J. Fac. Edu. Saga Univ.
Vol. 1, No. 2 (2017) 23～30

生物育成に応用可能な画像処理プログラムの作成

小野　文慈，嬉　　正勝

Production of Image Processing Program Applicable to Biological Development

Bunji ONO, Masakatsu URESHI

要　　旨

近年，パターン認証技術や画像処理技術の発展により産業界では自動化や無人化が進んでいる。個人認証・顔認証技術は社会の安全性や利便性向上に貢献している。特に自動車において衝突回避をする自動ブレーキ，人間に代わり認知，判断，操作を行う人口知能（AI），白線に沿って自動操舵する車線維持制御技術や車間距離を一定に保つ車間距離制御技術が開発されており，自動運転が実現するのも間近であると感じられる。この技術の中でも画像処理技術は広範囲な比重を占め，制御するプログラムのアルゴリズムは非常に複雑であり，専門家以外はほとんど理解不可能であると思われる。

一方，教育現場において，上記のようなブラックボックス的なところはさておき，簡単な画像処理であれば利用できる機会はあると思われ，その一例として画像処理技術を用いたプログラムの作成を試み，理科教材への応用可能性を検証した。プログラム作成は，他のプログラム言語に比べて画像処理関数が用意され，かつ一般に無料開放されておりプログラムの開発が有利であることからＣ言語を使用している。作成したプログラムは昆虫の生育・成長過程の観察に役立てることを想定してコオロギの個体数のカウント，体格および体長についての測定を実施した。

１．はじめに

画像認識技術は身近なところでは OCR（optical character reader），防犯セキュリティ技術として顔認証，虹彩（アイリス）認証などがあり，工業分野[1)]，医療分野[2)]，食品業界[3)]など多岐の分野で利用されている。たとえば工業分野においては印刷面の文字認識検査，機械部品のパーツ検査，密封容器のシール検査というように画像処理検査機として使用されている。画像認識では，画像データから対象物となる輪郭を抽出し，背景から分離した上で，その対象物が何であるかを分析するのが一般的である。本研究では対象物を認証するのが目的ではなく，生育途中の画像データから対象物（コオロギ）を正確にカウントすること，その後の輪郭処理から輪郭内面積や軸長を計算するプログラムを作成し，対象物の齢数でグループ化することで理科教材等の活用法として有効であるかを検証した。プログラムの開発環境はＣ言語（C ++，C#）[4)5)]を用い，さまざまな画像

佐賀大学　教育学部　学校教育講座

処理関数を使用するためにOpenCV（Open Computer Vision Library）を組み込んだ。OpenCVはインテルが開発・公開したオープンソースでインターネット環境があれば誰でも無料で簡単に使用できる。本論文では読者のプログラムの作成をアシストできるように使用した関数名を記述するようにする。

2．プログラム処理

今回のプログラムの画像解析手順のフローチャートを図1に示す。図中には示していないが画像を読み込む前に前処理を行っている。前処理をした後で二値化，反転，島化，個体測定，カットオフ処理，マーキング・ナンバリング，テキストデータの書き出しという手順で処理を行っている。

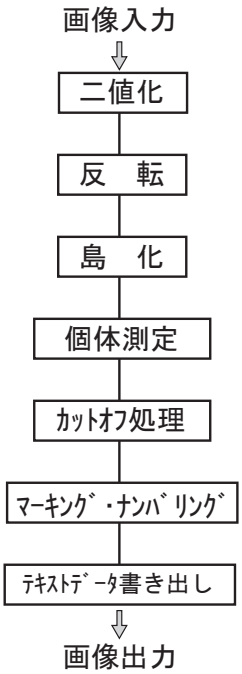
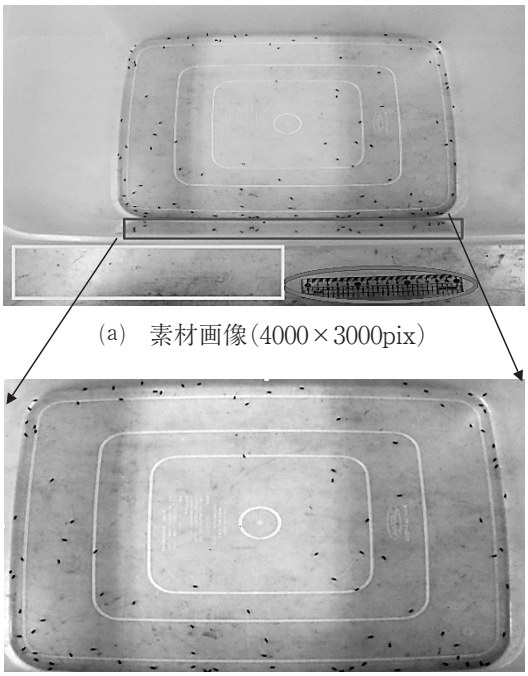


図1　フローチャート

2．1 前処理

図2に素材画像の前処理の1例を示している。素材となる画像は画素数4000×3000pixであるが，倍率が一定ではないために画像中にスケールをいっしょに撮影している。スケールの目盛と実寸法よりコオロギの体長の換算比率を算出できる。しかし解析のときは外乱（ゴミと呼ぶ）となりうる。また，画像により明るさが異なったり，対象物（コオロギ）以外の物体が映り込んでいるのであらかじめ手動で消去しておかなければならない。この例の場合はスケールを取り除き，コオロギが容器の壁面に反射しているのでトリミングを行うために画像を切り出した（スケールの除去は反転処理の後に行う場合もある）。前処理に使用したソフトはwindows標準ソフトpaintで可能である。



(a)　素材画像(4000×3000pix)

(b)　前処理後の画像

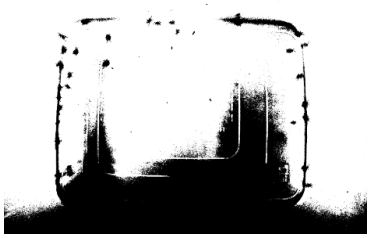
図2　素材画像の前処理

2．2 二値化および反転処理

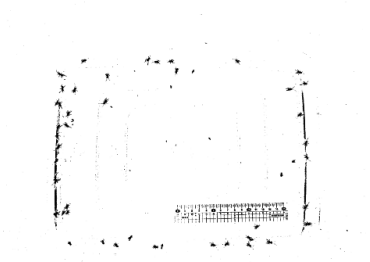
カラー画像は各ピクセルにR，B，G成分を内包しており，0～255までの256階調（8bit）で表現されている。二値化処理とはこの256階調に閾値を設定し，0（黒）か1（白）に設定しなおす処理である。図3(a)のようにOpenCV提供の単純二値化関数Threshold関数を使うと容器のふちも黒く塗られてしまい，部分的にコオロギを

しっかり判別することができなくなってしまう。単純二値化は RGB 成分を算術平均し，その値が 0 ～127は 0 に，128～255は 1 に置き換わり，すなわち黒 = 0，白 = 1 となっている。OpenCV ではさらに上位の適応二値化関数が用意されており，ピクセルの近傍のピクセルの平均値を元に閾値を設定する二値化法（適応的閾値処理）である adaptiveThreshould 関数（ここでは適応二値化と呼ぶ）を採用すると，図 3(b)のようになる。補足すると適応二値化は OpenCV だけではなく画像加工ソフト（例えば Illustrator）を利用してもコオロギをじょうずに認識できる。

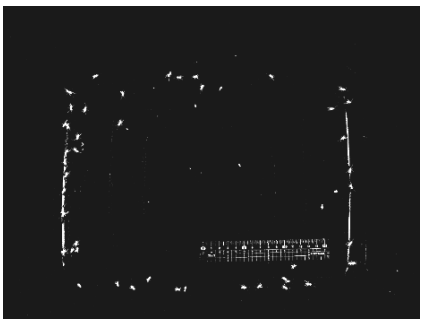
図 3(c)の反転処理は前述した図 3(b)の白と黒を入れ替えたものである。この処理を施す意味は大半の画像処理のライブラリは二値化画像を処理する際に白の場所を「モノ」として認識し，黒の場所を「穴」として認識する特性がある。そのためコオロギを対象物とするために「モノ」（白化）にする必要があり白と黒を入れ替える反転処理を行っている。



(a)　単純二値化 Threshold 関数処理画像



(b)　適応二値化 adaptiveThreshould 関数処理画像



(c)　反転処理

図 3　二値化処理および反転処理

	0	1	2	3	4	5	6	7	8	9
0										
1										
2			□					□		
3					□	□		□		
4								□		
5										
6						□	□			
7			□	□						
8										

(a)　個体画像

	0	1	2	3	4	5	6	7	8	9
0	0	0	0	0	0	0	0	0	0	0
1	0	−1	−1	−1	0	0	−1	−1	−1	0
2	0	−1	1	−1	−1	−1	−1	1	−1	0
3	0	−1	−1	−1	1	1	−1	1	−1	0
4	0	0	0	−1	−1	−1	−1	1	−1	0
5	0	0	0	0	−1	−1	−1	−1	−1	0
6	0	−1	−1	−1	−1	1	1	−1	0	0
7	0	−1	1	1	−1	−1	−1	−1	0	0
8	0	−1	−1	−1	−1	0	0	0	0	0

(b)　島化処理

図 4　島化の様子

2．3 島化（ラベリング）

島化とは白と黒の境界のピクセルに-1という数値を内包させ，白(1)を囲い，輪郭を生み出す作業のことである。単純化のために９行10列の画像を図４(a)に示してあり，これを島化したものが図４(b)となり浮き出た島がコオロギ個体に相当する。これをカウントすることで対象の個体数を測定することができる。輪郭処理とも言う。

輪郭処理関数は findContours（ bin_img, contours,CV_RETR_EXTERNAL，CV_CHAIN_APPROX_NONE）で表され，第１引数は画像名，第２引数はクラスを参照にする変数であり，contours を宣言することで輪郭を構成する画素数，座標や色の成分などが参照にできる。第３，４引数は予約語である。

2．4 個体測定（重心，面積，慣性モーメント）

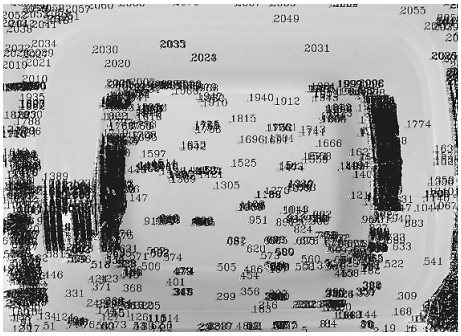
この処理が固有のプログラム部分で３章　個体測定アルゴリズムにおいて解説する。

2．5 カットオフ処理

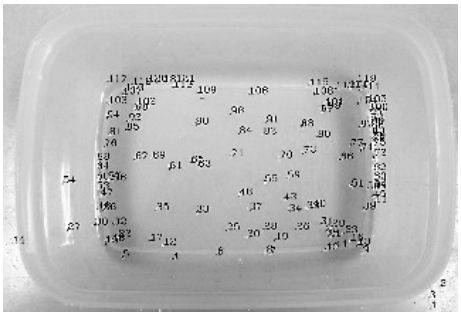
カウントする際，画像中の小さな島はゴミになる場合がある。そこで測定数の誤りを減少させるため，ある一定の画素領域はノイズとして削除する。Cut_off 値80を設定した場合，80pix 以下のものは個体としてカウントしないという意味である（図５）。図中数字はカウント数を示している。Cut_off 値の適正値は倍率やそれぞれの撮影条件により異なるので１枚ごとに適当に設定しなければならない。

2．6 マーキング・ナンバリング

カウントした個体の重心に○印をつけ右肩に番号を表示する（図５）。このカウントした画像は二値化画像にはなっていないが島化をする前にプログラム内では二値化処理を行っており，マーキングおよびナンバリングは元画像に施して出力画像としているので注意が必要である。



(a)　Cut_off なし（実匹数2075匹）



(b)　Cut_off 値80（実匹数121匹）

図５　カットオフ処理

2．7 テキストデータ書き出し

図６のようにナンバリングの番号に対応した個体の軸長（体長）と面積（体格）をエクセルで統計処理ができるようにテキストデータとして出力する（ただし図６の体長，体格はスケール比率を掛けていない）。コオロギは不完全変態昆虫であり成長が段階的である。今回用いたフタホシコオロギは孵化後（１齢）脱皮を８回行い，成虫となる。生育環境が一定であれば体長から齢を求めることが可能なため，齢毎にクラス分けするねらいがある。

A	B	C
コオロギ No	体長［mm］	体格［mm2］
1	55.59	874
2	53.89	859.5
3	59.2	897
4	35.73	541.5
5	69.04	907.5
6	50.21	1098.5
7	49.67	739.5
8	57.11	1068
9	83.46	1628.5
10	28.78	404.5

図６　個体データの書き出し結果

3．個体測定アルゴリズム

この章では対象物（コオロギ）の寸法測定に関するアルゴリズムを述べる。画像からコオロギの体長，体格を得るための必要な処理方法を説明する。

3．1 体長の測定

図６に示すように，コオロギの長軸（体長）を測定するのは個体の慣性モーメントが最小になる軸長を求めることに相当する。まず個体ごとの重心を求め，それを原点として回転させ，各画素に対する慣性モーメントの総和を計算する。その合計が最小になる傾斜角度を求めた後，傾斜角度分だけ回転させそのときの軸長を求める。

3．1．1 重心の求め方

図７の１個体の重心を算出するには，白塗り要素全部の x 座標の平均と，要素全部の y 座標の平均を求めればよい。それが重心（x_o，y_o）となる（図中の黒丸）。初期の画素の座標は整数型であるが，重心の座標は浮動小数点型でもよい。図中の重心は（4.4，4.8）になる。

3．1．2 座標変換

画素を回転させると原点からの距離 r は変わらないが直交座標系では新しい座標となる。１つの画素の初期座標を（x，y）とすると θ だけ回転移動した場合，新座標（x'，y'）は次のようになる。

$$x' = x\cos\theta - y\sin\theta$$
$$y' = y\sin\theta + x\cos\theta$$

3．1．3 慣性モーメント

慣性モーメントの定義は質量×距離である。厳密にはコオロギの微小画素面積×距離となるので面積（１次）モーメントと呼ばれるが便宜上，慣性モーメントと呼ぶことにする。慣性モーメントは任意の軸を基準に取ることができるが今回はＹ軸に関する慣性モーメント I_Y を求めている。

3．1．4 長軸（＝体長）の測定

島化した個体画素を１°おきに180°まで回転させ，慣性モーメント I_Y の総和を算出する。この総和が最小となる角度だけ回転させたとき，コオロギが直立または倒立する（図８）。このときＹ軸に沿う画素の原点から最も遠い上部の画素の

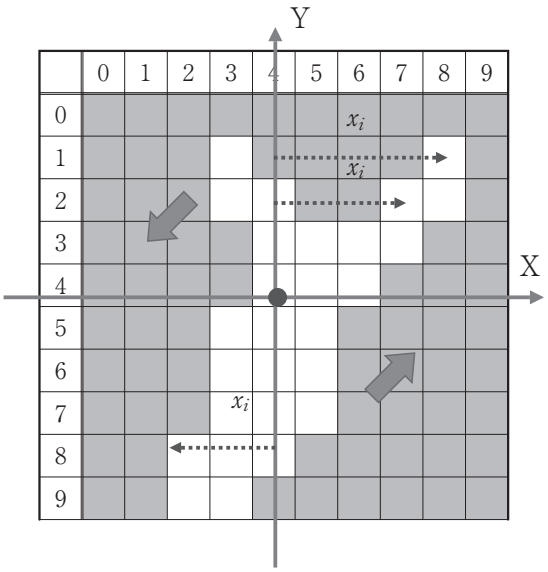


図７　重心および慣性モーメント

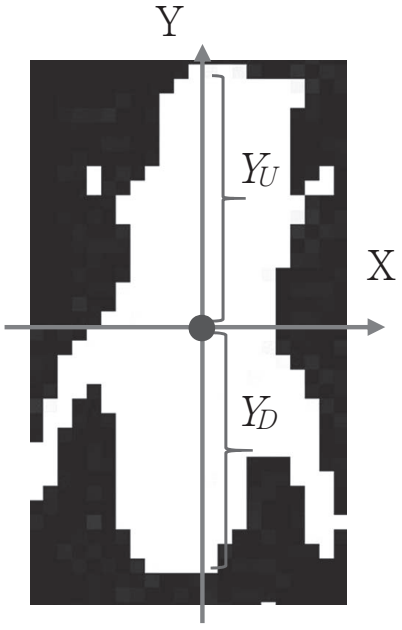


図８　直立した実際の画像

Y_U，下部の画素 Y_Dの合計が長軸の長さとなる。

3．2 体格（＝面積）の測定

島化された島の画素数を単純に足せばコオロギの面積を求めることができるが，島化関数で輪郭を配列として記憶しているのでcontourArea（引数）関数にその輪郭のポインタを引数として渡せば内部の面積を一気に求めることができる。contourArea関数はグリーンの定理を用いて閉曲線に囲まれた内部の面積を求めている。

4．画像処理結果および考察

本研究で作成したプログラムを用い，実際のコオロギ観察画像で（公称で100，800，1500匹）解析を行った。今回用いた画像はすべて1齢幼虫のみが写っているものである。

4．1 カウント

(a) **実匹数が100匹の場合**

実匹数が100匹の画像を3回測定した結果を表1に示す。1枚目のカウント写真を図9に示す。それぞれの画像は同じ個体ではなく独立したものである。実匹数が100匹に対し，誤差は約1割程度であるが3回ともプラスになっていることからゴミを拾っている可能性がある。2枚目のカットオフ値が30の時，誤差は大きくなっているのでカットオフ値を適当に選択すれば誤差数が少なくなると思われる。

(b) **実匹数が800匹の場合**

実匹数が800匹の画像を3回測定した結果を表2に示す。1枚目のカウント写真を図10に示す。100匹のときと比較して3回とも誤差がマイナスの値となった。これは個体と個体が重なっており，複数を1体としてカウントしていると考えられる。

(c) **実匹数が1500匹の場合**

実匹数が1500匹の画像を3回測定した結果を表3に示す。1枚目のカウント写真を図11に示す。800匹の画像よりもマイナスの誤差が大きいことから，重なっている個体が多いだけではなく，個

表1　実匹数100匹の画像の測定数

	Cut_off 値	測定数	誤差
1枚	40	102	+2
2枚	30	116	+16
3枚	40	105	+5

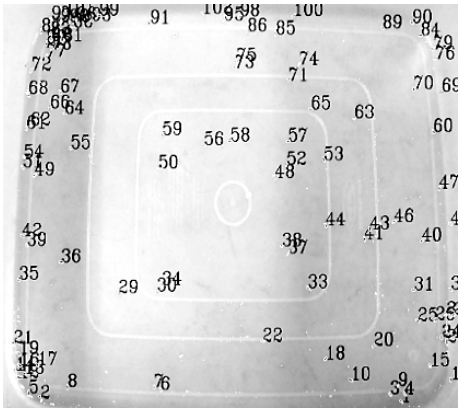


図9　実匹数100匹の画像の測定例

表2　実匹数800匹の画像の測定数

	Cut_off 値	測定数	誤差
1枚	20	700	−100
2枚	30	737	−63
3枚	20	714	−86

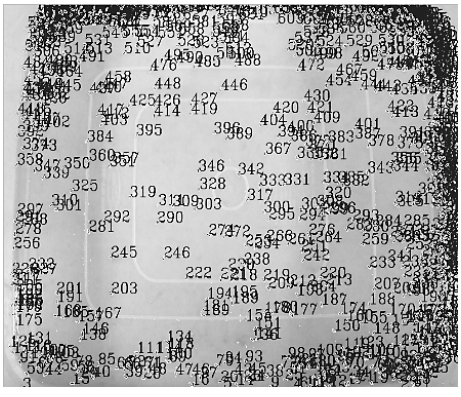


図10　実匹数800匹の画像の測定例

表３　実匹数1500匹の画像の測定数

	Cut_off 値	測定数	誤差
１枚	20	1247	－253
２枚	20	1239	－261
３枚	20	1237	－263

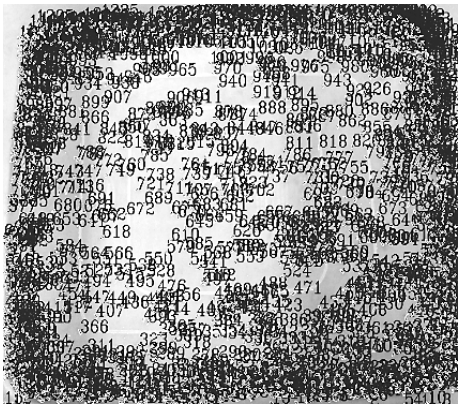


図11　実匹数1500匹の画像の測定例

体と個体の足や触覚が触れているのも大きな１匹としてカウントしていると考えられる。

以上の結果から100匹の画像のように重なりが少なく散らばっているような場合にはカットオフ値を適当に選ぶことによりかなりの精度でカウントできる。しかし，個体数が多く，重なってしまうとカウント数が減ってしまうため，素材画像となる対象物の面積密度を小さくする必要がある。今回の場合だと，複数回に分けて撮影することが考えられる。

さらに技術的にカウント精度を上げる方法として縮小処理（erode 関数）と言うものがある。個体が重なった画像に縮小処理を繰り返していくことにより個体が複数個に分離されるのでカウント数の補正ができるが，これも目視で確認しながらでなくてはならず，自動というわけにはいかない。

４．２　体長・体格について

図６に算出した体長・体格測定の結果とその解析画像の対応はほぼ正確にとれていた。しかし，個々に検査してみると手足が見えないものや，重なった場合（９番，18番）などが見受けられた（図12）。個体の測定精度としてはまだ不完全で個体１体が200pix～400pix で構成されているため体格の有意差（齢数：フタホシコオロギの場合は９段階）は識別できないと思われる。有意差を出すためには１体当たりの画素数を増やす，すなわち倍率をあげることが精度の向上となるであろう。したがって今回の倍率では目安程度である。

５．おわりに

OpenCV のライブラリ関数を使用して自動カウントプログラムを作成し，昆虫の生育・成長過程の観察に役立てることを想定してコオロギの個体数，体格および体長の測定を行った。その結果，素材画像の撮り方と分解能に相当する倍率を適当に選べば解析結果の精度は向上するという結果を得た。プログラムの内容としてはカウント機能，面積，軸長を求めるものであったが，理科教材への応用として植物育成の成長過程の観察，月の満ち欠け率の観察，鉱物の含有成分の面積割合など応用が十分できる場面があると考えられる。

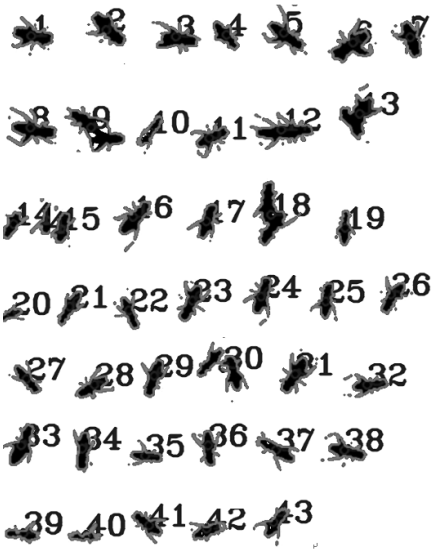


図12　実際のカウント画像例

参考文献

1）江口正夫：画像輝度ヒストグラムに基づく統計的真実接触面積測定（第1報），トライボロジスト，第57巻，第5号，pp.345-352(2012)
2）鳥居大哉・柴田義孝：医療画像処理を用いた診断システムの研究，情報処理学会第53回全国大会，Vol 4，pp.335-336(1996)
3）黒沢正明・中西祥八郎：画像処理による農畜産物の判別，日本ファジィ学会誌，Vol 6，No 1，pp.42-48(1994)
4）林晴比古：明快入門 Visual C ++，SB クリエイティブ，(2011)
5）皆本晃弥：やさしく学べるC言語入門，サイエンス社，(2007)