傾向がある．本論文ではアルゴリズム選定時に開発されていた YOLO，YOLOv2，および YOLOv3 に対し，GPU 搭載のワークステーションを使って予備実験を行った．その結果，組み込み環境においてリアルタイム実行速度・実用的な検出性能が見込まれたため，YOLOv3 を採用した．

[†]公益財団法人京都高度技術研究所
[‡]マイコム株式会社
[§]株式会社 RIKU

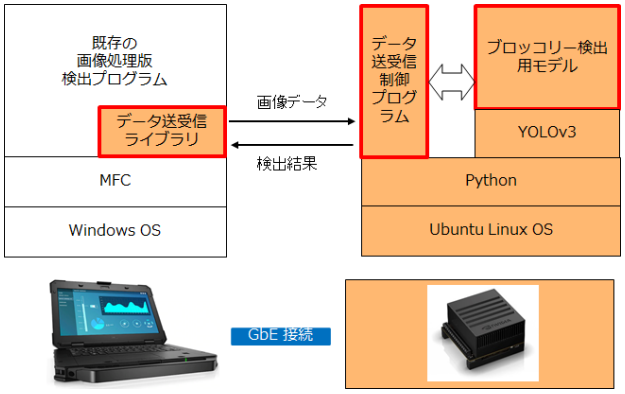


図 2: システム構成図

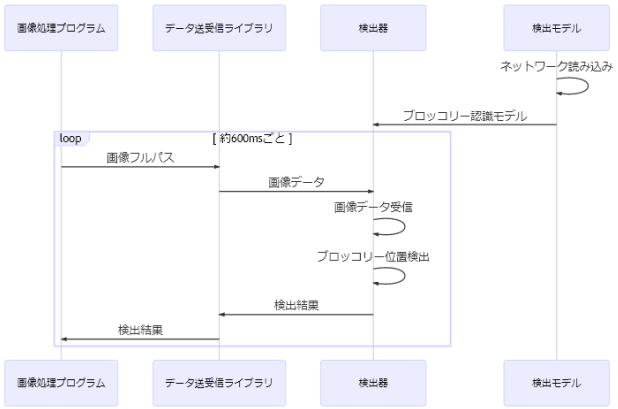


図 3: シーケンス図

## 4. 環境構築

実機に組み込んでリアルタイム処理を行うため，以下の動作環境構築と性能改善のためのモデル改良を行った．

1. シングルボードコンピュータ NVIDIA Jetson AGX Xavier [9]（以下，Jetson Xavier）と高速な物体検出を行うネットワーク YOLOv3 の動作環境構築
2. 実際に収穫機で撮影された画像に適切なアノテーションを付けた画像を追加学習したモデルを作成

### 4.1. 動作環境

花蕾の検出に画像処理を使ったものと，深層学習を使ったものを並行して動作できるよう，既存システムに外付け動作環境を構築した．価格・性能・大きさの観点から GPU を持つシングルボードコンピュータ Jetson Xavier を外付けの深層学習動作環境に選定した，図 2 にシステム構成図を示す．本論文で環境構築を行った箇所を色付き，本稿では特にブロッコリー検出用モデル，データ送受信制御プログラム，データ送受信ライブラリについて述べる（同図赤枠），収穫機のシステムは，防塵防滴保護機能 IP–53 認定の Dell Latitude 5420 Rugged（同図左下）上にソフトウェアで実装されている．Jetson Xavier とギガビットイーサネットで接続し，ブロッコリーの画像に対して検出結果を返すライブラリを開発した．YOLOv3 はいくつかのライブラリの異なる実装が公開されているが，Jetson Xavier では開発当時，Jetson 向け SDK である，NVIDIA JetPack SDK のバージョンに制約があり，使われるライブラリによって動かない実装がある．動作可能なもののうち，単精度浮動小数点指定による速度向上，学習の経過を図示する機能をもつ実装 [10] を利用している．図 3 にシーケンス図を示す．リアルタイム動作のため，処理に時間のかかるネットワークとブロッコリー認識モデル（ネットワークの重み）の読み込みはプログラム起動時に行う．収穫機システムからは一定間隔（外葉の掻き分け動作周期，ここでは約 600ms）ごとに撮影された画像データが送られ，検出器は検出結果をデータ送受信ライブラリに返す．

### 4.2. 検出モデルと追加学習

データセット COCO [11] により学習済みのネットワーク [12] を，ここでは一般物体認識モデルと呼ぶ．一般物体認識モデルはクラス数 80 の認識が可能であり，ブロッコリーのクラスは 2010 枚の画像で学習されている．

一般物体認識モデルには 2 つの問題がある．

1. ブロッコリーそのものの検出精度が得られない
2. ブロッコリーを検出できたとしても，収穫するための中心位置特定の精度が低い

1 つ目に，一般物体認識モデルのブロッコリー画像は，畑に生えるブロッコリー画像が無いわけではないが，そのほとんどが食材または料理のブロッコリーである [13] ため，畑に生えるブロッコリーとは色・形・撮影方向・周囲の環境が異なり検出精度が得られない．

2 つ目に，前述のように，畑のブロッコリーの花蕾に葉が被っている場合が多く，見えている花蕾部分のみを検出結果とすると，図 4 (a) の赤枠が検出領域となり，中心位置特定の精度が低い．葉に隠れた部分が検出領域とならず，花蕾の成長が不十分と判断されて本来収穫する所が見逃される．あるいは，収穫時期として適切と判断された場合も，葉によって隠れた花蕾が外葉カットの際に誤ってカットされる．また，図 4 (a) の赤枠の中心となる青十字部分を花蕾の中心，すなわち茎の位置と判断すると，茎のカット位置がずれるため，茎のカットで失敗する．

このように花蕾の大きさや範囲，中心位置を正しく認識できない場合，収穫判断の誤りや収穫作業によって花蕾を傷付ける結果となり得る．そこで，花蕾の正確な大きさや中心位置の取得のため，図 4 (b) のように隠れた部分も含めたアノテーションデータを作成し，学習に利用した．

追加学習には ImageNet [14] で学習済みの重みを使う．追加学習では学習されていない重みに対して 1 から学習する場合に比べ，少ない学習量で検出性能が高くなる事が知られている．

(a) 見えている部分の学習データ　(b) 見えていない部分を含めた学習データ

図 4: 画像

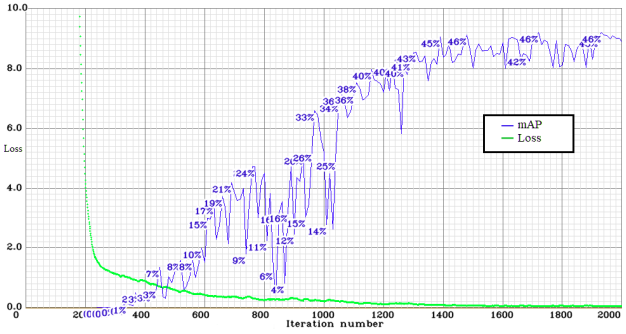


図 5: イテレーションに対する Loss，mAP

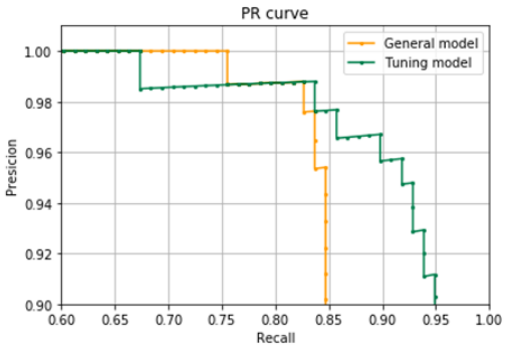


図 6: PR 曲線

## 5. 評価

本節では，一般モデルの課題が追加学習で解決できたかどうか，定量的評価と定性的評価により確認する．

### 5.1. 評価条件

ここでは，追加学習による課題解決を評価するため，Jetson Xavier の実機環境は使用せず，ブロッコリー選別自動収穫機の試作機により予め撮影された画像 684 枚に対し，ワークステーションで性能を評価している．葉によって花蕾が隠れていた場合，図 4 (b) 同様に隠れた部分を含めて花蕾全体を正解とするデータを作成した．撮影画像は 2048 × 1088 の画素であるが，YOLOv3 のニューラルネットワーク入力時に 416x416 にリサイズされる．データセットの数を表 1 に示す．表 1 の

表 1: データセットの数

| データ種類 | 花蕾完全 | 花蕾不完全 | 花蕾なし |
|---|---|---|---|
| 学習データ | 125 枚 | 169 枚 | 0 枚 |
| 検証データ | 42 枚 | 58 枚 | 0 枚 |
| 評価データ | 42 枚 | 56 枚 | 194 枚 |

「花蕾完全」は花蕾すべてが画像に映る画像，「花蕾不完全」は花蕾に一部葉が被る画像を意味する．学習データ：検証データ：評価データ = 3 : 1 : 1 として，評価データにのみ花蕾が映っていない（花蕾なし）画像を加えて評価している．YOLOv3 では，学習データに花蕾なし画像を加えなくても，花蕾以外の環境も学習しているため，花蕾なし画像は使わなかった．また，実際の収穫機では花蕾の大きさから収穫時期の判断をしたり，形がいびつな場合に花蕾でないと判断する処理が入るため，適合性の性能が高くなるが，ここではそのような処理を行う前の純粋な物体検出性能のみを評価する．

### 5.2. 性能評価

#### 5.2.1. 定量評価

イテレーションに対する Loss[¶]，mAP[‖]を図 5 に示す．イテレーション 1400 を超えると Loss は 0 近くに収束し，mAP は 46% 近辺の値を取る．クラスは「ブロッコリー」と「その他」の 2 種類存在するため，ブロッコリー単体では mAP は約 92% の値となる．以上から，検証データで十分に学習が進み，高い検出精度となる事が確認できる．IoU[**]（Intersection over Union）=0.5 以上を正解として評価した Precision[††]（適合率）に対する Recall[‡‡]（再現率）の曲線 PR 曲線を図 6 に示す．Precision=0.90 において，一般的な物体検出モデル Recall=0.85 に対し，追加学習モデルでは Recall=0.95 に改善した．これは検出した花蕾 10 個中 9 個が花蕾，1 個が花蕾でない閾値に設定した時，追加学習モデルに変更することで，見逃す花蕾が 20 個中 3 個から 1 個に減ることを意味する．以上のように，追加学習により，見逃しが少ない結果が得られた．

#### 5.2.2. 定性評価

追加学習により，花蕾が葉に隠れた場合でも花蕾全体を検出するかどうか評価する．一般物体認識モデル，および追加学習モデルによる花蕾検出領域を図 7 に示す．画像の上の矩形の枠内を花蕾として認識している．一般物体認識モデルでは花蕾が葉に隠れた場合や，葉によって分断された場合は見えている花蕾やその一部のみを花蕾として検出している．追加学習モデルでは見えている花蕾部分だけでなく，葉に隠れた部分も含めて花蕾全体を検出する結果が得られた，

¶正解とどれくらい離れているかを表す値
‖平均適合率 AP (Average Precision) の全クラス平均

**正解領域と予測領域の重なり度合い
††検出した花蕾総数のうち，正しい花蕾の割合
‡‡実際に存在する花蕾総数のうち，正しく検出した花蕾の割合

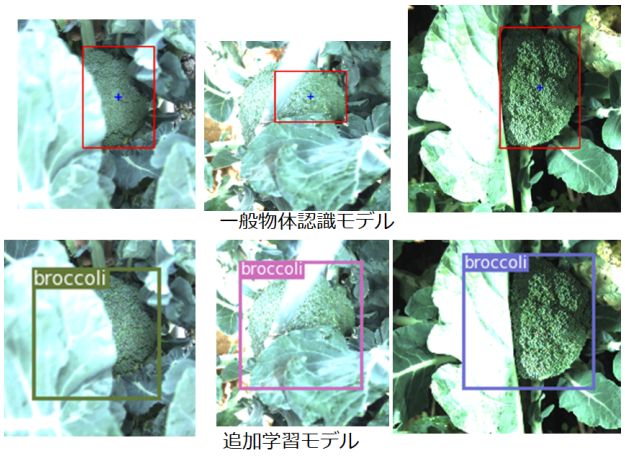


図 7: 花蕾検出領域の改善例

## 6. まとめと今後の課題

ブロッコリー選別自動収穫機において，深層学習の評価に加え，装置に組み込んでシステム化を行った．アノテーションデータに改良を加えた追加学習により，花蕾が隠れた場合でも隠れた部分を含む花蕾領域を検出する結果が得られた．今回，数百枚のデータの学習であるため，データ拡張により性能向上が見込まれる．

画像の左右反転，上下反転，および検出対象物の一部をマスクして学習することで，対象物の一部が隠れている場合の検出性能が向上する RandomErasing [15] といった手法を組み合わせ，データを拡張を行いたい．

また，実機動作に対応可能な性能であるが，実環境での動作は十分に確認できていない．今後は収穫機と接続して運用した際の問題の洗い出し，性能改善を進める必要がある．

## 謝辞

本研究は，農林水産省が主導する「福島イノベーション・コースト構想に基づく先端農林業ロボット研究開発事業」の一環として実施したものである．

## 参考文献

[1] “ブロッコリー選別自動収穫機”. http://www.mycom-japan.co.jp/n_topics/bah2-1800.html. (Accessed on 18/06/2021).

[2] Pieter M Blok, Ruud Barth, and Wim Van Den Berg. “Machine Vision for a Selective Broccoli Harvesting Robot”. *IFAC-PapersOnLine*, Vol. 49, No. 16, pp. 66–71, 2016.

[3] Pieter M Blok, Frits K van Evert, Antonius PM Tielen, Eldert J van Henten, and Gert Kootstra. “The effect of data augmentation and network simplification on the image-based detection of broccoli heads with Mask R-CNN”. *Journal of Field Robotics*, Vol. 38, No. 1, pp. 85–104, 2021.

[4] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. “Faster R–CNN: Towards Real-Time Object Detection with Region Proposal Networks”. *arXiv preprint arXiv:1506.01497*, 2015.

[5] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. “You Only Look Once: Unified, Real–Time Object Detection”. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 779–788, 2016.

[6] Joseph Redmon and Ali Farhadi. “YOLO9000: Better, Faster, Stronger”. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7263–7271, 2017.

[7] Joseph Redmon and Ali Farhadi. “YOLOv3: An Incremental Improvement”. *arXiv preprint arXiv:1804.02767*, 2018.

[8] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. “YOLOv4: Optimal Speed and Accuracy of Object Detection”. *arXiv preprint arXiv:2004.10934*, 2020.

[9] “AI Platform for Autonomous Machines — NVIDIA Jetson AGX Xavier”. https://www.nvidia.com/en-us/autonomous-machines/jetson-agx-xavier/. (Accessed on 18/06/2021).

[10] “AlexeyAB/darknet: YOLOv4/ Scaled–YOLOv4/ YOLO–Neural Networks for Object Detection (Windows and Linux version of Darknet )”. https://github.com/AlexeyAB/darknet. (Accessed on 18/06/2021).

[11] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. “Microsoft COCO: Common Objects in Context”. *CoRR*, Vol. abs/1405.0312, , 2014.

[12] “YOLO: Real–Time Object Detection”. https://pjreddie.com/darknet/yolo/. (Accessed on 18/06/2021).

[13] “COCO–Common Objects in Context. https://cocodataset.org/. (Accessed on 18/06/2021).

[14] “ImageNet”. https://www.image-net.org/. (Accessed on 18/06/2021).

[15] Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. “Random Erasing Data Augmentation”. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34, pp. 13001–13008, 2020.