ユーザーズマニュアル

物体認識システム

発行日 2021 年 3 月 31 日
公立大学法人会津大学
株式会社東日本計算センター

目次

1. はじめに 1
1.1. 物体認識システムとは 1
1.2. 本書の記載範囲 1
1.3. 動作環境 2
1.4. 使用機器 2
1.5. 関連資料 2
2. 本システムでできること 3
2.1. 動画、画像から物体の認識 3
2.2. 認識対象の３次元座標の算出 3
3. 本システムのアプリケーションについて 3
3.1. アプリケーションの起動 3
3.2. 画面の構成 4
3.3. 入力データ 5
3.4. 出力データ 9
3.5. 実行結果メッセージ 10
4. 物体認識システムの設定変更について 11
4.1. object.json ファイルについて 11
4.2. config.ini ファイルについて 12
4.3. 学習済みモデルの変更 13

1. はじめに

1.1. 物体認識システムとは

ロボットナビゲーションにおいて、事前に経路を計画しロボットを自動走行させせますが、走行経路における想定外の障害がある場合に経路を再計画し走行する必要があります。その経路を計画する際にはロボットの位置情報や障害物の位置情報も必要になるため、その情報を外部環境のカメラから計測しロボットへ提供させる必要があります。

今年度は LICTiA 1F に設置されたマルチ監視カメラシステムで撮影した動画を基に物体認識を行い、特定の基準から認識対象(ロボットや人、机、椅子)の位置を計測します。また、認識対象の位置計測結果とタイムコードを紐づけしたCSVファイルを出力します。

1.2. 本書の記載範囲

本書の記載範囲は、物体認識システムを使用するユーザー向けの操作マニュアルになります。

本システムの環境構築に関しては「インストールマニュアル_物体認識システム」を参照してください。

1.3. 動作環境

本システムの動作環境を記載します。
依存ライブラリバージョンは本システム検証時点となります。

表 1-1 使用機器一覧

環境		バージョン	補足
OS	Ubuntu	18.04 LTS	-
CPU	Intel Core i7	-	4 コア以上推奨
メモリ	16GB 以上推奨	-	-
ストレージ(SSD)	512GB 以上	-	-
依存ライブラリ	TensorFlow	1.15.0	-
	Keras	2.3.1	-
	YOLO	v3	-
	OpenCV	3.4 以上	-
	FFmpeg	3.4.8 以上	-
	PyQt	5.15 以上	-

1.4. 使用機器

本システムで使用する機材を次の表に記載します。

表 1-2 使用機器一覧

No.	使用機器	個数	補足
1	パーソナルコンピューター	1	-

1.5. 関連資料

本システムと関連する資料を次の表に記載します。

表 1-3 関連資料一覧

No.	資料名
1	インストールマニュアル_物体認識システム.docx

2. 本システムでできること

2.1. 動画、画像から物体の認識

マルチビューシステムで撮影した動画(mp4 形式)またはスクリーンショット(png、jpg 形式)とカメラキャリブレーションファイルを使用し、物体の認識をおこないます。

2.2. 認識対象の 3 次元座標の算出

物体認識した対象の位置から 3 次元座標の算出をおこないます。

3. 本システムのアプリケーションについて

本システムは GUI アプリケーションとなります。

物体認識を実施する映像または画像とカメラキャリブレーションファイル、実行結果出力先を選択することで物体認識を行います。物体認識結果として認識した物体名、物体の原点からの 3 次元座標などをＣＳＶファイル形式で出力します。

3.1. アプリケーションの起動

ターミナルを起動し、以下のコマンドを実行することでアプリケーションを起動します。

```
$ python3 object_recognition.py
```

3.2. 画面の構成

アプリケーション画面の各項目についての説明を記載します。

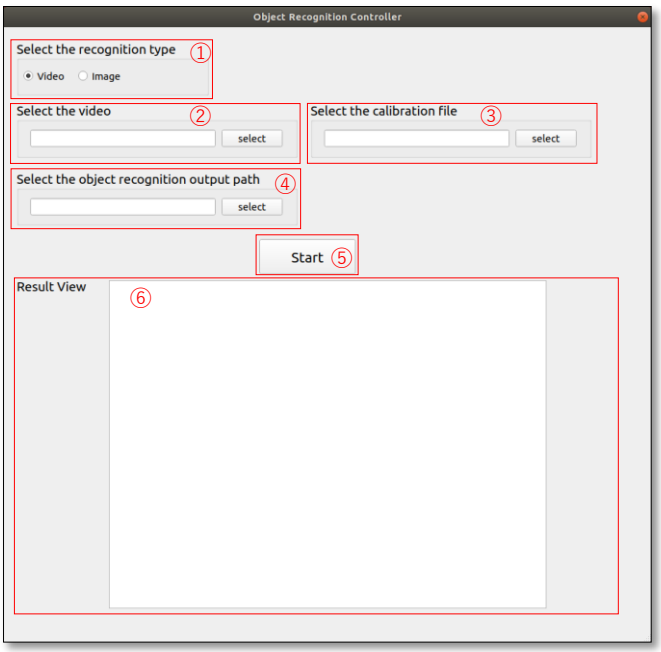


図 3-1. アプリケーション画面

表 3-1 画面項目一覧

No.	ラベル		種別	説明
①	Select the recognition type	Video	ラジオボタン	物体認識する入力ファイルを動画または画像をラジオボタンで選択します
		Image	ラジオボタン	
②	Select the video/image		入力	物体認識を行う動画/画像ファイルを選択します
③	Select the calibration file		入力	撮影したカメラのキャリブレーションファイルを選択します
④	Select the object recognition output path		入力	物体認識の結果を出力するフォルダを選択します
⑤	Start		キーボタン	物体認識を実行します
⑥	Result View		表示	物体認識した結果を表示します

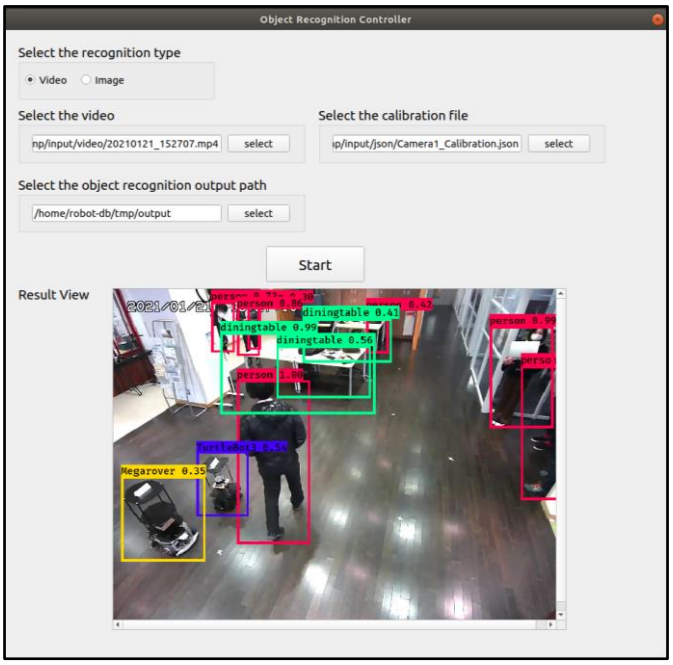


図 3-2. アプリケーション実行結果画面

3.3. 入力データ

(1) 動画ファイル形式とフォーマット

本システムに動画を入力する際は、mp4 ファイル形式とします。

以下に入力動画ファイル形式のフォーマットを記載します。

表 3-2 入力動画ファイルのフォーマット

スペック	フォーマット
ファイル形式	mp4
解像度	1280×960
フレームレート	24fps もしくは 30fps
映像コーデック	H.264
ビットレート	2Mbps 以上

(2) 画像ファイル形式とフォーマット

本システムに画像を入力する際は、JPEG または PNG ファイル形式とします。

以下に入力画像ファイル形式のフォーマットを記載します。

表 3-3 入力画像ファイルのフォーマット

スペック	フォーマット
ファイル形式	jpg、png
解像度	1280×960

(3) カメラキャリブレーションファイルの形式とフォーマット

本システムで入力する動画/画像を撮影したカメラのキャリブレーションファイルは Json ファイル形式で以下の構成とします。

以下に入力画像ファイル形式のフォーマットを記載します。

表 3-4 入力キャリブレーションファイルのフォーマット

スペック		フォーマット
ファイル形式		JSON
項目		説明
reprojection_error_range		カメラのキャリブレーションの実行結果で 再投影誤差の値が設定されます
internal_params	fx	内部パラメータ：カメラ行列が設定されます
	fy	
	cx	
	cy	
	k1	内部パラメータ：カメラの歪み係数が設定されます
	k2	
	p1	
	p2	
	k3	
external_params	rx	外部パラメータ：回転行列が設定されます
	ry	
	rz	
	tx	外部パラメータ：並進ベクトルが設定されます
	ty	
	tz	
camera_id		カメラ ID：撮影されたカメラの識別 ID です

```
{
    "reprojection_error_range":0,
    "internal_params":{
        "fx":679.21442,
        "fy":709.53296,
        "cx":617.23828,
        "cy":336.51652,
        "k1":-0.027378,
        "k2":0.020522,
        "p1":-0.039052,
        "p2":-0.0028827,
        "k3":-0.039194
        },
    "external_params":{
        "rx":2.04395,
        "ry":0.38909,
        "rz":-0.28667,
        "tx":-0.3109,
        "ty":0.92856,
        "tz":3.31736
        },
    "camera_id":2
}
```


図 3-3. カメラキャリブレーションファイルの記載内容

3.4. 出力データ

(1) 画像ファイル形式とフォーマット

物体認識結果の画像は png ファイル形式で出力されます。

出力する画像ファイル形式のフォーマットは以下の通りです。

表 3-5 出力画像ファイルのフォーマット

スペック	フォーマット
ファイル形式	png
解像度	640×460

(2) 物体認識結果 CSV ファイルの項目内容

各認識した物体名とその座標をタイムスタンプで紐づけた結果を CSV ファイル形式で出力します。出力するデータは以下の各項目となります。

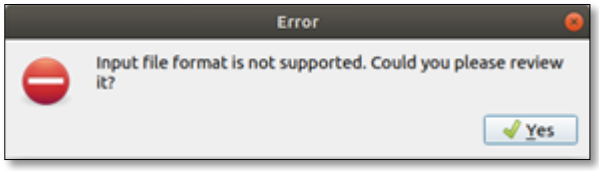
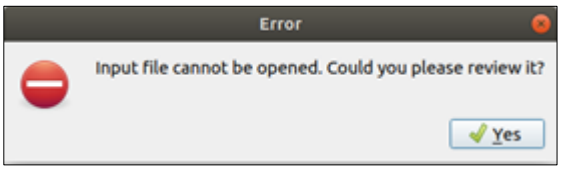
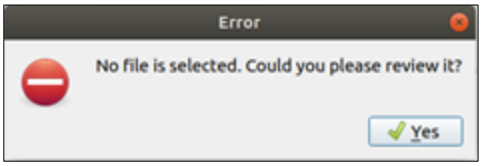
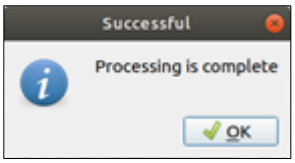
表 3-6 CSV 出力項目一覧

項目	説明
robot_id	ロボット ID、本システムでは”obj_recognition_01”固定です
video_id	動画 ID
video_datetime	動画の撮影日時
video_name	動画ファイル名
image_name	画像ファイル名
image_path	画像ファイルパス
image_date	画像撮影日時
image_x	ピクセル X 座標
image_y	ピクセル Y 座標
object_name	認識物体名
detection_rate	物体認識率
object_x	３次元 X 座標
object_y	３次元 Y 座標
object_z	３次元 Z 座標
camera_id	撮影カメラ ID
calibration_path	キャリブレーションファイルパス

3.5. 実行結果メッセージ

物体認識システムを実行終了時に以下のメッセージを出力します。

表 3-7 実行結果メッセージ一覧

No.	項目	説明
1		未対応の動画・画像ファイルが入力された場合に表示するエラー通知です 「Input file format is not supported. Could you please review it?」
2		動画・画像ファイルがオープンできない場合に表示するエラー通知です 「Input file cannot be opened. Could you please review it?」
3		物体認識を行う動画・画像ファイルが選択されていない場合に表示するエラー通知です 「No file is selected. Could you please review it?」
4		物体認識が正常に成功した場合に表示する通知です 「Processing is complete」

4. 物体認識システムの設定変更について

物体認識システムでは設定ファイルの値を変更することで物体の認識や座標の計算値を変更することができます。

また、学習モデルを更新させることで認識率などの向上させることができます。

表 4-1 設定ファイル

項目名	説明
object.json	認識する物体名の個別 Z 座標、認識先のモデルを設定します
config.ini	全体の Z 座標の計算値を設定します

4.1. object.json ファイルについて

object.json ファイルでは認識させる物体と検出させるモデルの切り替えや検証用 Z 座標を設定することができます。

下記の object.json ファイルの内容では 7 つの物体を認識させるように設定されています。

認識させる対象の各項目を追加削除することで認識対象を変更することができます。

```
1  [
2      {"class": "Megarover",   "z-coordinate": 0.35, "def_detect": false},
3      {"class": "TurtleBot3",  "z-coordinate": 0.25, "def_detect": false},
4      {"class": "diningtable", "z-coordinate": 0.36, "def_detect": false},
5      {"class": "chair",       "z-coordinate": 0.40, "def_detect": false},
6      {"class": "Pepper",      "z-coordinate": 0.60, "def_detect": false},
7      {"class": "UGV-Y",       "z-coordinate": 0.25, "def_detect": false},
8      {"class": "person",      "z-coordinate": 0.85, "def_detect": true}
9  ]
10
```

図 4-1. object.json の記載内容

表 4-2 実行結果メッセージ一覧

項目名	説明
class	認識する物体名が記載されます
z-coordinate	検証用 Z 座標設定値(m)を設定します
def-detect	物体認識を行うモデルを選択します(true:汎用モデル、false:会津大学学習モデル)

4.2. config.ini ファイルについて

config.ini ファイルでは認識した物体全体の検証用 Z 座標を設定することができます。

```
1 [coordinate]
2 Z = 0
```

図 4-2. config.ini の記載内容

表 4-3 実行結果メッセージ一覧

項目名	設定値	説明
Z	0	Z 座標を 0 に設定
	1	Z 座標を 0.325 に設定
	other	Z 座標が z-coordinate の値に設定

4.3. 学習済みモデルの変更

物体認識で使用する学習済みモデルを使用する際は” object_recognizer.py”の以下の項目を変更することでモデルを変更することができます。

表 4-4 実行結果メッセージ一覧

変数名	説明
uoa_model_path	学習済みモデルファイルパスを設定します
uoa_class_path	学習済みモデルのクラスファイルパスを設定します
uoa_learning_image_size	学習モデルで学習した画像サイズを設定します

```
3  # -*- Python -*-
4  import sys
5  import numpy as np
6  import configparser
7  import os
8  import json
9  from PIL import Image, ImageFont, ImageDraw
10 from keras import backend as K
11 from object_info import PerceptibleObject
12 sys.path.append(os.environ['HOME'] + "/src/keras-yolo3")
13 from yolo import YOLO
14 from yolo3.utils import letterbox_image
15
16
17 uoa_model_path = 'model_data/ep474-loss27.411-val_loss23.896.h5'
18 uoa_class_path = 'model_data/uoa_classes.txt'
19 uoa_learning_image_size = (640, 640)
```

図 4-3. “object_recognizer.py”のコード

著作権

本文書の著作権は公立大学法人 会津大学に帰属します。
この文書のライセンスは以下のとおりです。
クリエイティブ・コモンズ 表示 2．1日本
http://creativecommons.org/licenses/by/2.1/jp/