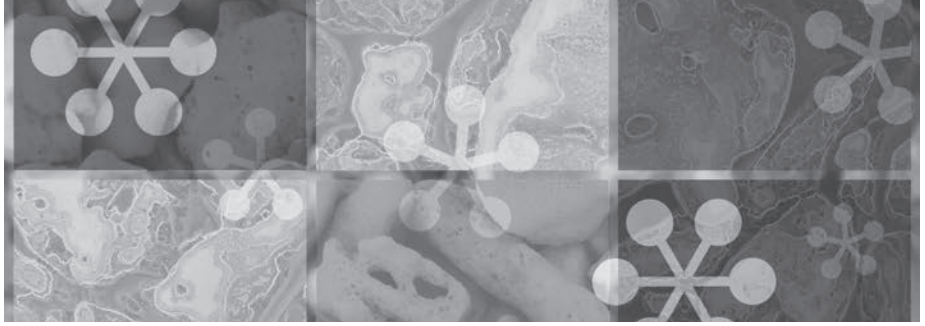
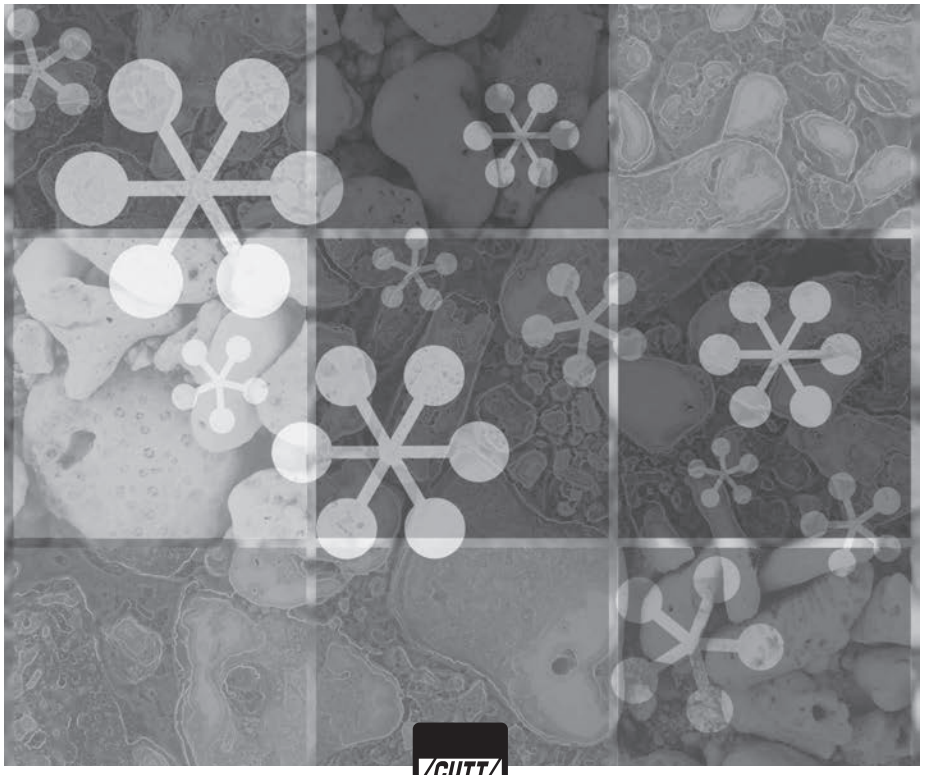


さらに進化した画像処理ライブラリの定番

OpenCV 4
基本プログラミング

北山洋幸◉著



CUTT
カットシステム

■サンプルファイルのダウンロードについて

本書掲載のサンプルファイルは、一部を除いてインターネット上のダウンロードサービスからダウンロードすることができます。詳しい手順については、本書の巻末にある袋とじの内容をご覧ください。

なお、ダウンロードサービスのご利用にはユーザー登録と袋とじ内に記されている番号が必要です。そのため、本書を中古書店から購入されたり、他者から貸与、譲渡された場合にはサービスをご利用いただけないことがあります。あらかじめご承知おきください。

本書で取り上げられているシステム名／製品名は、一般に開発各社の登録商標／商品名です。本書では、™ および ® マークは明記していません。本書に掲載されている団体／商品に対して、その商標権を侵害する意図は一切ありません。本書で紹介している URL や各サイトの内容は変更される場合があります。

はじめに

世界には多くのオープンソースが存在します。その機能は市販のソフトウェアを凌駕するものも少なくありません。OpenCV[※1]もその例にもれず、優秀な画像処理用のオープンソースであり、膨大な関数から形成される画像処理ライブラリ集です。画像処理を行うソフトウェアの開発は容易ではありません。ところが、OpenCV を用いると、少しのコードを記述するだけで、画像処理用のソフトウェアを開発できます。本書は、画像処理プログラミングに縁のなかった人や、画像処理プログラミングに二の足を踏んでいた人へ、画像処理プログラム開発の道案内を行います。OpenCV は、マルチプラットフォームであること、および複数の言語をサポートしていることも大きな特徴です。そして画像だけでなく動画をサポートする強力な機能も備えています。

OpenCV は、一般的な 2 次元の画像処理、ヒストグラム処理、ポリゴン処理、テンプレートマッチング、オプティカルフロー、および顔認識など多様なアプリケーションを開発できる関数群を用意しています。これらの関数リファレンス、ならびに大量のサンプルプログラムは、OpenCV のサイトに紹介されています。また、同サイトには、サンプルコードとともに、それに対するドキュメントまで提供されています。しかし、オープンソースにはつきものですが、最新の情報は、すべて英文で書かれ、サンプルが最新の OpenCV の推奨とは異なる場合も少なくない点に注意が必要です。

OpenCV 4 では、バージョン 3 で行われた大きなモジュール変更などは多くありませんが、G-API モジュールや機能追加、そして dnn モジュールなど deep learning に関する機能に実験的なものも組み込まれました。ただし、これらの追加モジュールについては、機能的な熟成が進んでいるとみられません。そこで、これらの機能を取り上げるには時期尚早と考え、本書の話題から外しました。

本書は、C++ インタフェースで OpenCV を使用します。OpenCV 2 で C++ インタフェースが用意され、OpenCV 3 で C インタフェースがメンテナンス対象外となり、そして十分な移行期間を設けたためか、OpenCV 4 で従来の C インタフェースは完全に削除されました。いまでも C インタフェースを使い続ける人は少なくありませんが、このような背景から、今後 OpenCV を学習する人は C++ インタフェースを前提に学習するのが賢明でしょう。本書は、

※ 1　Open Source Computer Vision Library

プログラミングスタイルだけでなく、パフォーマンス向上を狙った解説や応用についても多くのページを割きます。本書を参考に、最新の画像処理プログラミングの世界へ踏み出してください。

対象読者

- 画像処理プログラミングに挑戦したい人
- レガシーな C インタフェースから C++ インタフェースへ移行したい人
- 性能の高い画像処理プログラミングに挑戦したい人
- 動画処理を行いたい人
- OpenCV を利用した応用プログラムを開発したい人

謝辞

出版にあたり、お世話になった株式会社カットシステムの石塚勝敏氏に深く感謝いたします。

2019 年冬 東大和南公園の北にあるコーヒーショップにて　北山洋幸

■ 参考文献、参考サイト、参考資料

1. OpenCV ドキュメントなどのサイト、http://docs.opencv.org/
2. OpenCV 4.0.0-dev/3.4.4-dev documentation サイト、https://docs.opencv.org/master/
3. Mat の基本処理解説サイト、http://opencv.jp/cookbook/opencv_mat.html
4. OpenCV リファレンス・サンプルや解説サイト、http://opencv.jp/
5. 『さらに進化した画像処理ライブラリの定番 OpenCV 3基本プログラミング』、北山洋幸、株式会社カットシステム
6. 『OpenCL2 入門―メニーコア CPU & GPGPU 時代の並列処理、北山洋幸、株式会社カットシステム
7. 『OpenCV で始める簡単動画プログラミング 第 3 版』、北山洋幸、株式会社カットシステム

■ 本書の使用にあたって

開発環境、および、実行環境の説明を行います。

プラットフォーム

OpenCV はマルチプラットフォームをサポートしています。今回は、Windows を開発・実行環境に採用します。

Windows バージョン

特に Windows バージョンへの依存はありません。開発・実行は Windows 10 Home（64 ビット）を使用しました。

ユーザーアカウント

最近の Windows ではユーザーやアカウントの管理が強化されています。例えば、「標準ユーザー」ではプログラムのインストールやアンインストールは制限されます。コンパイラや OpenCV のインストールやセットアップで警告が出ることがありますので、なるべく「管理者」で実行してください。もちろん、管理者アカウントを使用する場合、危険なこともできますので、十分注意してください。

Visual C++ のバージョンとエディション

すべてを確認したのは、無償の Visual Studio Community 2017 です。Visual Studio Community 2015 やそれ以前の Visual Studio でも問題はないでしょうが、確認は行っていません。特に Visual Studio へのバージョン依存はないと思われますが、使用した OpenCV のバージョンには Visual Studio 2017 と Visual Studio 2015 用の 64 ビットバイナリしか用意されていません。CMake で OpenCV のビルドなどを避けたかったら、Visual Studio Community 2017 の使用を推奨します。

OpenCV のバージョン

OpenCV 4.0.0 を使用します。他のバージョンも多少確認し、問題は起きないことを確認しました。ただし、最終の確認を行ったのは OpenCV 4.0.0 です。OpenCV 4.0.0 には、Visual Studio 2015/2017 の x64 のバイナリしか含まれていませんでした。これ以外の開発環境や 32 ビットで使用したい場合、CMake の解説や OpenCV のビルドを参考にして自身の環境にあったバイナリを生成してください。自身でバイナリを生成する方法は付録 E に記述してあります。

x64 か x86 か

本書では x64 を中心にチェックしました。これは CMake などの手間を省くためです。もちろん、自身でバイナリを生成すれば x86 でも問題ありません。自身でバイナリを生成する方法は付録 E に記述してあります。

例外処理

OpenCV の関数に間違った引数を指定すると、プログラムが異常終了することがあります。本書はごく単純な例外処理を組み込みました。より安全なプログラムとしたいなら自身で異常処理に対応した機構を組み込んでください。異常処理への対応を簡略化したのは、プログラムをシンプルにしたかったためです。

Mat と UMat

OpenCV 3.0 以降では、Mat の代わりに UMat を使用する方が高速に処理できる場合があります。本書では Mat を使用することが多いですが、性能に興味のある人は Mat を UMat へ書き換えてみるのもよいでしょう。

URL

書籍中に記述されているURLは原稿執筆時点のものです。URLの変更やウェブサイトの構造は頻繁に変更されますので、記述したURLが存在するとはかぎりません。もし、ページなどが見つからない場合は、トップページへ移動して探すか、インターネットでキーワードを検索してください。

記述したURLに必ず紹介した内容が記載されていることを保証するものではありません。

■ 用語

用語の使用に関して説明を行います。

カタカナ語の長音表記

「メモリー」や「フォルダー」など、最近は語尾の「ー」を付けるのが一般的になっていますが、本書では統一していません。従来の表現と、最近の表現が混在しています。なるべく統一を心がけましたが、参考資料なども混在して使用しているため、統一が困難でした。

クラスとオブジェクト

本来はインスタンス化しているためオブジェクトと表現した方がよい場所でも、クラスと表現する場合があります。これは文脈から読み取ってください。

InputArrayとOutputArray

OpenCV 3.0以降の関数プロトタイプ宣言は、画像（行列）をInputArrayとOutputArrayで記述している場合があります。そこで、これにならいMat&などと記述した方が適切な場合も、InputArrayとOutputArrayを使用します。詳細については、節を設けて説明しましたので、そちらを参照してください。

画像と行列

C++でOpenCVの画像を保持する場合、Matオブジェクトなどを使用します。これらは基本的に行列を管理します。このため、画像を行列と表現する場合と、その逆の場合もあります。ただし、これらは同じものを指します。

「/」と「¥」

ソースコードを記述する際に、「/」と「¥」のどちらでも構わない場合、「/」を採用しています。なお、本文中でも「¥」が適している場合でも、ソースコードが「/」を使用している場合、「/」で表現しています。

アパーチャサイズとカーネルサイズ

両方とも同じものを指しますが、オリジナルドキュメントが混在して使用しています。なるべくオリジナルドキュメント通りとしたため、用語は統一されていません。

カーネル

フィルタなどのカーネルサイズと OpenCL のカーネルコードの両方で使用しています。混同しないように文脈から判断してください。

ソースリストとソースコード

基本的に同じものを指しますが、ソースリストと表現する場合ソース全体を、ソースコードと表現する場合、ソースの一部を指す場合が多いです。

オブジェクト

インスタンスと表現した方が良い場合でも、オブジェクトと表現している場合があります。両方を、厳密に使い分けていませんので、文脈から判断してください。あるいは、物体を指す場合もあります。

映像とフレーム

カメラから取得した画像を映像と表現する場合とフレームと表現する場合が混在しますが、同じものを指します。これらは文脈から判断してください。

関数とメソッド

本来ならコンストラクタやメソッドと表現した方が良さそうな場合でも、C インタフェースを使用していた時の名残か、関数という表現がオリジナルドキュメントで採用されています。本書も、それに倣って関数という表現を多用します。

動画、画像、フレーム

これらは混在して使用しています。動画は画像の集合です。ある瞬間では、動画も画像です。このため、動画を画像と表現した方が適切な場合があります。また、動画はフレームに分割できますので、フレームと表現する場合もあります。画像とフレームは、ほぼ同様の意味ですが、文章の流れからフレームと表現した方が良い場合、フレームを採用します。

関数とプロシージャ

本来なら統一した方が良いのですが混在して使用しています。主に、関数は OpenCV へ、プロシージャは自身で開発したものへ使用しました。特に使い分けに意味はありません。

■ ライセンス

OpenCV も、一般的なオープンソースに漏れず、下記に示すライセンスが示されています。以降の内容に同意できない場合、OpenCV をダウンロード、コピー、インストール、および使用しないでください。

ただ、通常のオープンソースのライセンス同様、難しい使用条件は付帯されていません。通常、本ライセンスが OpenCV 使用の制限になることは考えられません。ただ、必ず全文に自身で目を通し、内容を確認して使用してください。OpenCV のライセンスについては、パッケージ内の opencv¥license.txt に記述されています。

以下、ライセンス原文です。改行などから読みにくいですが、原文をそのまま掲載しています。

By downloading, copying, installing or using the software you agree to this license.
If you do not agree to this license, do not download, install,
copy or use the software.

License Agreement
For Open Source Computer Vision Library
(3-clause BSD License)

Copyright (C) 2000-2018, Intel Corporation, all rights reserved.
Copyright (C) 2009-2011, Willow Garage Inc., all rights reserved.
Copyright (C) 2009-2016, NVIDIA Corporation, all rights reserved.
Copyright (C) 2010-2013, Advanced Micro Devices, Inc., all rights reserved.

Copyright (C) 2015-2016, OpenCV Foundation, all rights reserved.
Copyright (C) 2015-2016, Itseez Inc., all rights reserved.
Third party copyrights are property of their respective owners.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

* Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

* Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

* Neither the names of the copyright holders nor the names of the contributors may be used to endorse or promote products derived from this software without specific prior written permission.

This software is provided by the copyright holders and contributors "as is" and any express or implied warranties, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose are disclaimed. In no event shall copyright holders or contributors be liable for any direct, indirect, incidental, special, exemplary, or consequential damages (including, but not limited to, procurement of substitute goods or services; loss of use, data, or profits; or business interruption) however caused and on any theory of liability, whether in contract, strict liability, or tort (including negligence or otherwise) arising in any way out of the use of this software, even if advised of the possibility of such damage.

目 次

はじめに iii

第1章 概 要 1

1.1 OpenCV とは 1
1.2 OpenCV 4 の特徴と変更点 2
1.3 OpenCV 3 の特徴と変更点 5

第2章 はじめての OpenCV プログラム 9

2.1 はじめてのプログラム 9
2.2 プロジェクトの 64 ビット対応 21

第3章 マトリックスクラスについて 27

3.1 マトリックスクラス 27
3.2 Mat 29
3.3 UMat 30
3.4 Mat と UMat のデータ交換 33
3.5 InputArray と OutputArray 35

第4章 アフィン変換 45

4.1 フリップ 45
4.2 リサイズ 50
4.3 回転 55
4.4 連続回転 61
4.5 透視投影 64

第5章 グラフィックス 71

5.1 円を描く 71
5.2 画像の上に線を描く 74

5.3　画像の上に四角形を描く 77
5.4　画像の上に楕円や円弧を描く 79
5.5　画像の上に文字を描く 81

第 6 章　色の処理 83

6.1　グレイスケール 83
6.2　輝度平滑化 85
6.3　閾値処理（スレッショルド処理） 87

第 7 章　フィルタ処理 97

7.1　画像反転 97
7.2　ブラー処理 99
7.3　ガウシアン処理 101
7.4　ラプラシアン処理 104
7.5　Sobel 処理 106
7.6　Canny 処理 108
7.7　画像の膨張 112
7.8　画像の収縮 114

第 8 章　画像合成 117

8.1　2 つの画像を加算 117
8.2　2 つの画像の差分 129
8.3　2 つの画像を論理和 131

第 9 章　動画処理 133

9.1　動画表示 133
9.2　動画表示・ファイル対応 136
9.3　グレイスケール表示 139
9.4　輝度平滑化 140
9.5　Canny 表示 141
9.6　動画キャプチャー 145
9.7　動画に画像処理を行い保存 149

第 10 章 オブジェクト検出・除去 159

10.1 コーナー検出 159
10.2 オブジェクト除去（1） 162
10.3 オブジェクト除去（2） 165
10.4 オブジェクト検出 168
10.5 動画のオブジェクト検出 172
10.6 特徴点検出 177
10.7 パノラマ 183

第 11 章 Mat と UMat の性能 189

11.1 加算 189
11.2 減算 200
11.3 乗算 202

第 12 章 OpenCL 205

12.1 加算 205
12.2 減算 217
12.3 乗算 218
12.4 対数 219
12.5 e のべき乗 221

第 13 章 独自カーネルコード 223

13.1 画像を左右反転 223
13.2 カーネルプログラムを分離 237
13.3 フィルタ処理 242
13.4 回転 250

第 14 章 画像比較 259

14.1 2 つの画像を比較 259
14.2 複数の画像から探す 263
14.3 重みテーブルを用いる 266
14.4 ヒストグラムで比較 273

14.5　ヒストグラムを用いて複数の画像から探す 278

第 15 章　画像検索 285

15.1　C++ 285
15.2　ビットマップクラス 286
15.3　プログラム本体 307
15.4　absdiff 316
15.5　matchTemplate 323
15.6　compareHist 328
15.7　画像検索高速化 331

第 16 章　応用 345

16.1　自動トリミング 345
16.2　マウスを使おう 357
16.3　透視投影 378
16.4　オブジェクトのサイズを変更 407
16.5　コンソールを使う 433
16.6　リサイズ 448
16.7　コンソールとマウス 457
16.8　パノラマ 484
16.9　オブジェクト交換 494

付　録 511

A　Visual Studio のインストール 511
B　OpenCV のインストール 518
C　環境の設定 521
D　CMake のインストール 527
E　OpenCV のビルド 532

索　引 542

第 1 章

概　要

1

OpenCV の概要、ならびに OpenCV 4.0 で変更された部分などについて解説します。

1.1 OpenCV とは

OpenCV[※1] は、膨大な関数を用意した画像処理ライブラリ集です。一般的な 2 次元の画像処理、ヒストグラム処理、ポリゴン処理、テンプレートマッチング、オプティカルフロー、および顔認識など多様なアプリケーションを開発できる関数群を用意しています。OpenCV を利用すると、数行のコードを記述するだけで簡単な画像処理プログラムを開発できます。このような環境は単に OpenCV だけがもたらしてくれたわけではなく、ハードウェアやオペレーティングシステムの進歩と相まって提供されています。

OpenCV は当初、C/C++ 言語用に用意されていました。しかし、今ではいろいろな言語から利用できる環境が用意されています。OpenCV で利用可能な機能を示します。

- フィルタ処理
- 行列演算処理
- オブジェクト追跡処理
- 領域分割処理

※ 1　Open Source Computer Vision Library

- カメラキャリブレーション処理
- 特徴点抽出処理
- 物体認識処理
- 機械学習処理
- パノラマ合成処理
- コンピュテーショナルフォトグラフィ処理
- GUI（ウィンドウ表示、トラックバーなど）
- カメラキャプチャ、動画ファイル処理

OpenCV は、2006 年に 1.0 がリリースされて以来順次バージョンアップが繰り返され、本書執筆時点ではバージョン 4.0 がリリースされています。2009 年にメジャーバージョンアップが行われ、バージョン 2.0 がリリースされました。しばらく 2.4.x 系のリリースが続き、C インタフェースと、C++ インタフェース、そして各言語用のバインディングなども用意されました。バージョン 2.x では C++ インタフェースが推奨されていましたが、バージョン 1.x で用意された C インタフェースを使い続ける人も少なくありませんでした。バージョン 3.0 では、C インタフェースは残っていますが、メンテナンス対象外となりました。再び 2018 年の 11 月にメジャーバージョンアップが行われ、バージョン 4.0 がリリースされました。このバージョンから C インタフェースは削除され、今後開発するプログラムは、C++ インタフェースで記述することとなります。しかし、C インタフェース使用者のためかバージョン 3.4.4 も同時にリリースされています。特別な理由がない限りバージョン 4.x を使用し C インタフェースを使用するのは避けるのが賢明でしょう。

1.2 OpenCV 4 の特徴と変更点

前回の 3.0 へのメジャーバージョンアップから約 3.5 年ぶりに、4.0 へのメジャーバージョンアップが行われました。OpenCV がバージョン 4 へ変更されることによって大きな影響を受けるのは、C++ インタフェースへ移行せず C インタフェースを使い続けた人です。バージョン 4 からは、バージョン 1 の多くの C API が削除されました。以降に、バージョン 4 で変更された主要な点を箇条書きで示します。

- C++ 11 ライブラリや C++ 11 準拠のコンパイラを必要とします。CMake を使用する際は、バージョンは 3.5.1 以上でなければなりません。
- OpenCV バージョン 1.x から用意されていた、多くの C API は削除されています。
- core モジュール（XML、YAML、または JSON への / からの構造化データの格納と読み込み）は C++ で完全に再実装されています。また、C API は削除されました。
- 新しい G-API モジュールが追加されています。
- dnn モジュールは、OpenVINO™ toolkit R4 の Deep Learning Deployment Toolkit で更新されています。詳細は OpenCV の DLDT サポート方法を参照してください。
- dnn モジュールには実験的な Vulkan バックエンドが含まれ、ONNX 形式のネットワークをサポートします。
- Kinect Fusion アルゴリズムは、CPU と GPU（OpenCL）用に実装され、最適化されています。
- QR コード検出器とデコーダが objdetect モジュールに追加されました。
- 非常に効率的かつ高品質な DIS 高密度オプティカルフローアルゴリズムが opencv_contrib からビデオモジュールに移行しました。

さらに詳細な変更点を知りたい人は、

- 4.0-alpha（https://opencv.org/opencv-4-0-0-alpha.html）
- 4.0-beta（https://opencv.org/opencv-4-0-0-beta.html）
- 4.0-rc（https://opencv.org/opencv-4-0-0-rc.html）

の変更点を参照してください

マルチプラットフォーム対応

OpenCV バージョン 4 では、以降に示す OS がサポートされます。

デスクトップ環境

　Windows、Linux、Android、MacOS、FreeBSD、OpenBSD

モバイル環境

　Android、Maemo、iOS

一般的なデスクトップ OS をサポートしていますので、パソコンで動作させることはもちろん、モバイル端末用の OS もサポートしていますので、モバイル端末のアプリケーションソフトウェアで利用することも可能です。Linux へも対応しているため、各社の Linux 対応の embedded 機器へも応用が可能でしょう。

サポート言語

OpenCV バージョン 4 では、以下の言語がサポートされています。

C++、Python、Java

C について

C インタフェース（C API）は、バージョン 4 で削除されました。

バージョン 3 で C インタフェース（C API）は、3.0 以降メンテナンス対象外でしたが、バージョン 4 では C++ で記述する必要があります。

サポート画像フォーマット

OpenCV は、以下の画像フォーマットをサポートしています。詳細は公式ドキュメントを参照してください。

表1.1●OpenCVがサポートする画像フォーマット

フォーマット	拡張子
Windows bitmap	.bmp、.dib
JPEG	.jpeg、.jpg、.jpe
JPEG 2000	.jp2
PNG	.png
Sun rasters	.sr、.ras
TIFF	.tiff、.tif
OpenEXR Image	.exr
WebP	.webp
Portable image	.pbm、.pgm、.ppm

サポート画像フォーマットを示しましたが、今後のバージョンアップによって変更されることも多いと予想されるため、参考にとどめてください。サポート画像フォーマットの詳細については実際に使用するバージョンのドキュメントやソースコードを参照してください。オープンソースなどではドキュメントと実際のコードに矛盾があることも少なくありませんので、そのあたりは十分注意が必要です。ソースコードのアップデートにドキュメントが追随できないのは良くあることです。

1.3 OpenCV 3 の特徴と変更点

本節は、既にバージョン 3 を使用中の人、あるいは OpenCV を初めて使用する人（バージョン 4.0 から）は読み飛ばして構いません。ここでは、バージョン 2 から 3 で行われた変更点を紹介します。これらの変更は、バージョン 4 にも引き継がれたものが少なくありません。

OpenCV バージョン 2.4 あたりから OpenCV バージョン 3 を飛ばし、一気に OpenCV バージョン 4 へ移行する人は、OpenCV バージョン 3 で行われた拡張や変更を一読しておくと良いでしょう。バージョン 3 は、バージョン 2 と同様な使い方は可能ですが、モジュール構成などが大きく変わりました。本節では、OpenCV バージョン 3 の特徴について簡単に解説します。説明の都合上、3.0 以降の拡張に限らない項目も含まれます。

モジュール変更

ここではモジュール構成の変更点についてまとめます。

モジュール追加

shape や hal モジュールが追加されました。hal モジュールを使用すると、性能向上を期待できます。このモジュールは今後の拡張が期待されますので、モジュール構成も変更される可能性が高いです。

モジュール削除

一部のモジュールが削除、あるいは移動しています。基本的に ocl モジュールは削除の方向

のようですが（cv::UMat へ統合）、UMat の機能を切り替えるのに ocl モジュールが必要なため、削除されるわけではなさそうです。

■ モジュール細分化

OpenCV 3.x では OpenCV 2.4.x のモジュールの一部が、機能ごとに分割されます。そのため、OpenCV 2.4.x のコードを 3.x でも流用する場合は注意が必要です。特に gpu 関係が廃止され、細分化されます。

なお、highgui と gpu の細分化が大きいため、以降に表で示します。

表1.1●highguiとgpuの細分化の詳細

バージョン 2.4.x まで	バージョン 3.x 以降	説明
highgui	highgui	GUI 関係
	imgcodecs	コーデック
	videoio	動画ファイルの入出力、カメラキャプチャ
gpu	cudaarithm	細かく細分化されます。モジュール名から機能を想像してください。詳細は OpenCV の資料を参照してください。
	cudabgsegm	
	cudacodec	
	cudafeatures2d	
	cudafilters	
	cudaimgproc	
	cudalegacy	
	cudaobjdetect	
	cudaoptflow	
	cudastereo	
	cudawarping	
	cudev	

高速化

ここでは高速化に関する変更点についてまとめます。

T-API

OpenCV 3.0 から、CPU と GPU の実装をカプセル化するための仕組みである Transparent API（T-API）が導入されました。OpenCV ユーザーは、Mat の代わりに、T-API で提供される UMat と呼ばれるデータ構造を用いて実装することで、CPU/GPU のどちらでも動作するプログラムを同一コードで記述できます。T-API を使用すると、プログラムの動作環境で OpenCL が利用できる場合、自動的に OpenCL を用いることで高速化を図れます。以降に、UMat を用いた場合の内部処理の概要を示します。

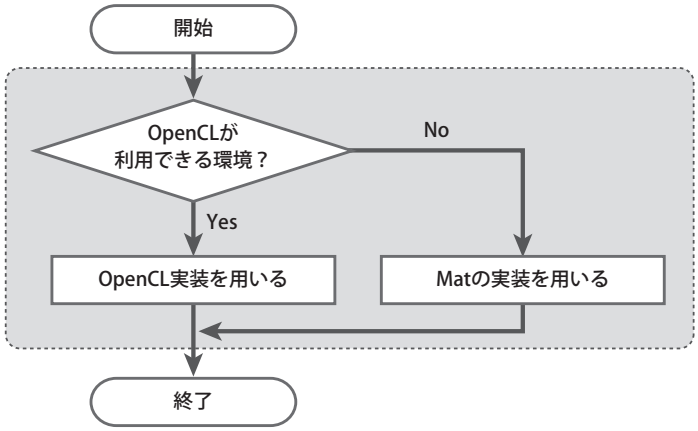


図1.1●UMatを用いた場合の内部処理の概要

UMat クラスは、OpenCL が使える環境かどうかをチェックします。OpenCL が使える環境であれば OpenCL 実装を用い、そうでなければ CPU で処理します。Mat を用いると、最初から CPU で処理します。UMat を使用すると、上図の処理は OpenCV 内部に隠ぺいされるため、OpenCV を利用するプログラマが CPU/GPU を意識する必要はありません。なお、ocl モジュールを使用し、UMat の動作を明示的に制御することも可能です。

IPPICV

Intel IPP のサブセットである IPPICV によって高速化されます。IPPICV は商用・非商用

問わず無料で利用できます。利用する際には、ライセンス文書を参照してください。本書はIPPICVについては解説しません。

■ HAL（Hardware Acceleration Layer）

ハードウェア機能（SSE2、AVX、NEON）を用いた最適化の実装をhalモジュールとして提供しています。OpenCV 3.0時点では一部の機能しか実装されていませんでした。OpenCV 3.1で実装が拡充されました。なお、halモジュールは3.0と3.1で異なっています。

第2章

はじめての OpenCV プログラム

2

OpenCV を使用するのに適した、単純なプログラムを紹介します。これによって OpenCV の入門を習得してください。

2.1 はじめてのプログラム

単に画像を表示するプログラムを開発します。開発するプログラムの機能を、図で簡単に説明します。ファイルから読み込んだ画像を、表示するだけの単純なプログラムです。

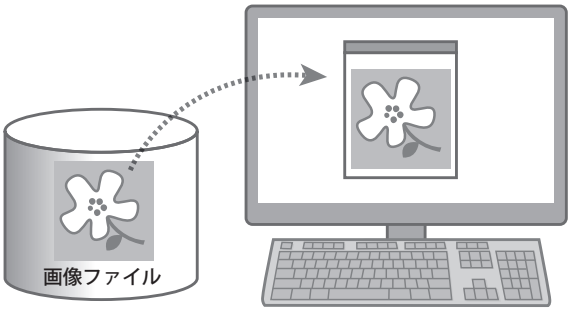


図2.1●プログラムの概要

C++ コンソールプロジェクト作成

Visual C++ のコンソールプロジェクトの作成について簡単に説明します。ここでは Visual Studio 2017 Community で開発する例を示します。

① Visual Studio を起動し、［ファイル］→［新規開発］→［新しいプロジェクト］を選びます。

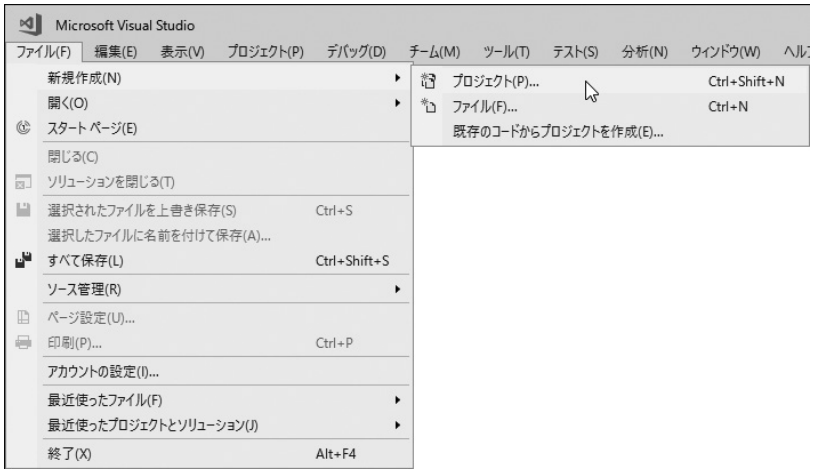


図2.2●新規作成

② さまざまな言語を使用することができますので、Visual C++ を選んでください。ここでは「インストール済み」の「その他」を選びます。右側の欄の「空のプロジェクト」を選択します。プロジェクトの保存場所を変更したい場合、［参照］を押すと「プロジェクトの場所」ダイアログが現れますので、プロジェクトを保存するフォルダ（ディレクトリ）を選択します。

図2.3●「新しいプロジェクト」ダイアログ

③すると、プロジェクトが作成されます。このプロジェクトは、ファイルなどを含まない空のプロジェクトです。

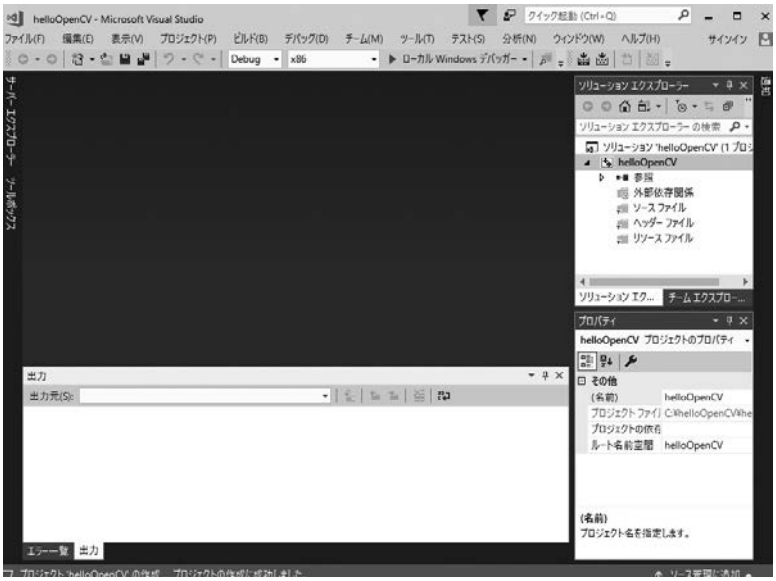


図2.4●空のプロジェクト

④ ソリューションエクスプローラのソースフォルダ上で、マウスの右ボタンを押し、［追加］→［新しい項目 ...］を選びます。

図2.5●新しい項目の追加

⑤「C++ ファイル」を追加します。ファイル名は変更できますが、ここではデフォルトで付けられた名前を使用します。

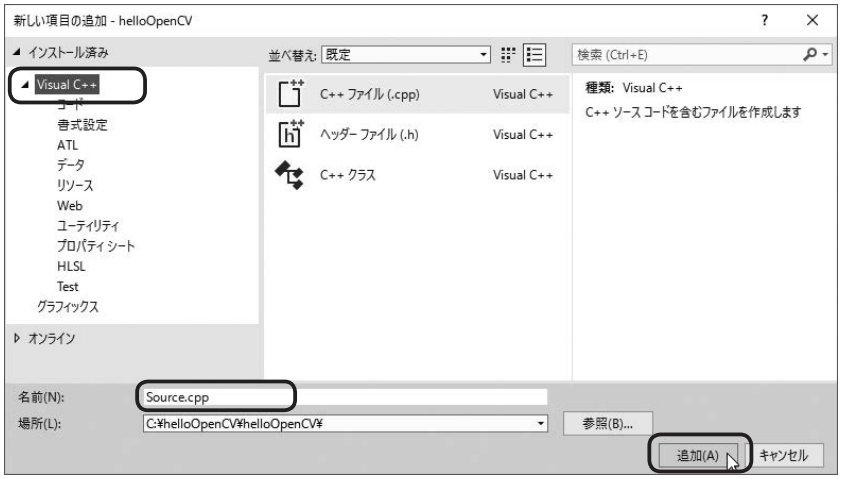


図2.6●「新しい項目の追加」ダイアログ

⑥ ソースファイルを追加した後の、ソリューションエクスプローラを示します。

図2.7●ソリューションエクスプローラ

⑦ 追加したソースファイルにソースコード入力します。以降に、入力したコードを示します。

リスト2.1●ソースリスト（01begin/01helloOpenCV/src/Source.cpp）

```
#include <opencv2/opencv.hpp>                    // OpenCVヘッダ
#pragma comment(lib,"opencv_world400.lib")       // OpenCVライブラリ

int main(int argc, char* argv[])
{
    if (argc < 2)
    {
        std::cerr << "no parameter." << std::endl;
        return -1;
    }

    cv::Mat src = cv::imread(argv[1]);

    cv::imshow("src", src);

    cv::waitKey(0);

    return 0;
}
```

本プログラムは、コマンドラインに与えられた画像ファイルをウィンドウに表示するだけの単純なプログラムです。OpenCV は強力な画像処理ライブラリですが、ユーザーイン

タフェースに関する機能は最低限しかサポートしていません。本プログラムは、標準で OpenCV が備えるユーザーインタフェース機能で、画像をウィンドウ表示します。Visual Studio や OpenCV のインストール、そして環境の設定については、該当する章を参照してください。その解説に従って Visual C++ のインクルードファイルとライブラリのパスが設定されていれば、OpenCV に関係するインクルードファイルやライブラリを、標準のヘッダやライブラリのようにファイル名を記述するだけで指定できます。ライブラリはプロジェクトのプロパティでも設定できますが、忘れないように #pragma で指定します。
プログラムの先頭で、引数の数が適切かチェックし、正常なら cv::imread 関数で入力画像ファイルを cv::Mat に読み込みます。次に、cv::imshow 関数で画像表示します。cv::waitKey 関数でキー入力を監視し、何か入力があるまで表示を続けます。画像を表示しているウィンドウに対し、何かキー入力が行われるとプログラムは終了します。

⑧ ソースコードの入力は完了しましたが、いくつかのオプションを設定する必要があります。OpenCV をインストールした場合、バージョン 3 あたりから 32 ビットのバイナリやライブラリは含まれなくなっています。CMake を使用すれば、各種バージョンの OpenCV をビルドできますが、ここでは標準で用意されているバイナリを使用することとします。そこで、まずプロジェクトを 64 ビットへ変更します。

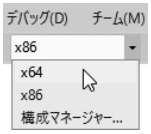


図2.8●64ビット（x64）へ変更

⑨ 次に、プロパティページを表示させ、インクルードファイルとライブラリファイルの存在場所を設定します。プロジェクトを選択した状態で［プロジェクト］→［プロパティ］メニューを選択するか、ソリューションエクスプローラでプロジェクトを選択し、マウスの右ボタンを押します。すると、メニューが現れますので、［プロパティ］メニューを選択します。最初に構成を「すべての構成」へ変更します。そして、「C/C++」→「全般］→「追加のインクルードディレクトリ」へ「**C:¥opencv¥build¥include;**」を、「リンカー」→「追加のライブラリディレクトリ」へ「**C:¥opencv¥build¥x64¥vc15¥lib;**」を入力します。これで、OpenCV のプログラムをビルドする設定が完了です。

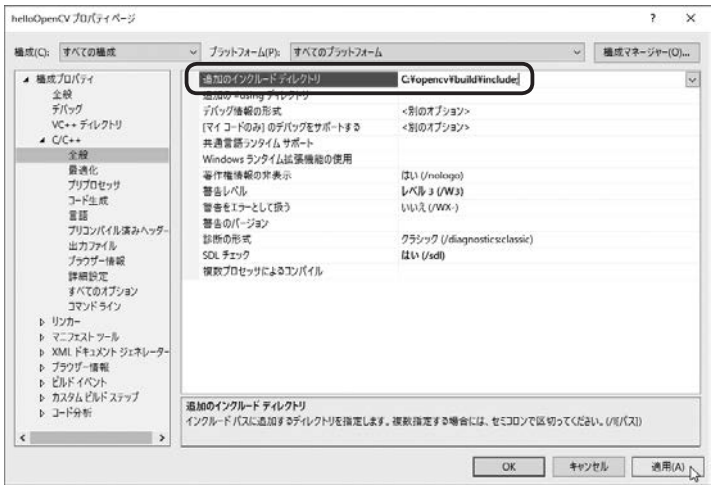


図2.9●インクルードディレクトリ設定

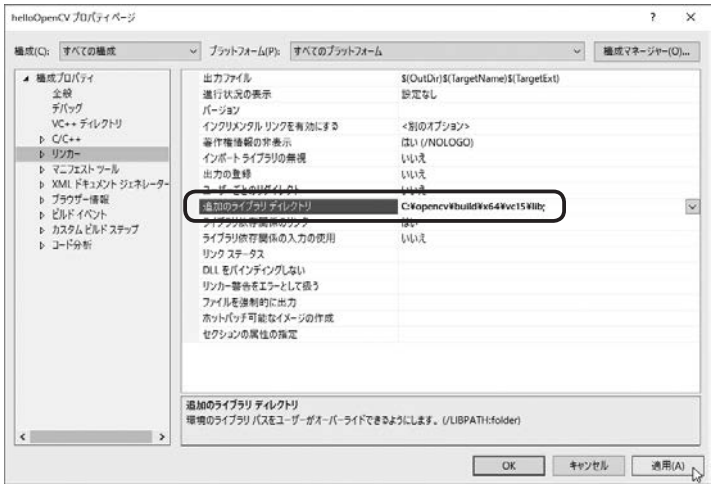


図2.10●ライブラリディレクトリ設定

OpenCV のディレクトリ

インクルードディレクトリやライブラリディレクトリにしているディレクトリは OpenCV のバージョンや、OpenCV をインストールしたときに変わります。これらのディレクトリは自身の環境に合わせて変更してください。

⑩ 準備が完了しましたので、プロジェクトをビルドします。ビルド前に、構成を Debug から Release に変更します。これを行わなければ Debug バージョンの実行形式ができます。プログラムを確認するだけなら、Debug でビルドしても構いません。

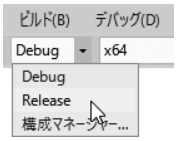


図2.11●DebugからReleaseに変更

⑪ この状態でビルドを行えば、プログラムができあがります。

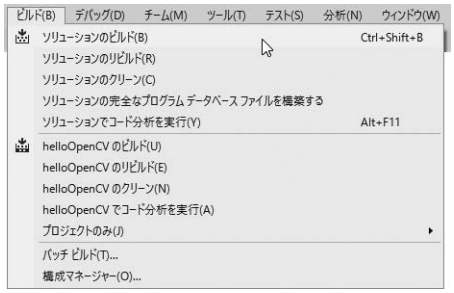


図2.12●ビルド

このとき、OpenCV が対象のバイナリやライブラリを提供していないと、リンカーで「ファイルを開くことができない」旨のエラーメッセージが表示されます。この問題を解決するには、CMake を使用して対象環境のバイナリを生成する方法と、デフォルトで提供されている環境を使用する方法の 2 つがあります。はじめの方法をとる場合は、付録 D「CMake のインストール」および付録 E「OpenCV のビルド」を参照してください。後の方法をとる場合は、本例で示したように「64 ビット」の環境を使用してください。

x86 が未サポート

ダウンロードした OpenCV に x86 のバイナリやライブラリファイルが含まれていない場合があります。そのような場合は CMake を使用して自身で生成するか、それが面倒なら x64 のみでプログラムを開発してください。