**カイワレダイコンの育成に必要な情報得る画像処理**

本作「シミュレーションソフト」を説明する前に、本作に必要な植物ライブラリの製作過程を説明します。

大浦丈晃

**資料３**

**カイワレダイコンの育成に**

**必要な情報を得る画像処理**

※カイワレダイコン(以下「植物」という。)

**＜目的＞**

コンピュータによって植物の育成に必要な情報を随時、

得られるプログラムを作成する。以下３つを必要とする。

・植物の伸長(小数点第一位まで)

・植物の成長過程での時期(発芽期、成長期、収穫期、過剰成長期、測定不可)

・植物育成LEDライトの入り・切り(成長期・収穫期：ON)

以上の項目以外にも、 シミュレーションソフトのために必要なプログラムも作成する。

・元の画像を1/4リサイズし、罫線のある画像を生成

**＜実験方法＞**

植物の情報を得るには伸長が重要となる。

植物の伸長を得るにはいくつかの方法がある。赤外線センサーなどの電子機器を利用して伸長を得る方法・測量技術を用いて精密な伸長を得る方法などがあるが費用や技術が限られているためできない。費用や技術力を考慮して以下の実験方法とする。

植物を対象に横から随時、撮影する。画像の位置が変わらないよう、植物の育成する位置とカメラの位置を固定し撮影。その画像から伸長を得る。実験１、植物を測る上で２㎝、３ｃｍ…１３㎝の境界線(高さ:y軸)の値(単位:pixel)が画像のどの位置なのかを求めるプログラムを作成し、求めたデータを実験２で活用する。実験２、随時、カイワレダイコンの個々の成長にばらつきがあるためその中で一番長く成長した点（以下「最長点」という。）の値(単位:pixel)を求めるプログラムを作成する。そして、最長点の値を元に、上記「目的４つ」のプログラムを作成する。



**最長点**

**実験時に**

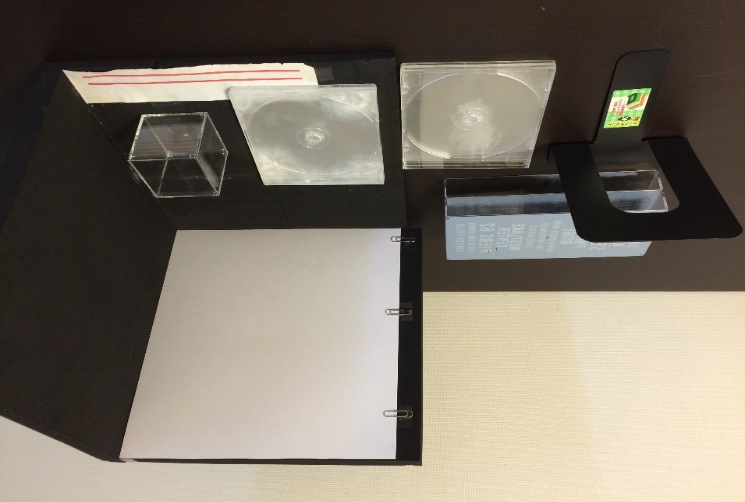
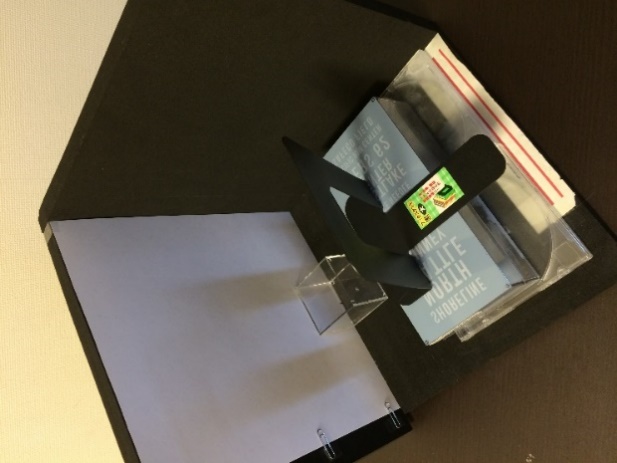
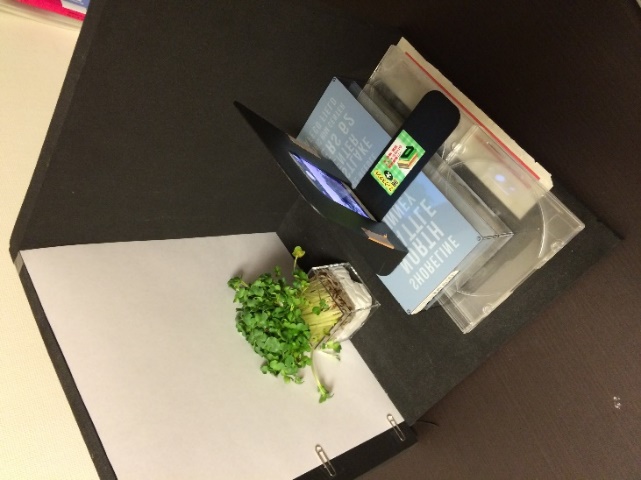
**撮影した画像**

高さ：何pixel？

**＜実験環境の構築**＞

コーナンや100円ショップなどから購入し５０００円以内で用具を購入

家にあるものも活用する。



**＜実験１＞**

植物を測る上で２㎝、３ｃｍ…１３㎝の境界線(高さ:y軸)の値(単位:pixel)が

画像のどの位置なのかを求めるプログラムを作成し、求めたデータを実験２で活用

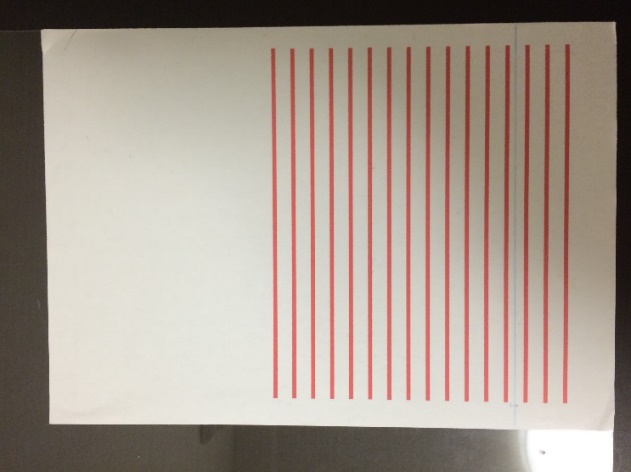
設備環境が保証されないため実験結果データが正しく出るか不安定ではあるのでカメラ等をしっかり固定し、植物とカメラの距離に気を付けながら撮影する(画像１)。

A4の紙に１㎝の間隔で赤色の線１６本を作成する(画像２)。その用紙を育成する位置に置き、画像内に１２本の赤色の線が映るよう撮影する(画像３)。

また、下から見て、最初の赤色の線を２ｃｍの境界線とし最後の線を１３ｃｍとする。

**画像２**

**画像１**



画像の幅：中央・高さ：0～3263(pixel)から各境界線の幅をピクセル単位で求める

画像をプログラムで扱いやすくするために「(1)元の画像.JPG=>(2)二値化画像-モノクロ.bmp(修正あり)=>(3)二値化画像-24ビット.bmp(この画像を使用) 」に変換。

**１３ｃｍ境界線**

**２ｃｍ境界線**

**画像３**

**実験１で作成するプログラムの情報**

（C言語）

コンパイラ：

Borland C++ 5.5.1 for Win32 Copyright (c) 1993, 2000 Borland

実行ファイル：

・「**./提出/bin/実験１/実験１.exe**」実行ファイル

・(3)二値化画像-24ビット.bmp

・幅：中央・高さ：0～3263(pixel)のRGB色数値.txt　(色数値：「0 0 0」か「ff ff ff」)

※プログラムで扱いやすくするためにモノクロビットにしてから２４ビットマップに変換

１～３の画像を参照する場合「./提出/data/実験１で使用した画像/」のディレクトリにある

**※コードレビュについて：**

「./提出/src/実験１/」フォルダ内に「実験１.c」があります。

　また、詳細なコメントはしていません。

**実行方法**

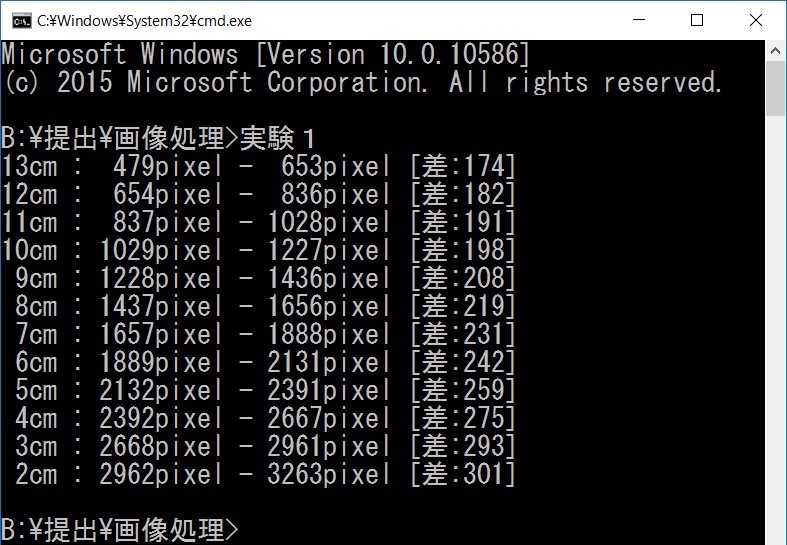
１．コマンドプロントを起動

２．カレントディレクトリを「./提出/bin/実験１/」に設定

３．コマンド入力「＞実験１」

４．最初の行「13cm : 479pixel - 653pixel [差:174]」

「479」は１３ｃｍの境界線の値・「653」は１２ｃｍの境界線を超過する境界線の値



撮影した画像(プログラムで扱いやすいよう２値化した画像)と

プログラムの出力結果、その他関連することをまとめた図

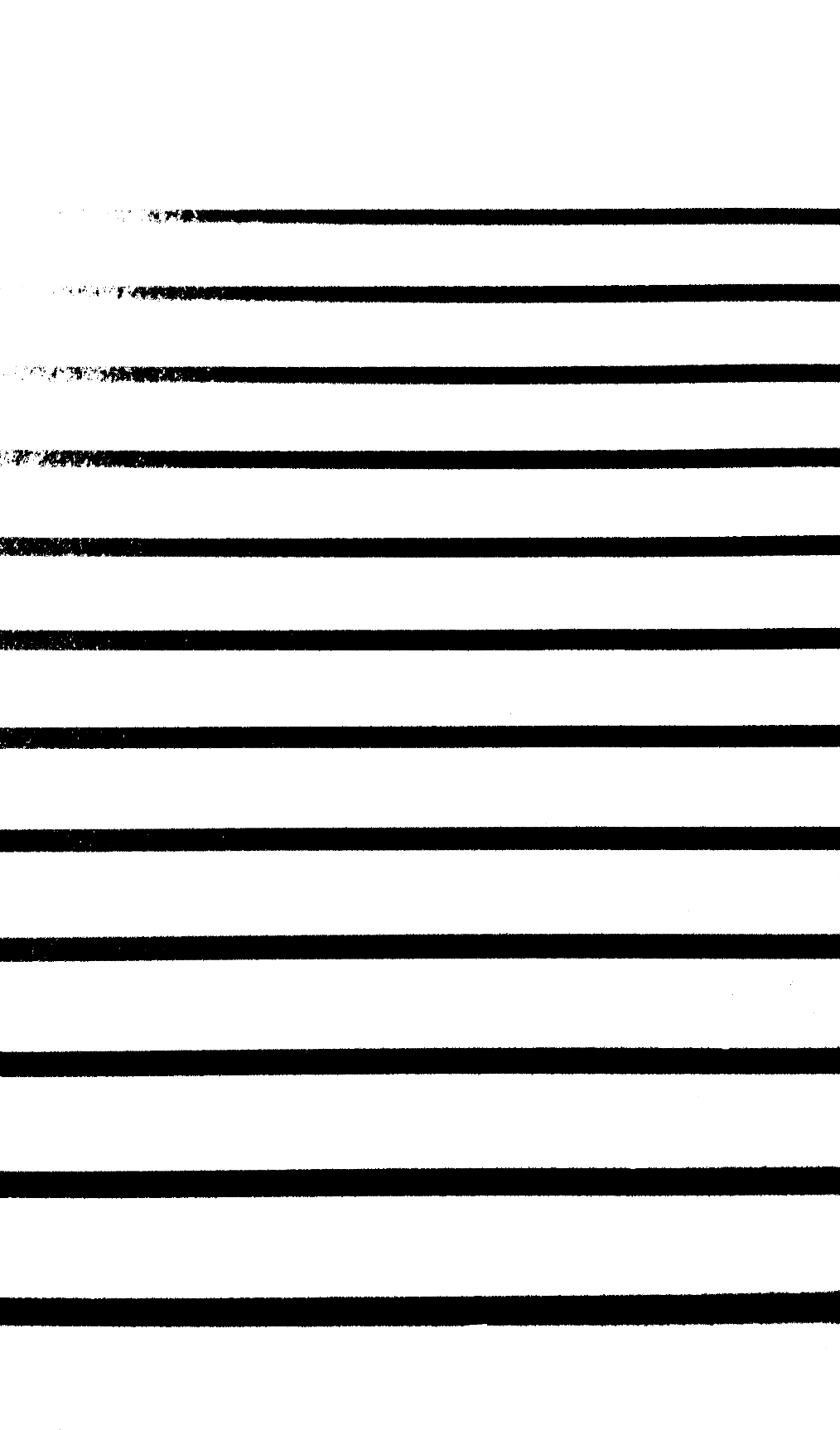
2392～2667

1889～2131

1657～1888

1437～1656

837～1028



479～653

654～836

1029～1227

1228～1436

2962～3263

2132～2391

2668～2961

0～478

単位：pixel

幅：中央・高さ：y

**発芽期**

**収穫期**

**成長期**

**過剰成長期**

**測定不可**

**13cm**

**9cm**

**10cm**

**11cm**

**12cm**

**6cm**

**5cm**

**7cm**

**8cm**

**3cm**

**4cm**

＊画像の位置関係により

0～1cmは求めない

カメラレンズ位置

**2cm**

遠近法によりレンズから離れていくにつれて１ｃｍの間隔が画像では間隔が短くなっている。

**OFF**

**ON**

**OFF**

ライトON/OFF

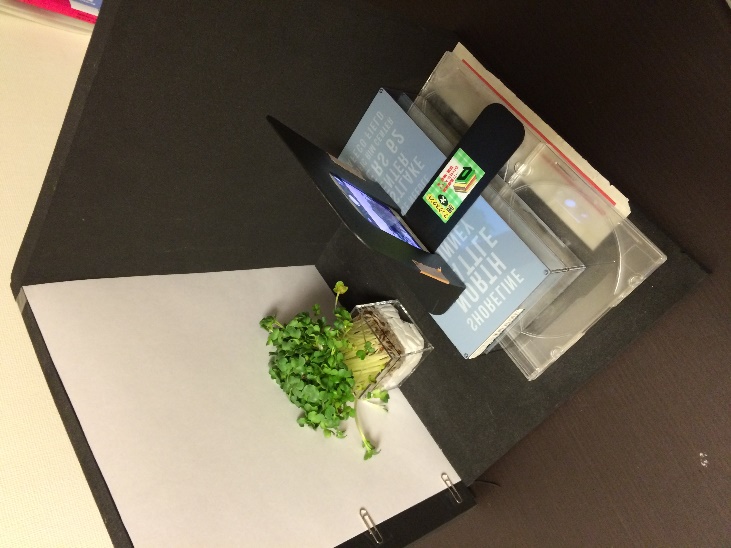
植物期間

**＜実験２＞**

植物を対象に撮影した画像から最長点の値(単位:pixel)を

求められるプログラムを作成

背景画像と植物育成時の画像を差分する方法と、植物育成時の画像をエッジ検出する方法がある。「差分する方法」では、２枚の画像が必要になり設備環境が整っていない中で行うには不向きと考え、**「エッジ検出する方法」**を模索する。



**最長点**

高さ：何pixel？

**今後、この画像を使用する**

青色の線は「種を置く位置」である。

青色から赤色の線は１ｃｍ分の長さである。この部分は、画像に映さない。

**撮影する部分**

**１．植物画像からエッジ検出をし、最長点を求める**

※カラー画像をエッジ検出する前に画像をグレースケールにする必要がある。



**１．元の画像をグレースケールに変換**

**３．最長点を求める**

２値画像なので、（白：２５５）の位置を探索し、画像の左上の端からピクセルごとに右に移動し最長点を求める

for (y ・・・) for (x ・・・)

２枚の画像は同じ大きさである。

幅（ｘ軸）：２４４８pixel

高さ（ｙ軸）：３２６４pixel

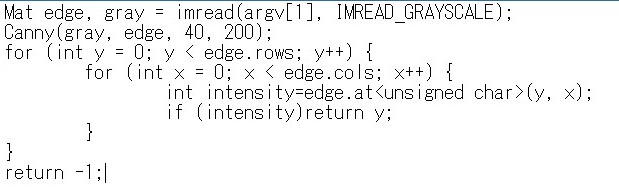
ｙの値で伸長を求める

**探索開始位置(0,0)**

（白：２５５）を探索

**２．エッジ検出**

**コード：**



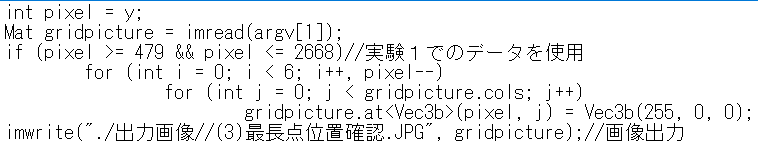
出力結果：

(最長点：ｙ軸)１４２６ (pixel)

この出力結果が正しく最長点を求められているか確認する。

以下、確認するため最長点の位置を青（255,0,0）の線を元の画像に印字して画像生成する

コード：



出力結果：(画像)



以上の結果から最長点の位置が正確であると言える。

よって、「植物画像からエッジ検出をし、最長点を求める」プログラムを今後、使用する。

最長点の値を元に…

２．リサイズし罫線ある画像を生成

３．伸長を求める

４．カイワレダイコンの各期間

５．成長促進ライトの入り切り

の各プログラムを作成

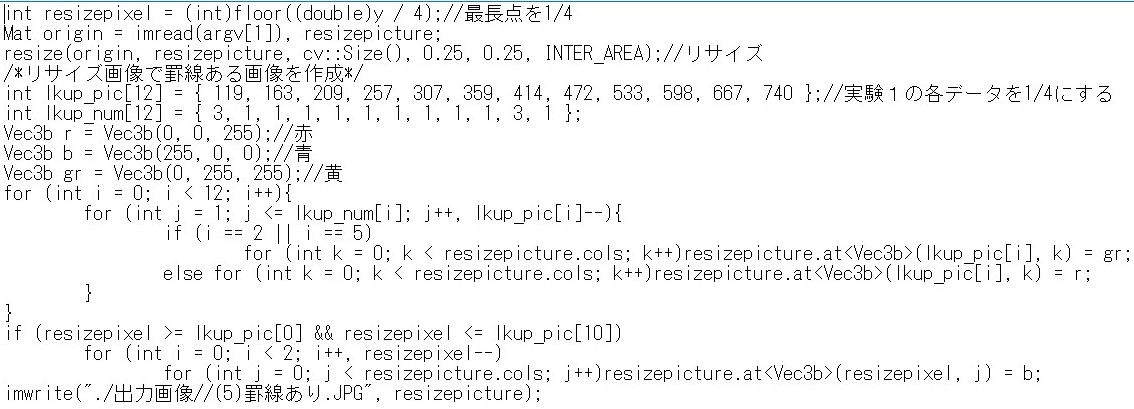
**まとめ**

**２．元の画像を1/4リサイズし、罫線のある画像を生成**

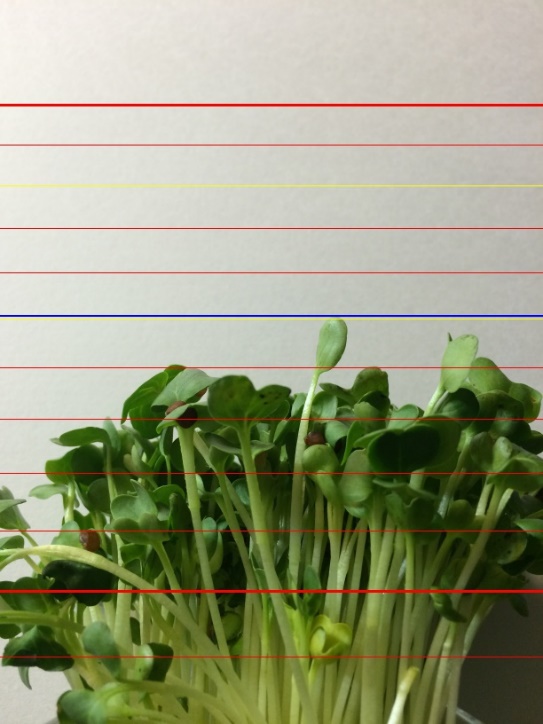
実験１で求めたデータと最長点の値を使用する。

※Iotを想定したシミュレーションソフトを製作しますので、ネットワークを使用する。よって、通信の安定性を高めるため、**元の画像を1/4にリサイズ**する。またリサイズした画像で罫線のある画像を生成する。

**コード：**



出力結果：



**元画像から1/4リサイズした画像**

**罫線のある画像**

612pixel

816

pixel

**３．伸長を求める**

以下の表は処理に必要なデータ表である。

プログラム１の出力結果 (**最長点**)を参考に伸長を求める。

**１４２６**

例外処理：以下、３つの条件

・⑩番目の境界線「479」と同じ値ならば、「13.0cm」である

・⑩番目の境界線「479」より小さければ、「-2.0(測定不可)」である

・⓪番目の境界線「2668」より大きければ、「-1.0(発芽期)」である

これらの条件にどれも当てはまらなければ処理を続行。

「1426」は当てはまらないので続行。

最長点が⑨番目から⓪番目の境界線の各数値以下の場合だったときの添字を探索する。

(以下、探索添字とする)　　「1426」の探索添字は⑤番目

「num[N]-num[N-1]」に「探索添字」を代入し「差」を求める。

「1426」の場合、「209」

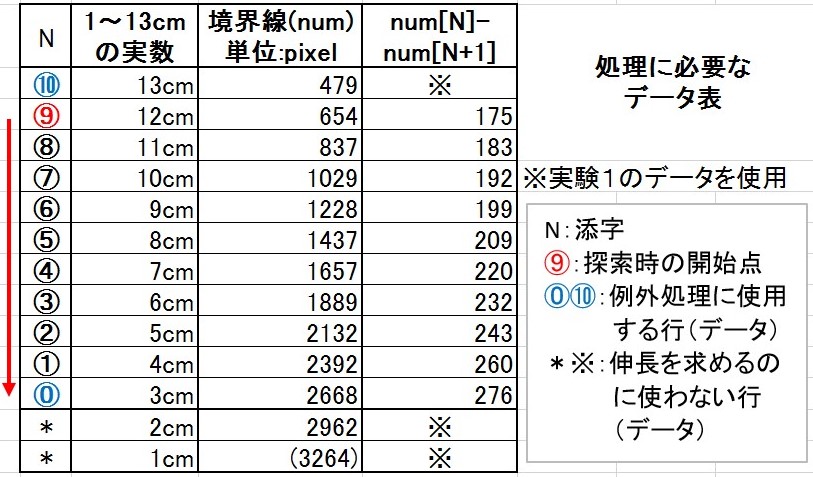
「差」は１㎝分のピクセル数である。また、1cmを「差」で割った値と「num[N]-最長点」を掛けた値、そして、「探索添字」を足した値を浮動小数(以下、仮伸長とする)で求める。

「1426」の仮伸長は、「5.052631578947368」

「仮伸長」を小数点第二位で切り捨て、3（発芽期の3cm）を足した値が伸長である。

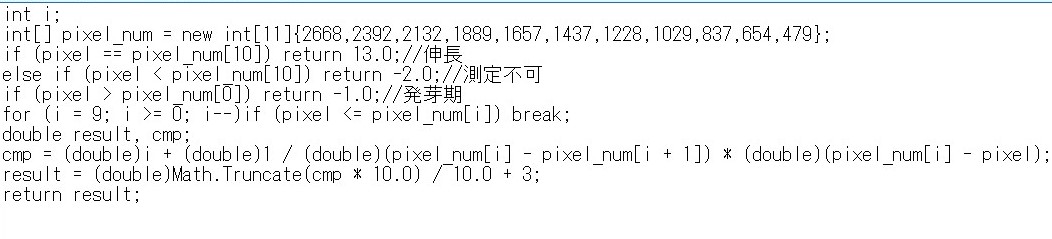
よって「1426」の伸長は「8.0」となる。

使用した植物の実際の伸長は「8.2」だった。

つまり、ある程度の近い値が求められていることが分かる。

以上、方法をプログラムにする。

**コード：**

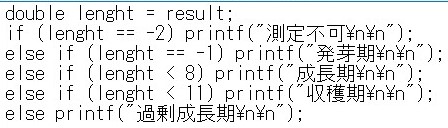


出力結果：「1426」=>「8.0」

**４．カイワレダイコンの各期間を求める**

0cm～3cm未満を発芽期(－１)・3cm以上～8cm未満を成長期・8cm以上～11cm未満を収穫期・11cm以上～13cm以下を過剰成長期・13cm超過で測定不可（－２）とする。

**コード：**



出力結果：「1426」=>「成長期」

**５．植物育成LEDの入り切り求める**

実験１のデータを使用

条件：最長点が「８３７を超過かつ２６６８以下」であれば植物育成LED：ON

　　　でなければ　OFF

**コード：**



出力結果：「1426」=>「植物育成LED：ON」

**＜１～５のプログラムをまとめて実行できる「植物画像処理.exe」作成＞**

(C++/64ビットアプリケーション)

コンパイラ：Visual Studio 2015 community

ライブラリ：Opncv3.1使用(64ビット)

実行ファイル・フォルダ：

・「**./提出/bin/実験２/植物画像処理/植物画像処理.exe**」実行ファイル

・「./提出/bin/実験２/植物画像処理/出力画像」内に５つの画像ファイルを生成

**著作権が関わるファイル：**

「VC＋＋ランタイムライブラリ」

msvcp140.dll・concrt140.dll・vcruntime140.dll

「Opencvライブラリ(bin)」

opencv\_world310.dll

以上、ファイルの著作権は資料１(本書)の「著作権が関わるライブラリ・ファイルの使用について」

記載しています。

※コードレビュについて

「./提出/src/実験２/植物画像処理」内に「植物画像処理.sln」があります。

「Visual Studio 2015 community」で開いてください。

コードを見ることができると思いますが、OpenCV3.1を利用しているので

**エラー非表示・ビルド等をする際には資料１(本書)の「コードレビュについて」=>「①」を参考に設定して下さい。**

ファイル・フォルダ構成：

「(フォルダ)出力画像」

・植物画像処理.exe

・msvcp140.dll

・concrt140.dll

・vcruntime140.dll

・opencv\_world310.dll

「（フォルダ）植物画像処理」

プログラム実行時、フォルダ「出力画像」内に５つの画像が生成される。

**この画像を確認してください。**

**画像に内容についてプログラムの説明している「P６～P１１」参考にしてください**

**実行方法**

フォルダ**「./提出/data/テスト用画像/**」内にある複数枚の画像を使用します。

その中の一つの画像ファイルのディレクトリを引数とする。

複数枚あるので違う画像で数回、実行できる。

コマンド入力：

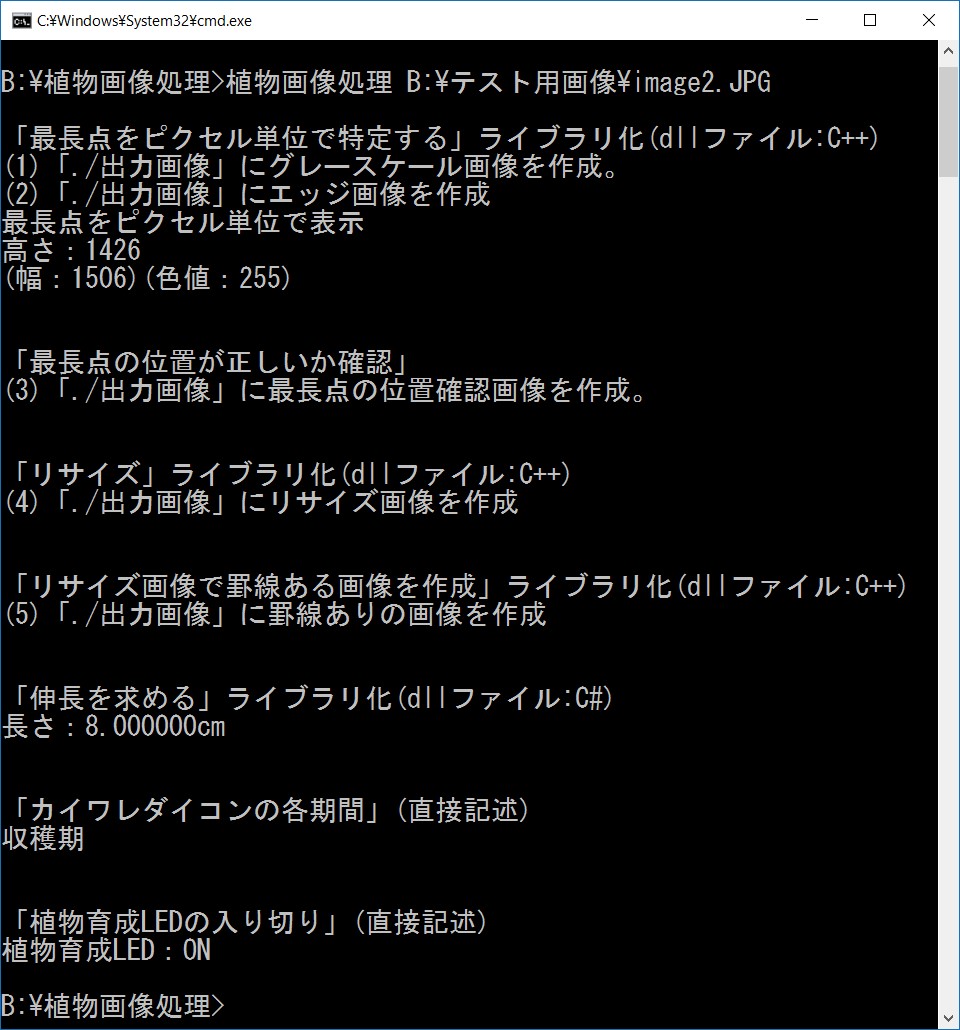
１．コマンドプロント起動

２．カレントディレクトリを「./提出/bin/実験２/植物画像処理」に設定

３．コマンド入力「＞植物画像処理 引数１(./テスト用画像/----.JPG)」

４．テスト用画像から抜き出した植物の情報が画面上に出力・

元の画像から生成した５つの画像が「(フォルダ)出力画像」に出力される。



**出力内容と画像**

**＜植物専用のライブラリの作成＞**

シミュレーションソフト(C#)で動かすため、「.NET対応」であるC++/CLIまたはC#でライブラリを作成。

プログラム１．植物画像からエッジ検出をし、最長点を求める

プログラム２．元の画像を1/4リサイズし、罫線のある画像を生成

プログラム３．伸長を求める

プログラム４．カイワレダイコンの各期間を求める

プログラム５．植物育成LEDの入り切り求める

**※コードレビュについて：**

「./提出/src/ MyLibrary /plant\_library/…」内に各ライブラリのファルダがあります。

「Visual Studio 2015 community」で開いてください。

OpenCV3.1を利用している自作ライブラリはコードを見ることができると思いますが、

**エラー非表示・ビルド等をする際には資料１(本書)の「コードレビュについて」=>「①」を参考に設定して下さい。**

**植物ライブラリ**

プログラム１：

(C++/CLI・64ビットdll)

コンパイラ：Visual Studio 2015 Community

ライブラリ：Opncv3.1使用(64ビット)

dllファイル・フォルダ名：

plant\_pixel.dll

(フォルダ名)「plant\_pixel」// Visual Studio 2015 Community

プログラム２：

プログラム２では、リサイズし、罫線のある画像を出力するプログラムだった。本ソフトのために「リサイズ」「罫線のある画像を出力」に２つに分けて、ライブラリを作成。

(C++/CLI・64ビットdll)

コンパイラ：Visual Studio 2015 Community

ライブラリ：Opncv3.1使用(64ビット)

dllファイル・フォルダ名：

plant\_resizepicture.dll・plant\_gridpicture.dll

(フォルダ名)「plant\_resizepicture」// Visual Studio 2015 Community

(フォルダ名)「plant\_gridpicture」// Visual Studio 2015 Community プログラム３：

このライブラリはOpenCVを使用しないため、C#で作成する

(C#・64ビットdll)

コンパイラ：Visual Studio 2015 Community

dllファイル・フォルダ名：

plant\_lenght.dll

(フォルダ名)「plant\_lenght」// Visual Studio 2015 Community

※プログラム４・５については「シミュレーションソフト」に直接記述する。

**上記より、４つの植物ライブラリを「シミュレーションソフト」で使用する。**