

Epson Moverio Sample App

Ryo Mitsumori
2014/12/11

Agenda

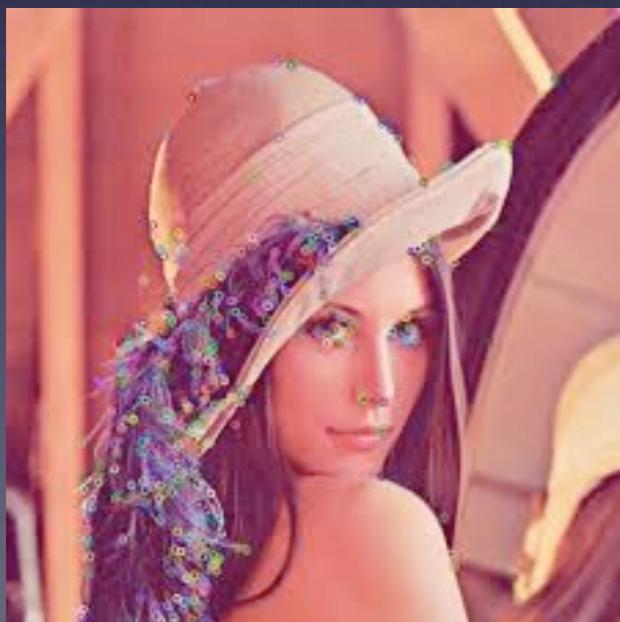
- ・ アプリ概要
 - フロー、仕様、問題点 など
- ・ Moverio本体セットアップ
- ・ 開発環境構築
 - Moverio本体
 - Android開発用Eclipse(Mac) - windowsの人ごめんなさい…
 - OpenCV
 - NDK開発(C, C++)環境設定
- ・ 実装説明(概略)
- ・ 参考サイト など

アプリ概要

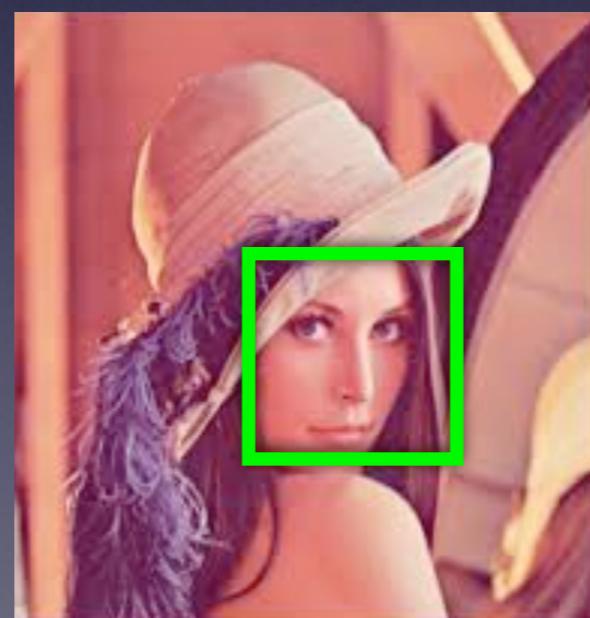
MoverioOpenCVSample.apk

実行可能な処理

- ・ Moverio付属のカメラで取得した下記の処理をリアルタイムで行い、シースルーパンに重畳する
 - 位置あわせ : Java で実行
 - ORB特徴点検出 : Java, C++どちらでも実行可
 - 顔検出 : C++ で実行
 - Canny Edge検出 : Java で実行



特徴点検出

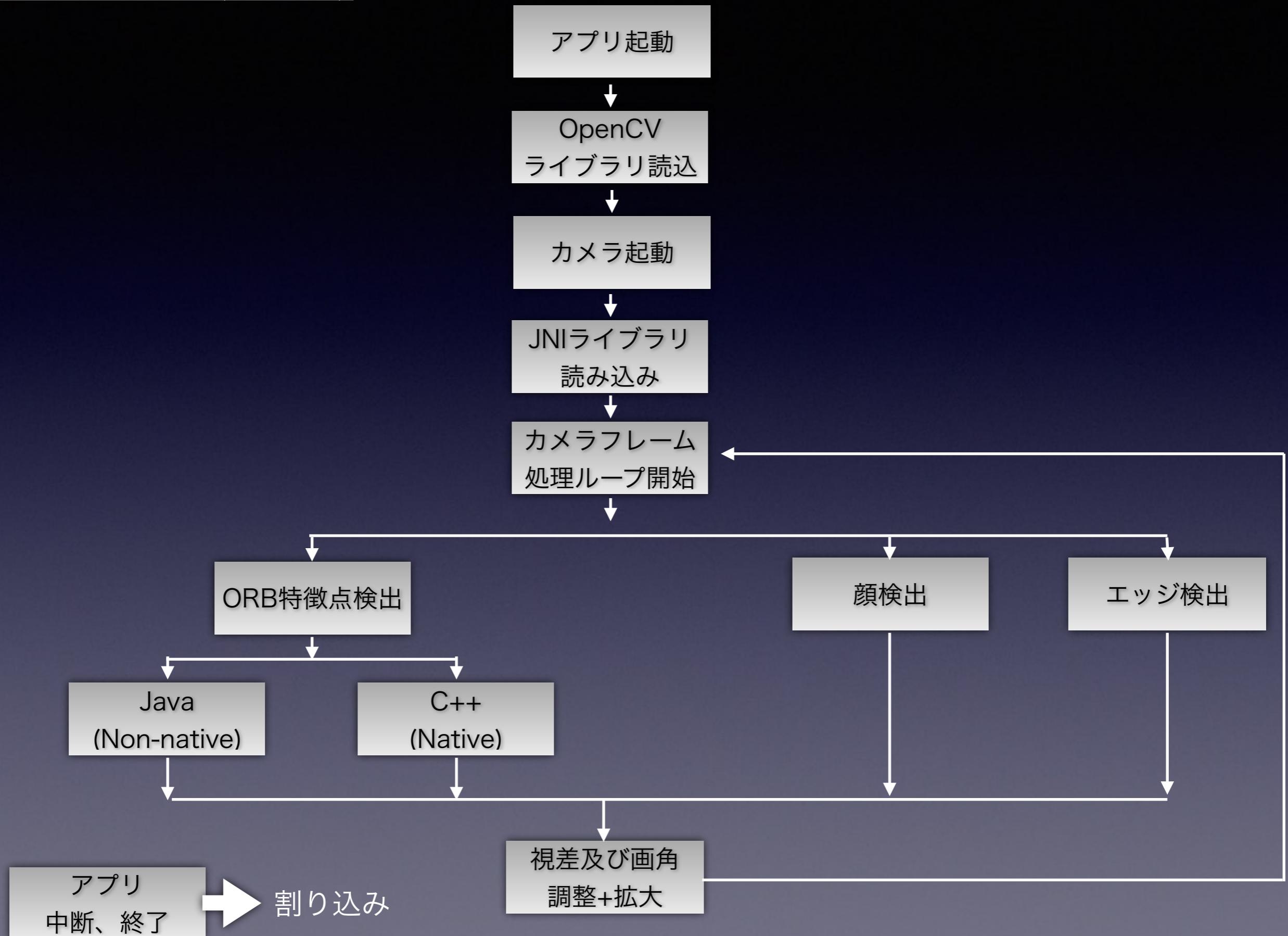


顔検出



Canny

処理フロー(概要)



アプリ仕様

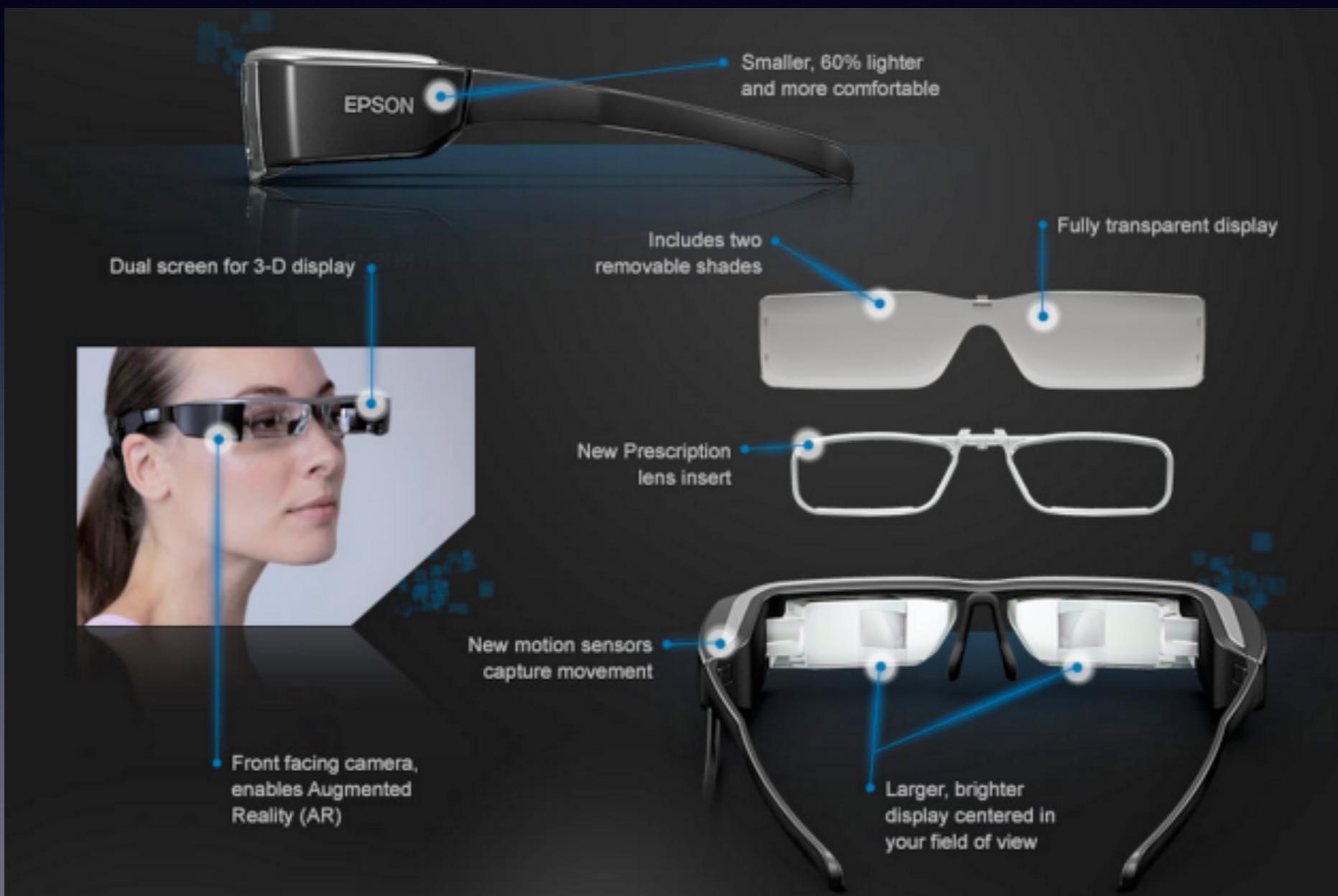


MoverioでSee Through重畳 を実施する際の問題点

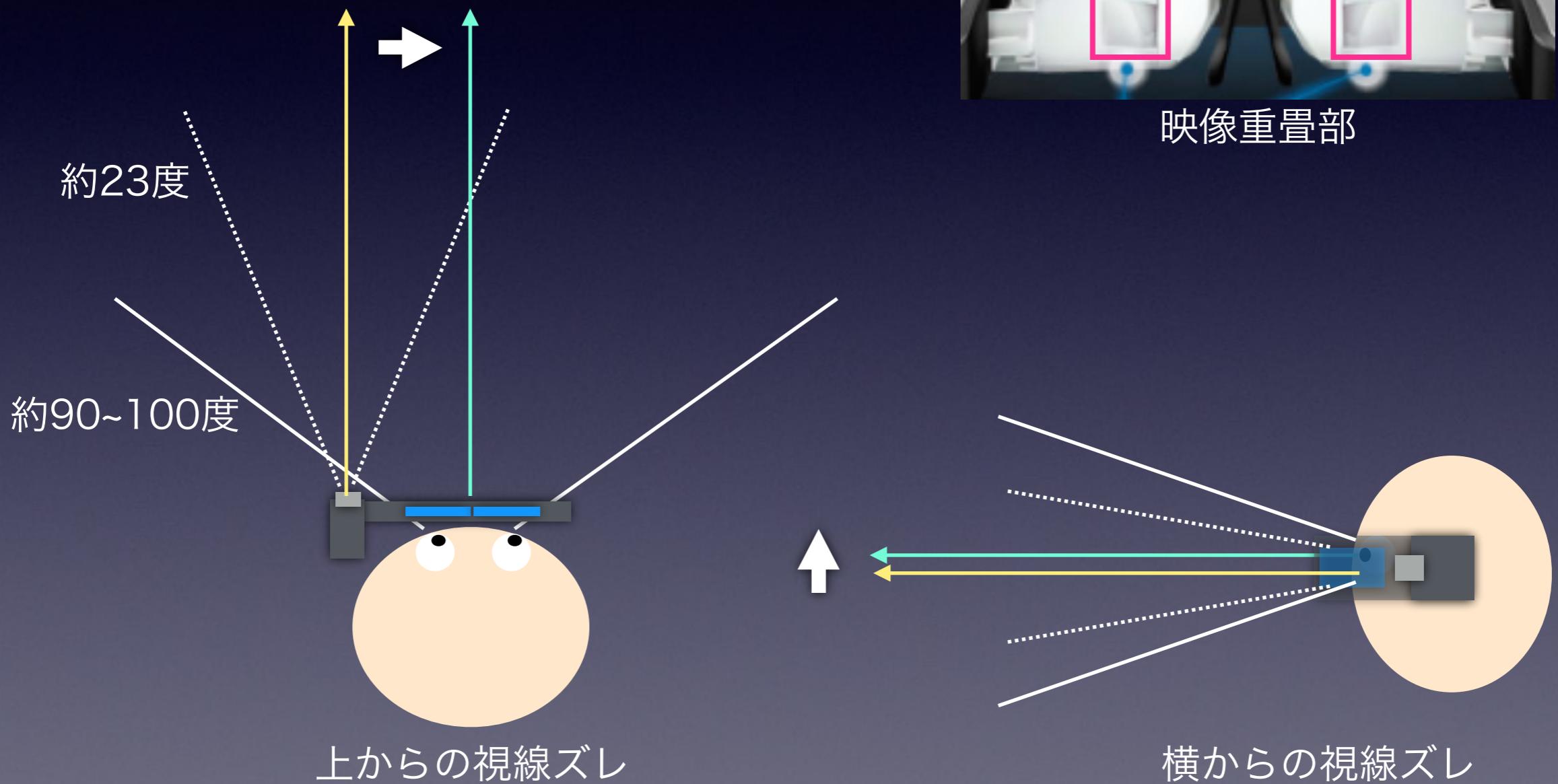
問題点

カメラと目の位置が異なることによる視差
カメラと目の捉える画角差 がある

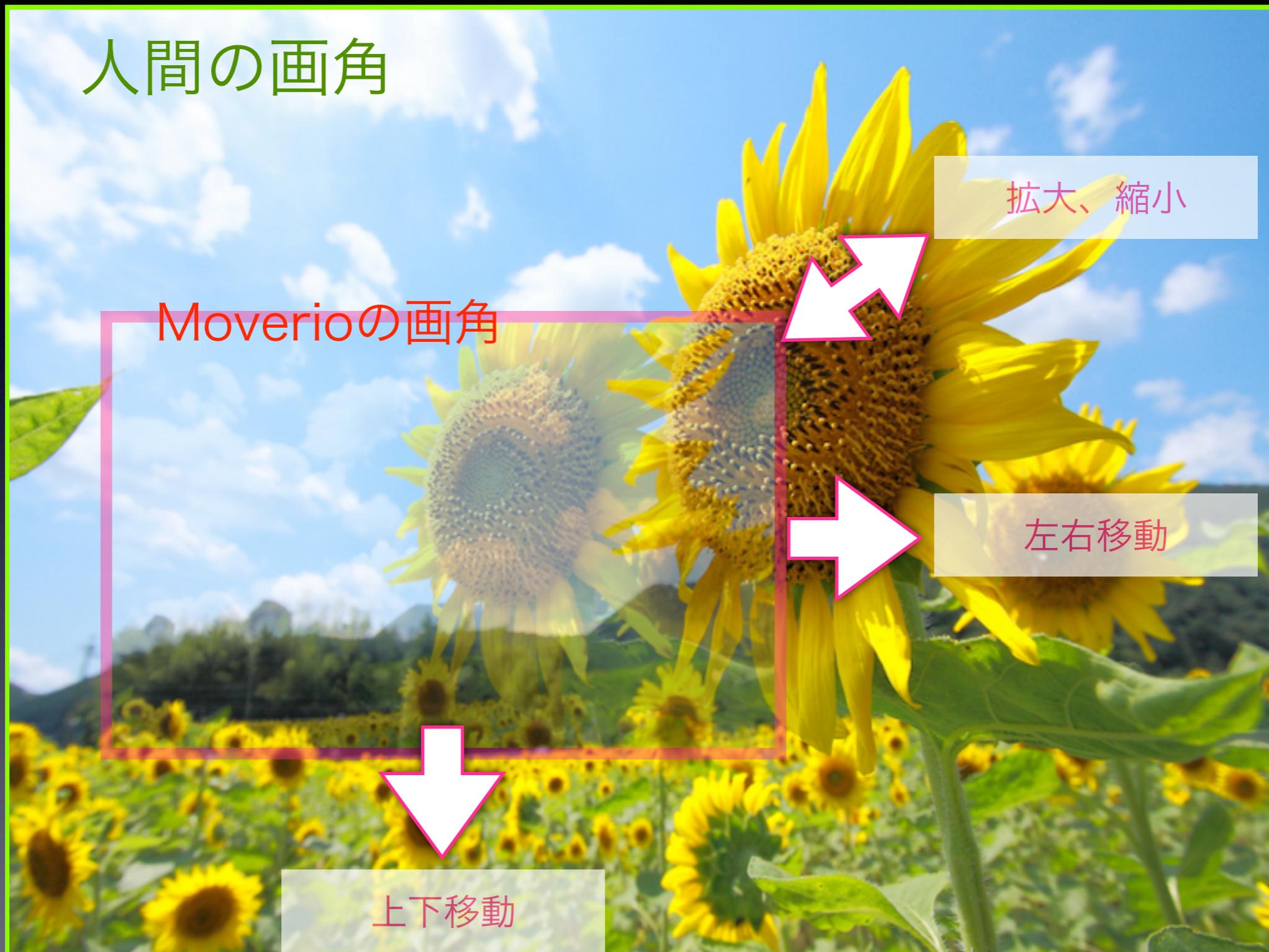
Moverio HW仕様図



上下方向ともに実際の視線とカメラの視点の中心をあわせて、さらに映像重畠部と人間の目が捉えるモノの大きさを揃える必要がある



位置合わせ



イメージ図

その他

- ・ https://github.com/rm0426github/moverio_opencv_sample_appにコードをpushしてあるので、pullしてください。(バグとか見つけたら修正しときます。)または下記からDLしてください。
<https://dl.dropboxusercontent.com/u/41262723/src-moverio-opencv-sample-app.zip>
- ・ Moverioのカメラから対象物までの距離を約1~2m程度としてチューニングした値(x, y軸への移動量、拡大率)を初期値としてある
対象物までの距離が変わると、位置がずれてしまうので調整が必要
- ・ Moverioのカメラ画素数がVGA640px*480pxなので拡大率が3.2倍になるようにチューニングをすると200px*150pxの解像度になる
→ 高解像度な入力画を必要とする処理にはむかない
実際に本アプリでも顔検出は正常に動作しているが、かなり厳しい
- ・ 3.2倍とした理由は640, 480の公約数から最もそれらしく見えるため
(理想は倍率も自由に変更できるようにするのがよいが、本アプリでは固定)

その他

- Android 4.0(ICS)以降であれば、スマフォ上でも実行可能
 - OpenCV Managerをインストールしているのが前提

<https://play.google.com/store/apps/details?id=org.opencv.engine>

※ただし、レイアウトがMoverio特化になっているのでボタンの配置などが
イケてない・・。相対位置に置いたり、レイアウトxmlファイルを切替などすると
スマートになるのだが時間が足りず・・(´・ω・｀)

- 拡大倍率の変更や、処理ももう少し高速化できるところなど色々あるが、
今回は”トータルで動く”を目的としたので、そこまで頑張っていません。
コードに色々とコメントがあるので、プログラしながらスマートにしてみてください。
バグがあっても生暖かく見てあげてください。。

Moverio/Android/OpenCV/NDK 開発環境設定など

Moverio本体開発の前手順

- ・ 開発者登録
 - <https://moverio.epson.biz/>
- ・ 開発用firmのDL, install
 - Dev Siteにログイン後、左欄の”技術情報”からpdfをDLして手順を追う
(MoverioSDKのDL、本体Update方法、開発手順 など)

The screenshot shows the 'MOVERIO Apps Market' Dev Site interface. The left sidebar has a yellow circle around the 'Technical Information' link. The main content area shows a list of technical documents with their names, file sizes, and file names.

名称	ファイル容量	ファイル名
ユーザーズガイド	10,373,651 Byte	H560_UG_JA.pdf
開発者様向け技術資料	4,949,189 Byte	BT200_TIW1405CJ.pdf
開発者用システムソフトウェアリリースノート	3,122,630 Byte	BT200_RLND110A_JA.pdf
開発者用システムソフトウェア運用手順書	6,646,066 Byte	BT200_DOS1409_JA.pdf
MOVERIO ADB USBドライバインストール説明書 (開発者用システムソフトウェア向け)	578,874 Byte	BT200_ADB1406A_JA.pdf
技術情報FAQ	—	公開中(Rev.1.0)
SDKダウンロードページ	—	SDKダウンロードページ
アプリ公開申請用チェックリスト(pdf)	—	appcheck_ja.pdf
アプリ公開申請用チェックリスト(docx)	—	appcheck_ja.docx

MoverioDevSite

Android開発環境設定

- ・ 必要なもの
- ・ Eclipse (統合環境)
- ・ Android SDK (Java開発)
- ・ Android NDK (C/C++開発) : 後述
- ・ Debugモード(開発者モード)になった端末
(Moverioのみでの開発はつらい、、Xperiaで動作確認してました)
- ・ Android Developer SiteにAPIの説明や導入など色々と記載があるので参考にしてください
<http://developer.android.com/index.html>

※Android開発導入は基本的にググると出てくるので、詳細はググって下さい。
ADBの設定や、エミュレータ、USB driver設定などやることが多いです。

色々とパッケージになったEclipse ADTのDL

<https://developer.android.com/sdk/index.html?hl=i>



MoverioDevSiteで配布されている
pdfにも記載あり

Eclipseの公式HPから取得したEclipseでもよいが、色々pluginのDLなどをしないといけないので面倒
公式HP: <https://www.eclipse.org/downloads/>

OpenCV開発環境設定

- 2014/12/08現在でAndroid向けSDKは2.4.10
 - <http://opencv.org/downloads.html>
- Android端末上でOpenCVを使用する方法は下記二通りある
(詳しくはググってください)
 - 1. Libごとごっそりapkファイルにいれてしまう
 - 2. Libが入ったOpenCVManagerアプリを別で端末にDLしておき、それを参照する

本資料およびアプリでは、2.を用いる

- ・ OpenCVManager.apkを利用する場合は、アプリサイズが小さくなるメリットがある反面、アプリとManagerのOpenCVのversionが異なると、実行エラーになることもあるので注意。



Google play

参照：<http://rest-term.com/archives/3010/> など

- ・ MoverioはGoogle Playに接続ができないので、OpenCVManagerをインストールするのに少しコツがいる。
- ・ 手持ちのAndroid端末(Nexus, Xperia, Galaxyなど)にGoogle Play経由でDLをして、その端末から
<http://xxaxxfanxx.blog.fc2.com/blog-entry-106.html>の手順で.apkファイルを引き抜く
- ・ MicroSDにその.apkファイルをコピーしてMoverioに挿す
- ・ コンソールからadbコマンドで.apkファイルをインストールするか、
<http://www.epson.jp/download/moverio/install/>の手順でインストールする

SlideShare(<http://www.slideshare.net/>)で参考にしたサイト例

- Moverio BT-200チュートリアル
<http://www.slideshare.net/ksasao/bt200>
- PlayCanbasでアプリ作ってみた
<http://www.slideshare.net/rerofumi/pc-bt200?related=1>
- Win8+OpenCVでMoverioのアプリ作ってみた
<http://www.slideshare.net/rerofumi/pc-bt200?related=1>

ググっても全然情報が出てこないが、SlideShareだとぼちぼち情報あり

NDK(JNI)開発環境設定

- ・ Android上でC/C++で実装したコードを動作させるために必要な設定、環境
 - 研究で使っているC/C++で書かれたコードの移植が可能になる

※2014/12/08現在 DLしてきたSDK内に格納されているNativeActivityは
端末にインストールして実行しても、 予期せぬエラーがうんぬん言われて動かない
(ひどい・・)

- ・ NDK設定：<https://developer.android.com/tools/sdk/ndk/index.html>

デバッグはEclipse上では、 VisualStudioくらいリッチにできないので、 もしかしたらC/C++の開発はVSでやっておいて、 最後の最後に移植するのがよいのかもしれない。(MacならXCodeを使う。)

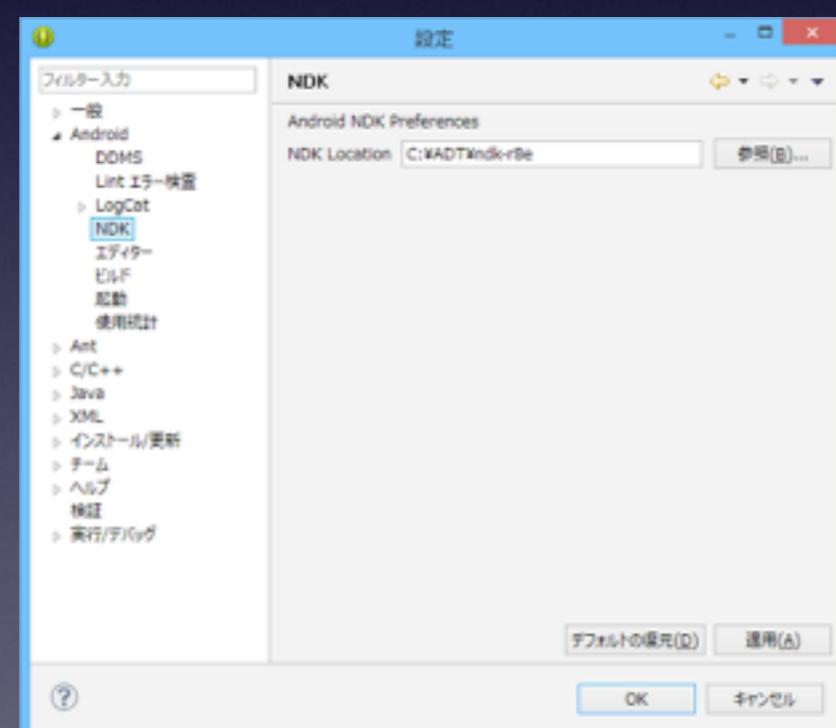
NDKのビルドはコンソールからする必要があるので、 ndk-build実行ファイルへのパスを通しておく必要がある

NDK(JNI)開発環境設定

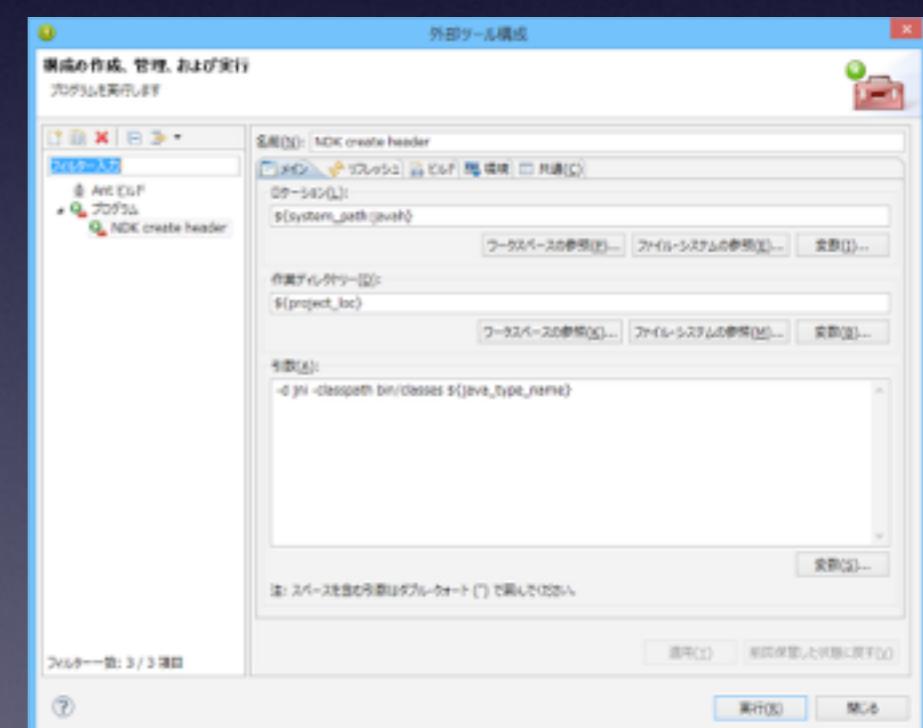
- Android NDKを使った開発環境の構築
 - <http://note.chiebukuro.yahoo.co.jp/detail/n136598>
- Android開発をJava 7で行う
 - <http://note.chiebukuro.yahoo.co.jp/detail/n191253>



NDK-pluginのDL



NDKビルドをEclipseで設定

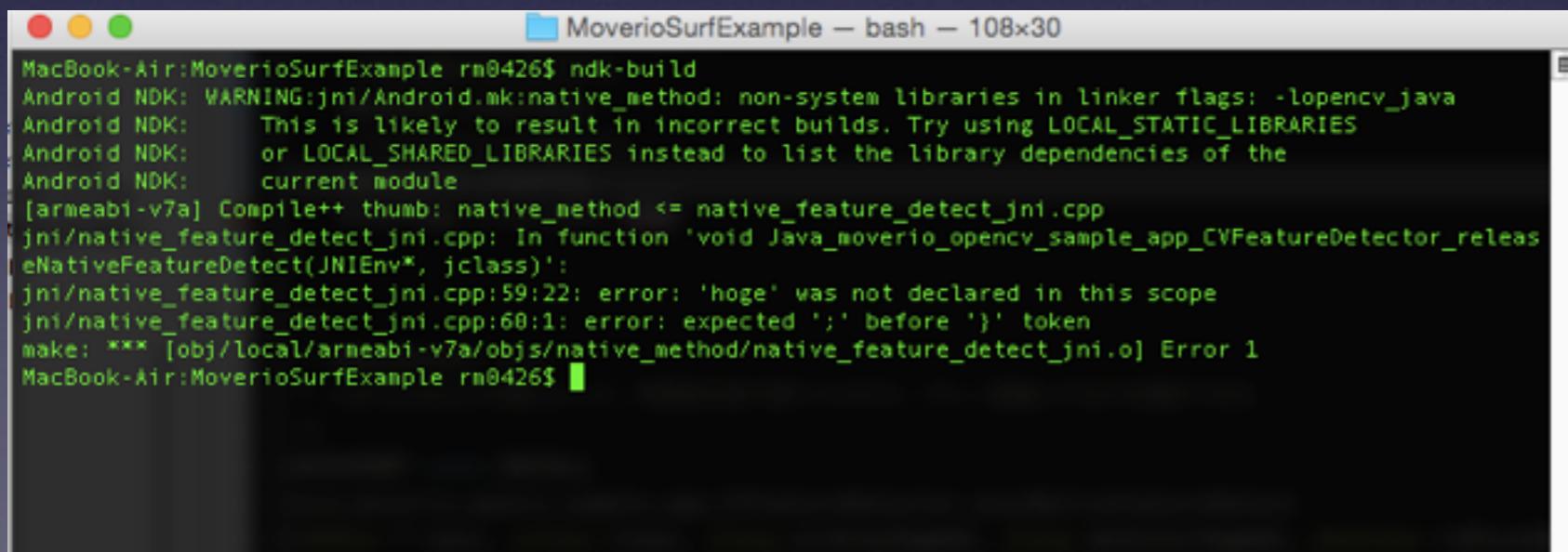


項目	設定
名前	NDK create header (適当な名前で良い)
ロケーション	\${system_path:javah}
作業ディレクトリー	\${project_loc}
引数	-d jni -classpath bin/classes \${java_type_name}

.hppファイル自動生成設定

※ただし、Eclipse@Macでは、前頁のndk-buildの設定や.hppの自動生成設定が正常に動作しないため、下記で対応した

- ndk-build: .bash_profileにndk-buildへのパスを通してterminalから実行
- .hppファイル作成: \$ javah -o jni/(ファイル名).hpp -classpath bin/classes <パッケージ> を実行前提として
 1. (ファイル名).cppは先にEclipseないしは、コンソールから作成しておく
 2. (nativeメソッドを使用するクラス).javaの中に使用するnativeメソッドの宣言をしておく



```
MacBook-Air:MoverioSurfExample rm0426$ ndk-build
Android NDK: WARNING:jni/Android.mk:native_method: non-system libraries in linker flags: -lopencv_java
Android NDK:     This is likely to result in incorrect builds. Try using LOCAL_STATIC_LIBRARIES
Android NDK:     or LOCAL_SHARED_LIBRARIES instead to list the library dependencies of the
Android NDK:     current module
[armeabi-v7a] Compiling++ thumb: native_method <= native_feature_detect_jni.cpp
jni/native_feature_detect_jni.cpp: In function 'void Java_moverio_opencv_sample_app_CVFeatureDetector_releas
eNativeFeatureDetect(JNIEnv*, jclass)':
jni/native_feature_detect_jni.cpp:59:22: error: 'hoge' was not declared in this scope
jni/native_feature_detect_jni.cpp:68:1: error: expected ';' before '}' token
make: *** [obj/local/armeabi-v7a/objs/native_method/native_feature_detect_jni.o] Error 1
MacBook-Air:MoverioSurfExample rm0426$
```

ndk-buildをterminalから実行した例

.hppや.cppファイルの中で#importで色々なhppファイルなどをimportする際に、“そんなファイルはないです”エラーが出た際には、対象のプロジェクト右クリック→propertiesからC/C++ → paths and Symbolsをまず疑う

OpenCVのサンプルプロジェクトのものと比較して、何へのパスを通さないといけないかを比較するとよい

※SDKの位置、環境変数設定などが開発環境によって異なるので、ビルドエラー、参照エラーが起こるなどの場合はこの辺りがすごく怪しい

```
8 #include <native_feature_detect_jni.hpp>
9 #include <opencv2/features2d/features2d.hpp>
10 //##include <opencv2/nonfree/features2d.hpp>
11 //##include <opencv2/nonfree/nonfree.hpp> //SURF
12 #include <opencv2/core/core.hpp>
13 #include <opencv2/highgui/highgui.hpp>
14 #include <string>
15 #include <vector>
```

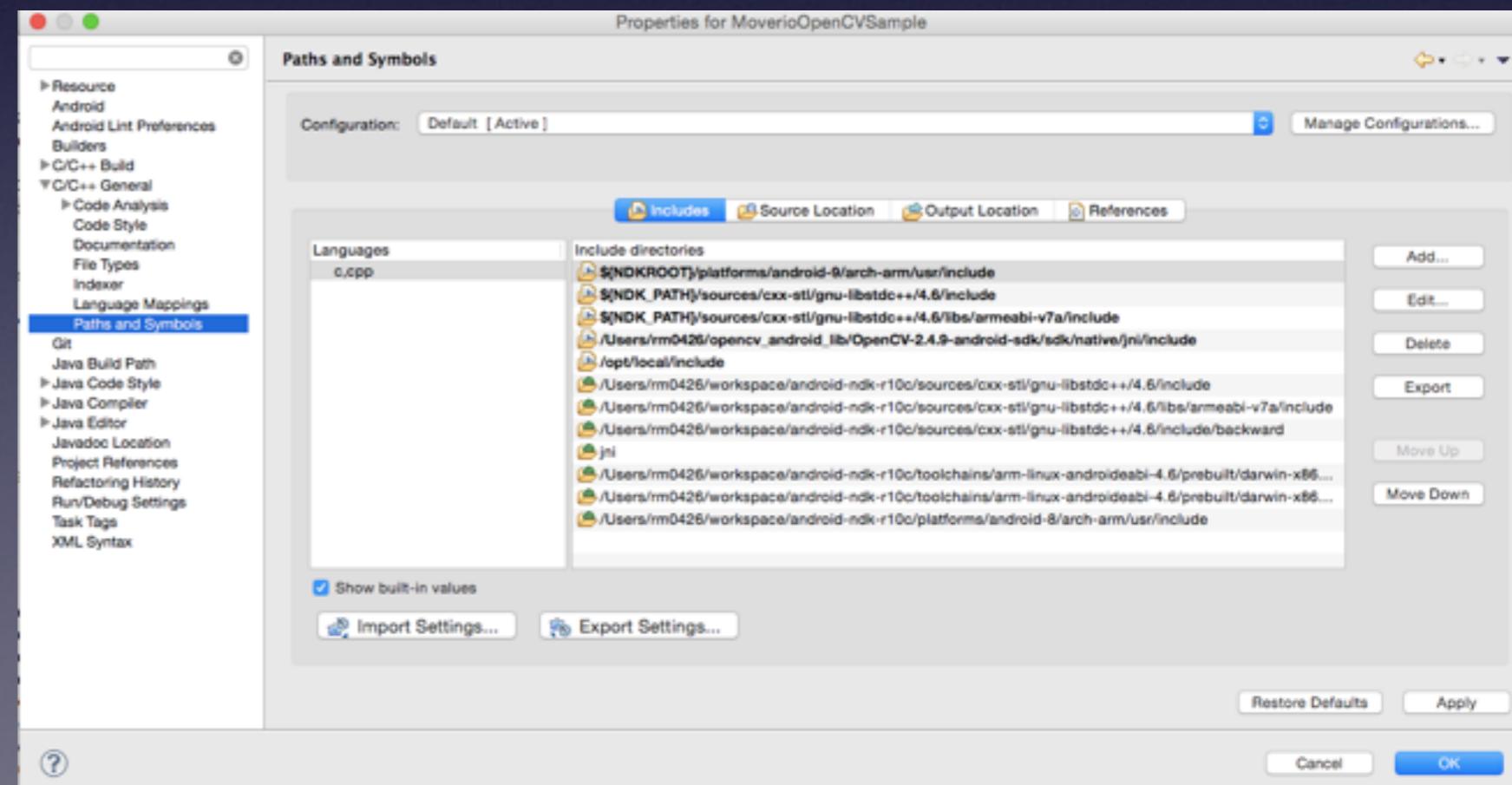
#importの例

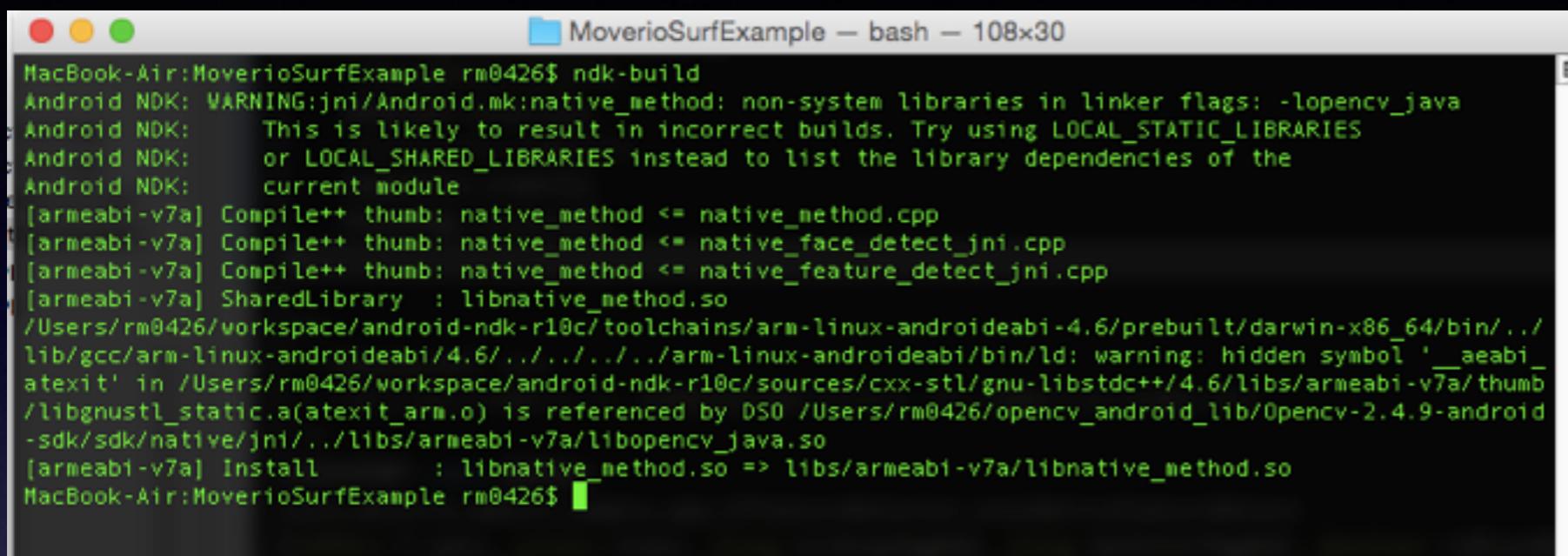
(参照)

algorithm.hppがimportできない

コンパイルエラー

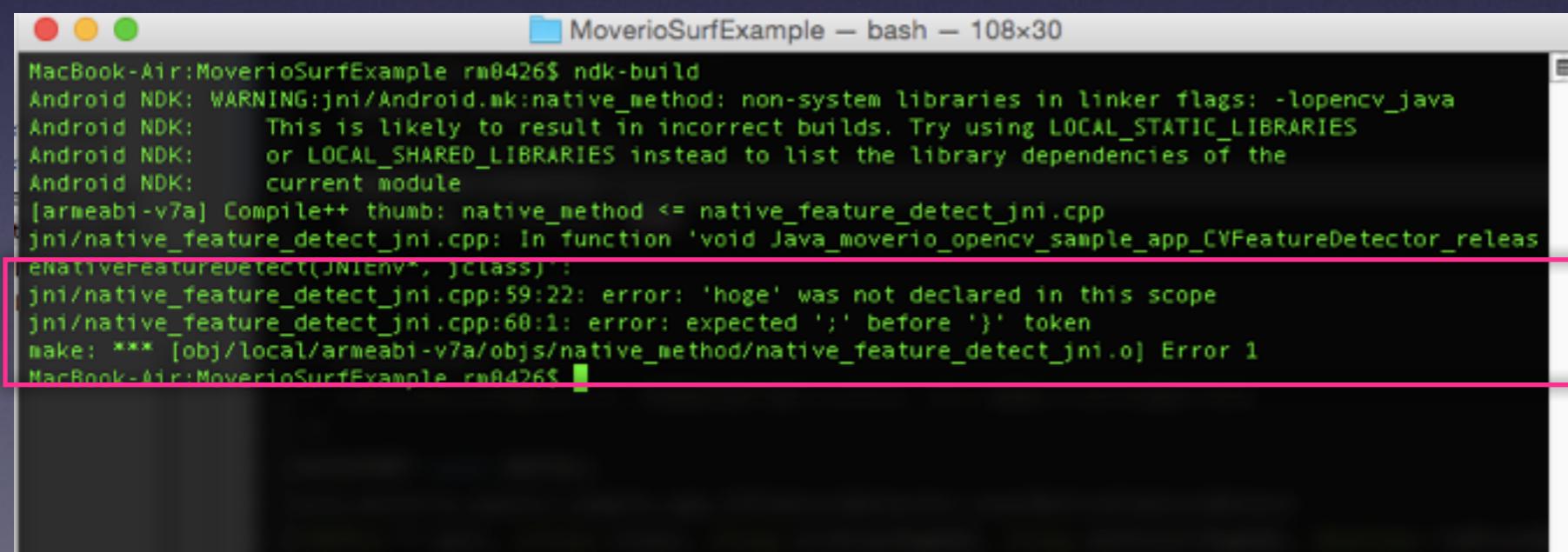
<http://bit.ly/1s6GgfJ>





```
MacBook-Air:MoverioSurfExample rm0426$ ndk-build
Android NDK: WARNING:jni/Android.mk:native_method: non-system libraries in linker flags: -lopencv_java
Android NDK:     This is likely to result in incorrect builds. Try using LOCAL_STATIC_LIBRARIES
Android NDK:     or LOCAL_SHARED_LIBRARIES instead to list the library dependencies of the
Android NDK:     current module
[armeabi-v7a] Compile++ thumb: native_method <= native_method.cpp
[armeabi-v7a] Compile++ thumb: native_method <= native_feature_detect_jni.cpp
[armeabi-v7a] Compile++ thumb: native_method <= native_feature_detect_jni.cpp
[armeabi-v7a] SharedLibrary : libnative_method.so
/Users/rm0426/workspace/android-ndk-r10c/toolchains/arm-linux-androideabi-4.6/prebuilt/darwin-x86_64/bin/...
lib/gcc/arm-linux-androideabi/4.6/.../.../.../arm-linux-androideabi/bin/ld: warning: hidden symbol '_aeabi_
atexit' in /Users/rm0426/workspace/android-ndk-r10c/sources/cxx-stl/gnu-libstdc++/4.6/libs/armeabi-v7a/thumb
/libgnustl_static.a(atexit_arm.o) is referenced by DSO /Users/rm0426/opencv_android_lib/opencv-2.4.9-android
-sdk/sdk/native/jni/.../libs/armeabi-v7a/libopencv_java.so
[armeabi-v7a] Install      : libnative_method.so => libs/armeabi-v7a/libnative_method.so
MacBook-Air:MoverioSurfExample rm0426$
```

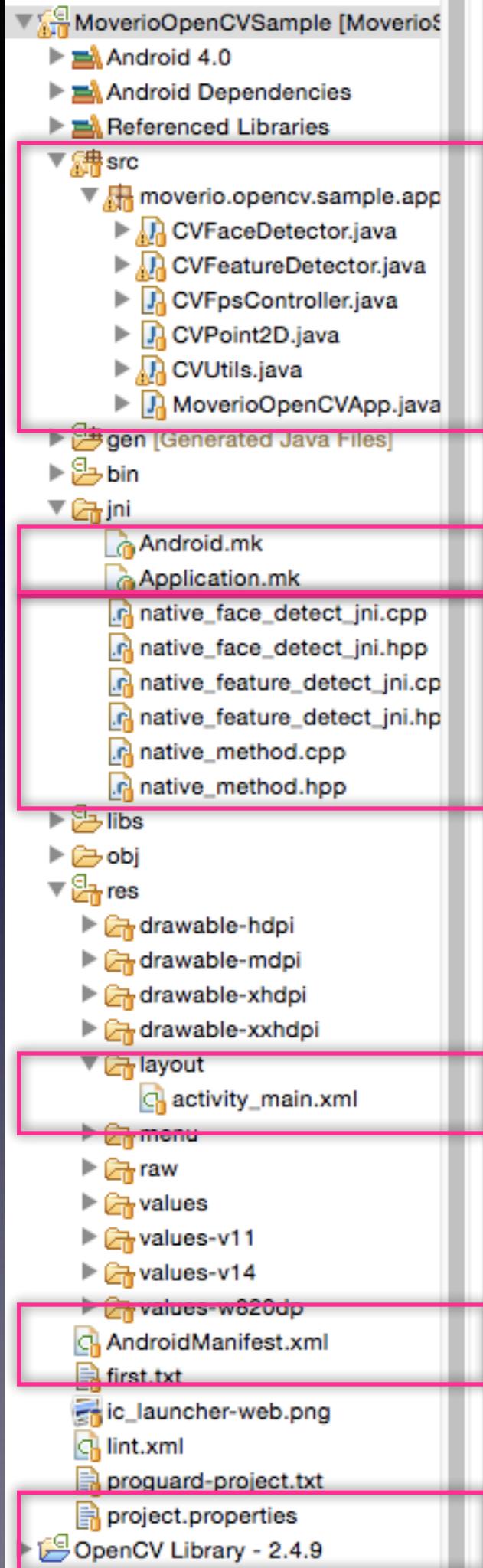
ndk-build正常終了時



```
MacBook-Air:MoverioSurfExample rm0426$ ndk-build
Android NDK: WARNING:jni/Android.mk:native_method: non-system libraries in linker flags: -lopencv_java
Android NDK:     This is likely to result in incorrect builds. Try using LOCAL_STATIC_LIBRARIES
Android NDK:     or LOCAL_SHARED_LIBRARIES instead to list the library dependencies of the
Android NDK:     current module
[armeabi-v7a] Compile++ thumb: native_method <= native_feature_detect_jni.cpp
jni/native_feature_detect_jni.cpp: In function 'void Java_moverio_opencv_sample_app_CVFeatureDetector_releas
eNativeFeatureDetect(JNIEnv*, jclass)':
jni/native_feature_detect_jni.cpp:59:22: error: 'hoge' was not declared in this scope
jni/native_feature_detect_jni.cpp:60:1: error: expected ';' before ')' token
make: *** [obj/local/armeabi-v7a/objs/native_method/native_feature_detect_jni.o] Error 1
MacBook-Air:MoverioSurfExample rm0426$
```

ndk-build異常終了時
(コンパイルエラー発生)

実装説明(概要)



MoverioOpenCVSampleの構成 (左図: EclipseのPackage Explorer)

1. Javaで記述されたソースファイル

2. スクリプトファイル(ビルド設定などを記載)

3. C++で記述されたNativeファイル

4. 画面構成が記載されたXMLファイル

5. アプリの全体構成などを記載するXMLファイル

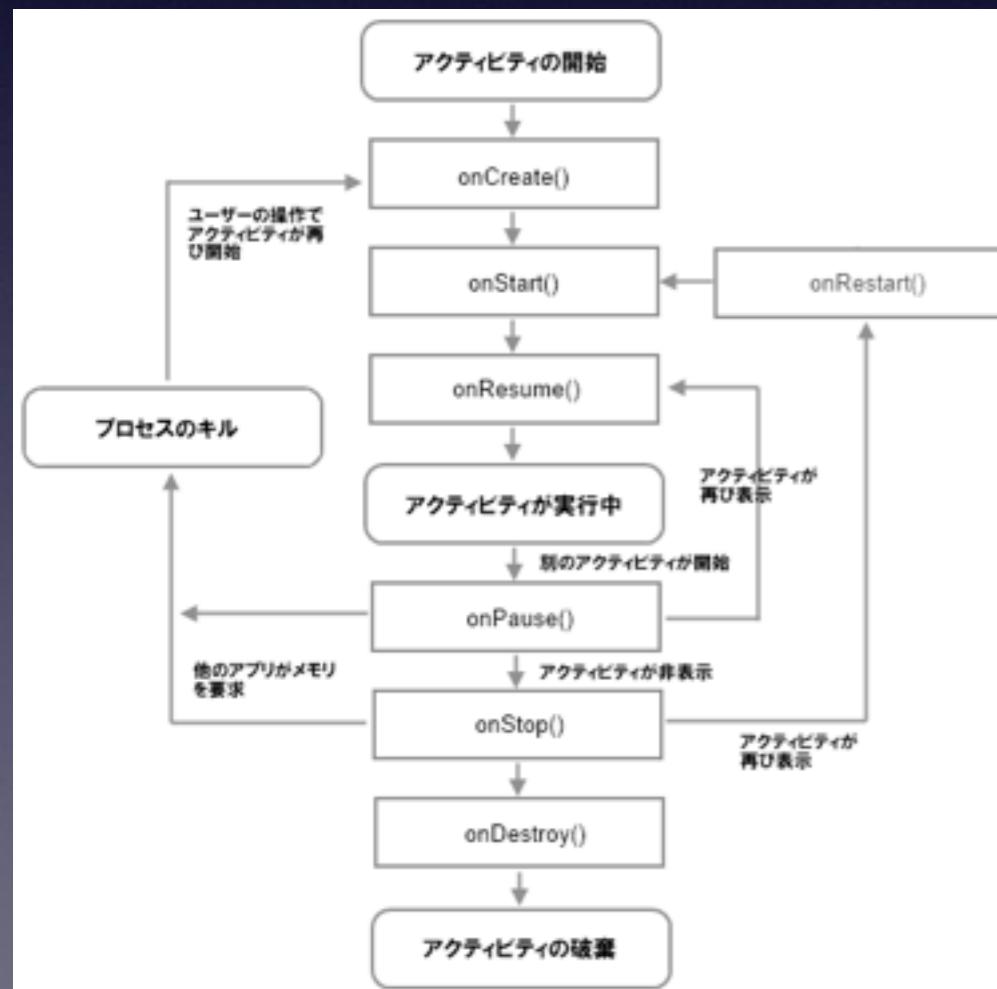
6. アプリの依存関係を記載する詳細ファイル

1. Javaファイル

- **MoverioOpenCVApp.java** : このアプリの親玉 (詳細は後述)
 - Activityと呼ばれるアプリのライフサイクルの管理
 - OpenCV使用時にカメラスレッドからのコールバックなどを受ける
 - 各モジュール、クラス変数を作成して、必要な処理を適切なタイミングで実行する
- **CVFaceDetector.java** : 顔検出用クラス
 - 顔検出に必要なcascadeファイルの読み込みや顔枠描画に必要な前処理をコンストラクタで処理
 - native_face_detect_jni.hpp/cppで記載されたnative実装を呼び出して実行する (3.で後述)
 - 処理の中身などは、OpenCV SDKのsamples/face-detectから移植
- **CVFeatureDetector.java**: 特徴点検出 + 描画用クラス
 - 特徴点抽出に必要な前処理をコンストラクタで処理 (詳細後述)
 - 特徴点抽出処理をnative/javaで切替をする機能の提供 (詳細後述)
 - native_feature_detect_jni.hpp/cppで記載されたnative実装を呼び出して実行する (3.で後述)
 - サポートしている特徴点抽出手法の取得、それらの処理に必要なしきい値を取得するAPIの提供(Stub)
- **CVFpsController.java**: 処理速度(FPS)の計算用クラス
- **CVPoint2D.java**: (x, y)座標を一度に扱うためのクラス (OpenCVまたはJavaで実装があるかもしれない…)
- **CVUtils.java**: 色々なメソッドや、クラスから使用される一般的な処理を実装、今回はリサイズなど

MoverioOpenCVApp.java

- ActivityとCVCameraViewListener2をimplementしている
- ライフサイクルは下記左図のAndroidで定義されているものと同様なので定石に従って実装
例) UI部品の初期化はonCreateで行う、 onPauseでメモリ解放 など
- CVCameraViewListener2で実装しなければいけない処理は下記右図参照
コードバックはUIスレッドではなくて、カメラスレッドから飛んでくるので、
とくに表示系に関わる処理はRunnableやHandlerを使って行う



PREV CLASS [NEXT CLASS](#) [FRAMES](#) [NO FRAMES](#)
[SUMMARY](#) [NESTED](#) [FIELD](#) [CONSTR](#) [METHOD](#) [DETAIL](#) [FIELD](#) [CONSTR](#) [METHOD](#)

org.opencv.android
Interface CameraBridgeViewBase.CvCameraViewListener2

Enclosing class:
[CameraBridgeViewBase](#)

public static interface CameraBridgeViewBase.CvCameraViewListener2

Method Summary

Modifier and Type	Method and Description
Mat	onCameraFrame (CameraBridgeViewBase.CvCameraViewFrame inputFrame) This method is invoked when delivery of the frame needs to be done.
void	onCameraViewStarted (int width, int height) This method is invoked when camera preview has started.
void	onCameraViewStopped () This method is invoked when camera preview has been stopped for some reason.

Method Detail

[onCameraFrame](#)

MoverioOpenCVApp.java

基本的にはコードにJavadocを記載しているので
そちらを参照してください。下記注意点となります

- onResume()内で下記実装していますが、Asyncと名前にもある通りOpenCVのライブラリのloadは非同期で行われるため、同期処理を直前直後に書いてしまうと記載順と実行順が異なることがあるので注意

カメラの初期化なども、OpenCVのloadが完了するのを待ってから行う

```
@Override
protected void onResume() {
    super.onResume();
    /* 非同期でライブラリの読み込み/初期化を行う
     * static boolean initAsync(
     *     String Version,
     *     Context ApplicationContext,
     *     LoaderCallbackInterface Callback) */
    OpenCVLoader.initAsync(OPENCV_LIB_VER, this, mLoaderCallback);
}
```

```
/**
 * ライブラリ初期化完了後に呼ばれるコールバック (onManagerConnected) <br>
 * public abstract class BaseLoaderCallback implements LoaderCallbackInterface
 */
private BaseLoaderCallback mLoaderCallback = new BaseLoaderCallback(this) {
    @Override
    public void onManagerConnected(int status) {
        Log.i(TAG, LOG_ON_MNG_STATUS+status);
        switch (status) {
            // 読み込みが成功したらカメラプレビューを開始
            case LoaderCallbackInterface.SUCCESS: {
                Log.i(TAG, LOG_OPNCV_SUCCESS);
                mCameraView.enableView();
            } break;
            default: {
                super.onManagerConnected(status);
                Log.i(TAG, LOG_ON_MNG_DEFAULT);
            } break;
        }
    }
};
```

initAsyncの引数mLoaderCallbackの
コールバック

MoverioOpenCVApp.java

- ・カメラの初期化が終わり、カメラからの入力が開始された通知(下記)で各クラスの初期化(initNativeMethods)を実行している

```
@Override
public void onCameraViewStarted(int width, int height) {
    Log.i(TAG, getClass().getSimpleName() +": (w,h)=( "+width+", "+height+")");
    mCameraWidth = width;
    mCameraHeight = height;

    initNativeJNImethods();
    initAdjustButtons();
    setVisibilityProcessBtns(View.VISIBLE);
    setVisibilityAdjustBtns(View.VISIBLE);
    setVisibilityNativeBtn(View.VISIBLE);
    setVisibilityBlackBackBtn(View.VISIBLE);

    mInitTextView.setVisibility(View.GONE);

    /**
     * Mat(int rows, int cols, int type)
     * rows(行): height, cols(列): width
     */
    mGrayImg = new Mat(height, width, CvType.CV_8UC1);
    //onCameraFrameの引数で渡される引数がα有りなので4ch
    mRgbaImg = new Mat(height, width, CvType.CV_8UC4);
    mUtils = new CVUtils();
    mTrans2D = new CVPoint2D(mTrans_x, mTrans_y);
}
```

MoverioOpenCVApp.java

基本的にはコードにJavadocを記載しているので
そちらを参照してください。下記注意点となります

- ・カメラからの画の入力を引数として、下記のコールバックで受ける
この中で現在設定されている各処理を実行している

```
@Override
public Mat onCameraFrame(CvCameraViewFrame inputFrame) {
    //TODO 高速化、リサイズの順番とか頑張るところ

    //fps計算
    mFpsController.count();
    String fps = mFpsController.getFrameRateText();

    //fps表示更新
    mHandler.post(new fpsShowRunnable(fps)); ←

    if(CV_CANNY.equals(mProcessName)) {
        //TODO 決め打ちになっているので、可能であればスライダーで変更など
        Imgproc.Canny(inputFrame.gray(), mGrayImg, 80, 100);
        //Core.bitwise_not(mOutputFrame, mOutputFrame);
        mUtils.resizeCameraInputFrameForMoverioSeeThrough(
            mGrayImg, mGrayImg, RESIZE_RATE, mTrans2D);
        return mGrayImg;
    } else if(CV_FEATURE_DETECT.equals(mProcessName)) {
        if(mIsBlackBack) {
            mRgbaImg = Mat.zeros(inputFrame.rgba().rows(),
                inputFrame.rgba().cols(), inputFrame.rgba().type());
        } else {
            inputFrame.rgba().copyTo(mRgbaImg);
        }
        mFeatureDetector.execFeatureDetect(←
```

カメラスレッドからこのメソッドは実行
されているので、表示はhandlerを使って
UIスレッドから処理している

各クラスの変数に画像情報を引数で
渡して画像処理を実行している
(Feature/Face)

CVFaceDetector.java

- C/C++ で記載されたメソッドを実行するにあたっては、関数の宣言のみをクラス内で行いそれを実行するような記載方法をとる
- 引数などは、実行したい処理に応じて決めればよいがjava層からC/C++層への値の変換などには注意が必要
javaからC/C++に画像を渡す際には、Mat型変数に格納した画像のnativeAddressを取得してそれを渡す
(native_face_detect_jni.hpp/cpp, android/application.mkの後述記載も参照)

```
private static native long nativeCreateObject(String cascadeName, int minFaceSize);
private static native void nativeDestroyObject(long thiz);
private static native void nativeStart(long thiz);
private static native void nativeStop(long thiz);          宣言のみ
private static native void nativeSetFaceSize(long thiz, int size);
private static native void nativeDetect(long thiz, long inputImage, long faces);
```

```
public CVFaceDetector(Context ctxt, int camera_width, int camera_height) {
    try {
        // load cascade file from application resources
        InputStream is = ctxt.getResources().openRawResource(R.raw.lbpcascade_frontalface);

        Log.e(TAG, LOG.Cascade_Load_Fail);
        mJavaDetector = null;
    } else {
        Log.i(TAG, LOG.Cascade_Load_From + path);
    }
    mNativeObj = nativeCreateObject(path, INIT_FACE_SIZE);
    cascadeDir.delete();

} catch (IOException e) {
    e.printStackTrace();
}
```

```
private void setMinFaceSize(int size) {
    nativeSetFaceSize(mNativeObj, size);
}

private void detect(Mat imageGray, MatOfRect faces) {
    nativeDetect(mNativeObj, imageGray.getNativeObjAddr(), faces.getNativeObjAddr());
}
```

Nativeメソッドへ画像のNativeアドレスを渡す

CVFeatureDetector.java

- ・コンストラクタで特徴点検出のために必要なクラス変数の生成、初期化を行う (Java, Nativeとともに)
- ・MoverioOpenCVApp#onCamreaFrameから実行されたexecFeatureDetect内で、引数isNative, isBlackBackの値を見て、Java/Nativeで実行するのか、背景をカメラ画にするのか黒画にするのかを分岐している

```
/*
 * 引数なしコンストラクタ <br>
 * @param feature 初期設定時に実行したい特徴点抽出手法
 * @param threshold 初期設定時に設定したい特徴量抽出に使用するしきい値 (今回は不使用)
 */
public CVFeatureDetector(int camera_width, int camera_height, int feature, int threshold) {
    Log.i(TAG, getClass().getSimpleName());

    mCameraWidth = camera_width;
    mCameraHeight = camera_height;
    mBlackImg = Mat.zeros(camera_height, camera_width, CvType.CV_8UC3);

    //Javaで実行するようの初期化
    mSupportedFeatureDetectMethods = new ArrayList<Integer>();
    mSupportedFeatureDetectMethods.add(FEATURE_ORB);

    mKeyPoint = new MatOfKeyPoint();

    changeFeatureDetectMethod(feature);
    changeFeatureDetectThreshold(threshold);

    //nativeで実行するようの初期化
    initNativeFeatureDetect(mCameraWidth, mCameraHeight, mFeature, mThreshold);
}
```

```
public void changeFeatureDetectMethod(int feature) {
    mFeature = feature;
    mFeatureDetector = FeatureDetector.create(mFeature);
    //mFeatureExtractor = DescriptorExtractor.create(mFeature);
}
```

```
/*
 * @param srcGrayImg 特徴点を抽出するための入力グレー画像
 * @param dstColorImg 抽出した特徴点を描画するためのカラー画像
 * @param isNative native(C++)で実行するか、javaで実行するか
 */
public void execFeatureDetect(Mat srcGrayImg, Mat dstColorImg, boolean isNative, boolean isBlackBack) {
    if(srcGrayImg == null || dstColorImg == null) {
        Log.e(TAG, LOG_IMG_NULL);
        return;
    }
    if(isNative) {
        //native method実行
        //Log.i(TAG, LOG_EXEC_DETECT+LOG_NATIVE);
        execNativeFeatureDetect(srcGrayImg.getNativeObjAddr(), dstColorImg.getNativeObjAddr(), isBlackBack);
    } else {
        //Log.i(TAG, LOG_EXEC_DETECT+LOG_JAVA);
        if(mDescriptors == null) {
            mDescriptors = new Mat(dstColorImg.rows(), dstColorImg.cols(), dstColorImg.type());
        }
        mFeatureDetector.detect(srcGrayImg, mKeyPoint);
        //mFeatureExtractor.compute(srcGrayImg, mKeyPoint, mDescriptors);
        if(isBlackBack) {
            Features2d.drawKeypoints(mBlackImg, mKeyPoint, dstColorImg);
        } else {
            Features2d.drawKeypoints(srcGrayImg, mKeyPoint, dstColorImg);
        }
    }
}
```

java側の初期化処理

2. スクリプトファイル

- **Android.mk** : 作成したcppファイル、それらのmodule名などを記載
AndroidやOpenCV SDK内のsampleと同様に書けばよさそう、あまり理解できていない
詳細は調べてください。。(´・ω・｀)
- **Application.mk** : サポートするAndroidのAPI versionなどを記載
同上

OpenCV.mkの場所指定 :

自分のプロジェクト以外のプロジェクト(ライブラリ)を
静的リンクするための記載. 詳細はよくわからず。。おまじない的に記載している

```
14#
15LOCAL_PATH := $(call my-dir)
16
17include $(CLEAR_VARS)
18include $(HOME)/opencv_android_lib/opencv-2.4.9-android-sdk/sdk/native/jni/OpenCV.mk
19
20LOCAL_MODULE      := native_method
21
22#複数ファイルの時は、とか使わなくてスペースで羅列
23LOCAL_SRC_FILES := native_method.cpp native_face_detect_jni.cpp native_feature_detect_jni.cpp
24
25LOCAL_LDLIBS      += -llog -ldl
26
27#APP_STL := stlport_static
28
29include $(BUILD_SHARED_LIBRARY)
```

Javaから呼び出す際のモジュール名

ソース・ファイルの指定

3. Native(C/C++)ファイル

- **native_face_detect_jni.hpp/cpp**: C++で実装されたOpenCVの顔検出メソッドを実行する
 - OpenCV/samples/face-detectionから移植、必要な箇所を少し修正
 - 背景を黒画にするか、カメラからの入力画にするかの分岐条件を実装
- **native_feature_detect_jni.hpp/cpp**: C++で実装されたOpenCVの特徴点抽出メソッドを実行する
 - <http://kesin.hatenablog.com/entry/20120810/1344582180>
 - http://docs.opencv.org/doc/tutorials/features2d/feature_detection/feature_detection.htmlなどを参考に実装
 - 複数枚から特徴点を抽出して、対応点を求めるコード例も記載があるが本アプリではカメラからの入力画1フレームに対して、ORB特徴点を求めて描画している
 - 背景を黒画にするか、カメラからの入力画にするかの分岐条件を実装
- **native_method.hpp/cpp**: 文字列をnative側から表示するためのテストファイル(アプリ内では未使用)
 - 本アプリでは使用していないが、特定の文字列をnativeメソッドから出力するテストコード

native_feature_detect_jni.hpp

- _hppファイルは特に作成する必要はないが、宣言をわかりやすくするために作成した
- 厳密にはNDK(JNI)はCのコードしか理解できないので、C++でかかれたコードはexternする必要がある
この辺りは、先にcppを作成してからjavahを用いてコンソールからhppを生成すると、自動的に記載してくれる
<http://d.hatena.ne.jp/shouh/20140329/1396051712>
<http://www.ne.jp/asahi/hishidama/home/tech/java/jni.html> など参照

```
5 #ifndef _Included_moverio_opencv_sample_app_CVFeatureDetector
6 #define _Included_moverio_opencv_sample_app_CVFeatureDetector
7 #ifdef __cplusplus
8 extern "C" {
9 #endif
10
11 /**
12 * 初期化native_feature_detect_jni.hpp/cpp
13 */
14 JNIEXPORT void JNICALL
15 Java_moverio_opencv_sample_app_CVFeatureDetector_initNativeFeatureDetect
16 (JNIEnv * jenv, jclass clazz, jint width, jint height, jint method, jint threshold);
17
18 /**
19 * 終了時のメモリ解放など
20 */
21 JNIEXPORT void JNICALL
22 Java_moverio_opencv_sample_app_CVFeatureDetector_releaseNativeFeatureDetect
23 (JNIEnv * jenv, jclass clazz);
24
25 /**
26 * 抽出
27 */
28 JNIEXPORT void JNICALL Java_moverio_opencv_sample_app_CVFeatureDetector_execNativeFeatureDetect
29 (JNIEnv * jenv, jclass clazz, jlong srcGrayImgAdd, jlong dstColorImgAdd, jboolean isBlackBack);
30
31 #ifdef __cplusplus
32 }
33#endif
34#endif
```

native側のメソッドの命名規則は、
Java_(使用するクラスpackageを_でつなぎ)_hogefuncとする

native_feature_detect_jni.cpp

基本的にはコードにJavadocを記載しているので
そちらを参照してください。下記注意点となります

- ・.hpp側で宣言をしたメソッドの中身を実装
- ・FeatureDetector::createに実行したい特徴点検出手法を引数で渡して初期化

```
36 /**
37 * 初期化、特徴点抽出クラスに何の手法を使うかなどを指定
38 */
39 JNIEXPORT void JNICALL
40 Java_moverio_opencv_sample_app_CVFeatureDetector_initNativeFeatureDetect
41 (JNIEnv * jenv, jclass clazz, jint width, jint height, jint method, jint threshold){
42
43     string mode = mSupported_features[0];//今回は何が渡されてきてもORBのみの動作とするため
44     detector = FeatureDetector::create(mode);
45 //    extractor = DescriptorExtractor::create(mode);
46     blackImg = Mat::zeros(cv::Size(width,height), CV_8UC3);
47 }
```

特徴点の検出
(jlong型で受け取ったaddressからの変換
がえげつない)

native側で特徴点検出を行うための
クラスを指定した検出手法で初期化
(本アプリではORBのみサポート)

```
62 /**
63 * 特徴点抽出実行 <br>
64 * isBlackbackの値によって、背景画を黒で塗りつぶすか、グレー画像にするかを選択できる
65 */
66 JNIEXPORT void JNICALL
67 Java_moverio_opencv_sample_app_CVFeatureDetector_execNativeFeatureDetect
68 (JNIEnv * jenv, jclass clazz, jlong srcGrayImgAdd, jlong dstColorImgAdd, jboolean isBlackBack){
69
70     if(srcGrayImgAdd == 0x0 || dstColorImgAdd == 0x0) {
71         LOGD("dstColorImgAdd or srcImgAdd is null. Can't start");
72     } else {
73         LOGD("execNativeFeatureDetect");
74         keypoints.empty();
75         detector->detect((*((Mat*)srcGrayImgAdd)), keypoints);
76     //}
77     if(isBlackBack) {
78         drawKeypoints(blackImg, keypoints, (*((Mat*)dstColorImgAdd)),
79                         Scalar::all(-1), DrawMatchesFlags::DEFAULT );
80     } else {
81         drawKeypoints((*((Mat*)srcGrayImgAdd)), keypoints, (*((Mat*)dstColorImgAdd)),
82                         Scalar::all(-1), DrawMatchesFlags::DEFAULT );
83     }
84 }
```

検出した特徴点の描画
引数のisBlackBackの値によって背景画を
黒画かカメラ画を切替

native_feature_detect_jni.cpp

基本的にはコードにJavadocを記載しているので
そちらを参照してください。下記注意点となります

・ http://docs.opencv.org/doc/tutorials/features2d/feature_detection/feature_detection.html

にあるように、FeatureDetectorやOrbFeatureDetector, SurfFeatureDetectorを下記のように宣言、初期化をしようとすると、図のようなエラーがでてしまう。

“detectImpl(またはcreateImpl)はpure virtualメソッドだからを継承して実装しろ”とでるが、
実際にFeatureDetectorを継承したMyFeatureDetectorを実装したからといってこのエラーは消えない。
→解決策としては、 `cv::Ptr<cv::FeatureDetector> detector;` を使う

参考サイト：<https://www.ngxo.com/thread/17478424>

<https://groups.google.com/forum/#topic/android-group-japan/M85uSG3wXp4>

<http://stackoverflow.com/questions/17478424/android-opencv-feature-detectors-implement-detect> (こちらのアドバイスを採用)

http://opencv.jp/opencv-2svn/cpp/basic_structures.html (Ptrの説明)

```
39 JNIEXPORT void JNICALL
40 Java_moverio_opencv_sample_app_CVFeatureDetector_initNativeFeatureDetect
41 (JNIEnv * jenv, jclass clazz, jint width, jint height, jint method, jint threshold){
42
43
44     FeatureDetector fd(100);
45
46     //The type 'cv::FeatureDetector' must implement the inherited pure virtual method 'cv::FeatureDetector::detectImpl'
47
48     //extractor = DescriptorExtractor::create(mode);
```

宣言時エラー

```
29 cv::Ptr<cv::FeatureDetector> detector;
```

解決案

基本的にはコードにJavadocを記載しているので
そちらを参照してください。下記注意点となります

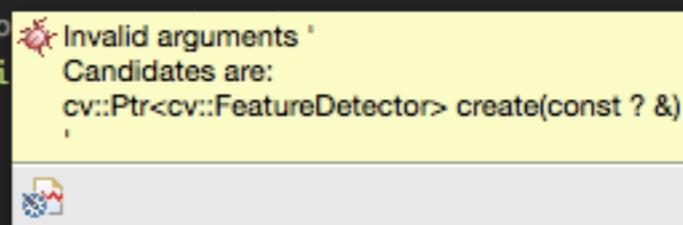
native_feature_detect_jni.cpp

- 前頁のようにFeatureDetectorを宣言して、::createに引数で検出手法を渡す際に、**Invalid argument**エラーが出ることがある(下図)

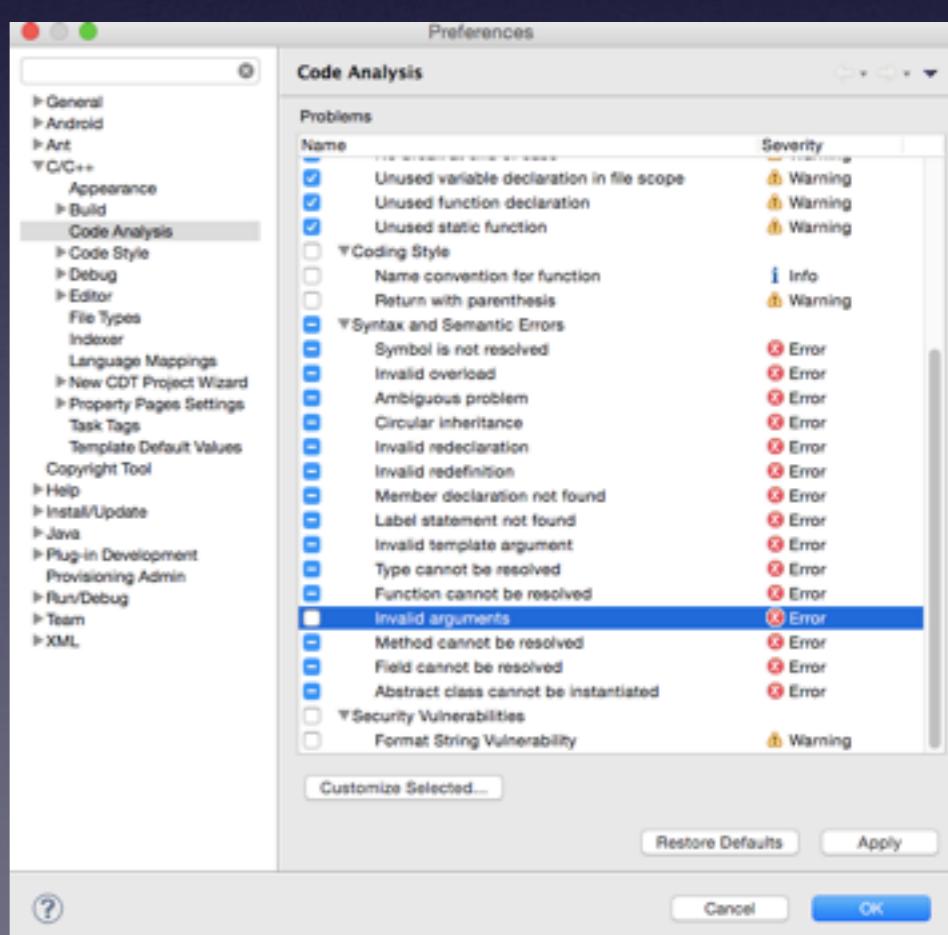
その時は、**Eclipseの環境設定->CodeAnalysys**から**Invalid argument**のチェックを外せばよい。
(こういうところでEclipseでのC/C++開発はまだ難あり感が出てくる・・・)

参考サイト：<http://n2works.net/frontend.php/column/pickup/id/105>

```
39 JNIEXPORT void JNICALL
40 Java_moverio_opencv_sample_app_CVFeatureDetector_initNativeFeatureDetect
41 (JNIEnv * jenv, jclass clazz, jint width, jint height, jint method, jint threshold){
42
43     string mode = mSupported_features[0];//今回は何が渡されてきてもORBのみの動作とするため
44     detector = FeatureDetector::create(mode);
45 //extractor = ecriptorExtractor;
46     blackImg = Mat::zeros(cv::Size(width, height), CV_8UC3);
47 }
48 /**
49 */
50
51
52 }
```



Invalid Argument エラー



参考サイトなど

Android+OpenCV開発で参考にしたサイト例

- OpenCV for Android
 - <http://rest-term.com/archives/3010/>
 - <http://rest-term.com/technote/index.php/OpenCV%20for%20Android>
- カメラプレビューのキャプチャ, AndroidManifest.xmlへの記載など
 - <http://techbooster.jpn.org/android/device/9632/>
- [Android] AndroidManifest.xmlの uses-feature タグについて
 - <http://blog.be-style.jpn.com/article/55916343.html>
- ARに使える画像処理をAndroidで実行(JNIも軽く触れて説明)
ORB特徴量を使って道路標識認識
 - http://www.atmarkit.co.jp/ait/articles/1203/01/news159_2.html

Android+OpenCV開発で参考にしたサイト例

- OpenCV 画サイズ変更
 - <http://blog.goo.ne.jp/nobotta50/e/eea4c096108ac91d2c457ba2402593d3>
- OpenCV 画切り抜き
 - <http://blog.goo.ne.jp/nobotta50/e/c8ed07241c191f9506d3e49b92c136be>
 - http://geekn-nerd.blogspot.jp/2012/02/opencv-for-android_18.html
- OpenCV CookBook
 - <http://opencv.jp/cookbook/index.html>
- cv::Matを初期化する方法
 - <http://physics-station.blogspot.jp/2013/03/cvmat.html>

NDK開発で参考にしたサイト例

- MacにNDKをインストールと設定する
 - <http://ohwhsmm7.blog28.fc2.com/blog-entry-392.html>
- Android NDKでHelloWorldしてみる
 - <http://opamp.hatenablog.jp/entry/20130115/1358266941>
- NDKサンプルプログラムのビルド
 - <http://technobrain.seesaa.net/article/246793612.html>
- C++/Javaから双方向でメソッドを呼び出す
 - <http://cflat-inc.hatenablog.com/entry/2014/07/14/074535>
- JNI実装時のC, C++の違い
 - <http://inujirushi123.blog.fc2.com/blog-entry-103.html>
- NDK、jniビルド時、実行時のエラー
 - <http://blog.epyOnOff.com/archives/category/android/ndk>
- Native methodないよ(Unsatisfied link error)発生
 - <http://kotatsu52.hatenablog.com/entry/2012/02/21/005029>
 - <http://blog.toridge.com/archives/618>
- ndk-build時に__android_log_printが参照画エラーを吐く
 - <http://stackoverflow.com/questions/4455941/undefined-reference-to-android-log-print>

その他

- java系のログがterminalで文字化け
 - <http://bit.ly/1z1ibpZ>
- Android.mk の書き方
 - <http://qiita.com/alingogo/items/0df5738d94af162356aa>
- OpenCV for Android 2.4.9のNativeActivity (Macでは実行不可)
 - <http://www.programering.com/a/MjN1cjNwATQ.html>
 - <http://code.opencv.org/issues/3542>
 - <http://stackoverflow.com/questions/17211933/fatal-signal-11-sigsev-at-0x00000000-code-1-thread-32140>
- malloc, memset, memcpyに型変換が違うぞと怒られるエラー
 - <http://maxportanami.blogspot.jp/2013/03/android-jni.html>
- CDT pluginがインストール/アップデートできない
 - <http://inujirushi123.blog.fc2.com/blog-entry-103.html>
- JNI_OnLoadってなんやねん
 - <http://serenegiant.com/blog/?p=1417>

その他

- Gitを使ってソースの管理をするのがおすすめ !!
- サルでもわかるGit
 - <http://www.backlog.jp/git-guide/>
- GitHub導入
 - shim0mura.hatenadiary.jp/entry/20111212/1323660740
(ちょっと古い)
- DropboxをGitのリポジトリにする
 - <http://m.designbits.jp/13061220/>

Fin