

NetVision
UVC SDK Manual
(Linux)
V2.23

2025/05/07

株式会社ネットビジョン

改訂履歴

版数	日付	内容	担当
V1.00	2016/06/20	新規作成	山田
V2.00	2017/07/10	V4I2Captureの更新	山田
V2.10	2018/03/09	Raw8対応	山田
V2.20	2021/12/14	“NVCapSimple-Linux”を追加	山田
V2.21	2025/02/04	ドキュメントを新規フォーマットに対応 一部文言の修正	天野
V2.22	2025/05/07	一部の図、文言を更新 推奨動作環境を更新 “SVMCtl Linux”を追加	天野
V2.23	2025/05/15	一部誤字の修正	天野

目次

1. 概要	4
1.1. プロジェクト構成	5
1.2. 推奨動作環境	5
2. フォルダ構成	6
2.1. センサ設定ファイル	7
2.1.1. 内容	7
2.2. 関連ドキュメント	8
3. Linux 版 SDK	8
3.1. 開発ツールのインストール	8
3.2. 使用するライブラリのインストール方法	8
3.2.1. Qt	8
3.2.2. OpenCV	9
3.2.3. libv4l	10
3.2.4. libSDL2	10
3.3. libsvm ライブラリ&ソースコード	10
3.3.1. 説明	11
3.3.2. インストール方法	11
3.3.3. 関数リファレンス	11
3.4. v4l2Capture プロジェクト	15
3.4.1. 説明	16
3.4.2. ビルド	17
3.4.3. 操作方法	18
3.4.4. コマンドラインオプション	18
3.4.5. Qt ライブラリとの併用	19
3.4.6. 出力動画ファイルフォーマット	20
3.4.7. 備考	21
3.5. v4l2Capture_split プロジェクト	21
3.5.1. 説明	22
3.5.2. 備考	22
3.6. v4l2SDLCapture プロジェクト	22
3.7. v4l2Player プロジェクト	23
3.7.1. 説明	23

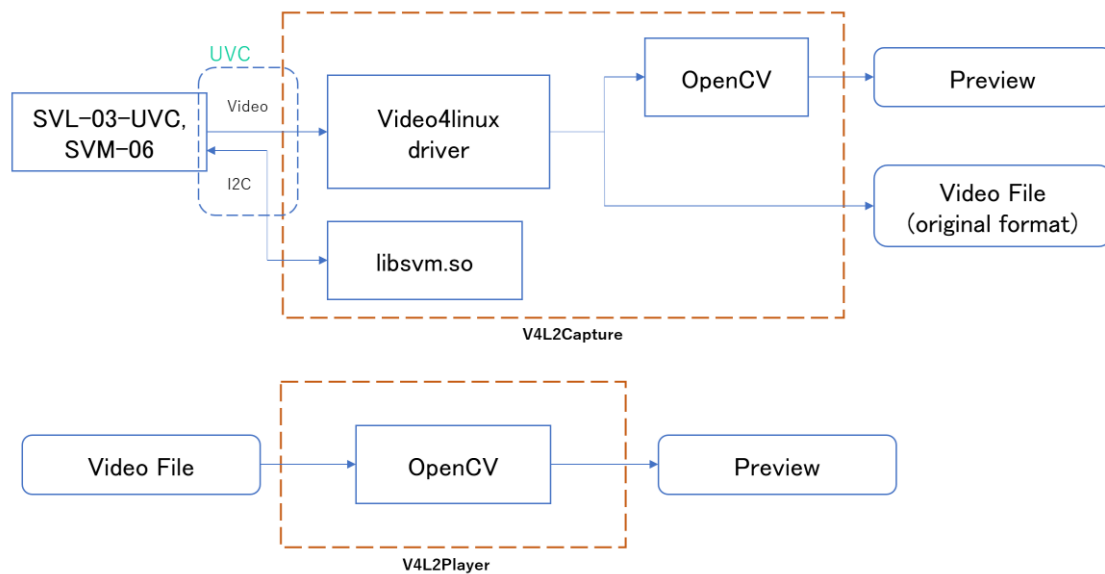
3.7.2.	ビルド.....	23
3.7.3.	コマンドラインオプション.....	23
3.8.	ArroundView プロジェクト.....	24
3.8.1.	説明.....	24
3.9.	SVMSEnder プロジェクト.....	25
3.9.1.	説明.....	26
3.9.2.	ビルド.....	27
3.9.3.	コマンドラインオプション.....	27
3.9.4.	備考.....	27
3.10.	SVMCtlLinux プロジェクト.....	28
3.10.1.	説明.....	29
3.10.2.	ビルド.....	29
3.10.3.	備考.....	29
I2C	通信フォーマット.....	30
1.1.	I2C Write	30
1.2.	I2C Read (Restart Condition チェックあり).....	30
1.3.	I2C Read (Restart Condition チェックなし).....	30

1. 概要

NetVision UVC SDK は、NetVision SVL ボードや SVM-06 ボード等、弊社 USB Video Class (UVC) 対応の USB3.0、USB3.2-Gen2 キャプチャボード(以下 SVM ボードと表記)を使用するためのソフトウェア環境一式を含む SDK です。

Linux 版 SDK は OpenCV ライブラリを用いたキャプチャソフト、再生ソフト、I2C 通信を行うためのライブラリ、ボード設定用のユーティリティソフトを含みます。

ライブラリやサンプルはすべて C++ 言語で書かれています。Linux 版 SDK は gcc およびオープンソースのライブラリを使用してビルドできるようになっています。



Linux 版 SDK を使用した SVM ボードのキャプチャ環境の構成図を上図に示します。UVC を使用したキャプチャは図中に橙色で示した「V4L2Capture」によって動作し、映像信号のプレビュー、キャプチャと I2C 通信が可能です。キャプチャされた映像ファイルは入力されたデータ構造が非圧縮で格納されており、「V4L2Player」によって再生することができます。このほか、I2C 通信や SVM ボードの設定を行うためのユーティリティも SDK に含まれています。

I2C 送受信を含むこれらの機能は UVC 準拠なので、OS 標準のドライバおよび Video For Linux (v4l2) Driver を使用して行われます。

NV_UVC_SDK(Linux)は弊社ホームページからダウンロード可能です。

https://www.net-vision.co.jp/support/app/NV_UVC_SDK_Linux.zip

SDK 付属のキャプチャソフト以外にも、OpenCV や ROS など、NVCap 以外のサードパーティー製ライブラリを使用して映像処理を行うこともできます。

1.1. プロジェクト構成

Linux 版 SDK は下記によって構成されています。

- v4l2Capture (キャプチャソフト)
- v4l2StereoCapture (v4l2Capture に 2ch 受信機能を追加したもの)
- v4l2Player (録画データの簡易プレーヤソフト)
- v4l2Capture_split
- v4l2SDLCapture
- SVMSender
- SimpleI2C
- SVMCtlLinux
- NVCapSimple-Linux (YUV, RAW10, RAW12 表示のサンプルソフト)
- ArroundView
- undisort_init
- sdk_lib (SVM ボード用ライブラリ)

それぞれのプロジェクトの詳細は 2 章以降に説明します。

1.2. 推奨動作環境

名前	値	備考
OS (Linux 版 SDK)	Ubuntu 22.04 LTS, 24.04 LTS	
CPU	Intel Core i5 以降	
Memory	3GB 以上	
主記憶装置	SSD 使用	録画機能を使用する場合
シリアルインタフェース	USB 3.0 ポート	マザーボード搭載のコントローラを推奨 PCI カードや USB ハブは相性問題あり
コンパイラ (Linux)	g++	

2. フォルダ構成

フォルダ名	説明
dat	センサ設定ファイルの例を格納しています。
Linux_SDK	Linux 用 SDK を格納しています。
+ doc	Linux 用ドキュメントを格納しています。
+ ArroundView	4 台の SVM ボードを用いてア라운드ビュー表示を行うデモです。2015 年の展示会で使用したソフトで、詳細説明はありませんが、応用例として SDK に付属します。
+ sdk_lib	SVM ボード用ライブラリです。Linux 版 SDK のサンプルプロジェクトから使用します。
+ SimpleI2C	I2C 送受信を行うためのシンプルなコマンドラインベースのプロジェクトです。
+ SVMCtlLinux	Windows 版 SVMCtl (SVM ボード用ユーティリティソフト) の UI をベースにした SVM ボードの設定用ユーティリティです。UI は qt を用いており、ボードとの通信は Linux 版 UVC SDK を使用しています。
+ SVMSender	I2C 送受信を行うためのコマンドラインベースのプロジェクトです。Windows 版 SVMCtl で送信できる .dat 形式の I2C 設定ファイルの送信に対応します。
+ undistort_init	カメラのひずみ補正を行うプログラムです。OpenCV のサンプルとほぼ同等です。「ArroundView」と組み合わせて使用します。
+ v4l2Capture	SVM ボード用 Linux 版キャプチャソフトです。最大 4 つまでの SVM ボードの同時プレビュー、録画に対応します。映像取り込みは video4linux、映像表示は OpenCV によって行います。
+ v4l2Capture_split	基本的に v4l2Capture と同じですが、取得された映像の右半分、左半部分を別のウィンドウで表示します。
+ v4l2Player	v4l2Capture で保存したファイルを再生するためのソフトです。

フォルダ名	説明
+ v4l2SDLCapture	基本的に v4l2Capture と同じですが、描画部分を SDL に変更しています。v4l2Capture より高速に描画できますが、v4l2Capture で行っている映像処理や情報表示は実装していません。また、フレームフォーマットは YUV 形式のみサポートします。
+ v4l2StereoCapture	基本的に v4l2Capture と同じですが、2 台の SVM-03 ボードを使用したステレオキャプチャ用のコードを実装しています。
+ NVCapSimple-Linux	YUV 表示と Raw10, Raw12 モノクロ表示を行うためのサンプルコードです。弊社ウェブページからダウンロードできるバージョンと同じものが格納されています。 詳細は「NVCapSimple_Linux_動作説明書」を参照してください。

2.1. センサ設定ファイル

SDK_Linux

```

├── v4l2StereoCapture/
├── v4l2Capture_split/
├── v4l2Player/
├── ...
└── dat/
```

2.1.1. 内容

SVMCtl (Windows)、NVCap (Windows) アプリケーションや SVMSEnder (Linux) はセンサ設定ファイルに書かれた I2C シーケンスをセンサに送信することができます。dat フォルダにはこのセンサ設定ファイルのサンプルをいくつか格納しています。

ファイル名	説明
imx219_720p.txt	Raspberry Pi Camera v2 を 1280x720 / 47.5 fps / Raw10 / MIPI 2 Lane で初期化するための設定ファイルです。
imx219_1080p.txt	Raspberry Pi Camera v2 を 1920x1080 / 47.5 fps / Raw10 / MIPI 2 Lane で初期化するための設定ファイルです。
ov5647_1080p.txt	Raspberry Pi Camera を 1920x1080 / 30 fps / Raw10 / MIPI 2 Lane で初期化するための設定ファイルです。
ov5642_720p_30fps_vs_neg_uyvy.txt	OV5642 (Omnivision) を 1280x720 / 30 fps / UYVY / 8bit で初期化するための設定ファイルです。
ov5642_vga_30fps_vs_neg_uyvy.txt	OV5642 (Omnivision) を 640x480 / 30fps / UYVY / 8bit で初期化するための設定ファイルです。

2.2. 関連ドキュメント

SDK_Linux

```

├── v4l2StereoCapture/
├── v4l2Capture_split/
├── v4l2Player/
├── ...
└── doc/

```

ファイル名	説明
NV_UVC_SDK_Manual (Linux).pdf	本書です。
Extension_Unit_Specification_v2.0_Release.pdf	SVM ボードの Extension Unit の仕様書です (I2C 関連部分のみ抜粋)。

3. Linux 版 SDK

3.1. 開発ツールのインストール

Linux 版 SDK のライブラリやサンプルプロジェクトをビルドするための開発ツールをインストールする必要があります。

下記コマンドにより開発ツールをインストールすることができます。

```
$ sudo apt install g++ cmake
```

3.2. 使用するライブラリのインストール方法

Linux 版 SDK は以下の外部ライブラリを使用しているので、あらかじめインストールする必要があります。

OpenCV (動作確認済: 4.11.0)

Qt (動作確認済: 5.15.13 もしくは 6.4.2)

libv4l (動作確認済: 1.26.1)

libsd12 (動作確認済: 2.30.0)

本 SDK は、上述のバージョンのライブラリをインストールされた環境で正常に動作することを確認しています。ライブラリ間の依存関係の関係から、新規環境にインストールする際は、以下の説明の順序に従うことを推奨します。なお、上記は本書執筆時点でのバージョンであるため、インストール時の最新バージョンと異なる場合や、インストール方法が異なる可能性があることをご了承ください。

3.2.1. Qt

Qt は複数プラットフォームで動作する C++ のユーザインタフェースライブラリです。Qt のインストールはオプションであり、必須ではありません。Qt を使用するオプションで OpenCV をビルドすることで、UVC SDK のプロジェクトに追加のユーザインタフェースを導入することができます。また、SDK 付属の SVMctlLinux プロジェクトは Qt Creator のプロジェクトの形で

提供されており、ビルドには Qt のインストールが必要です。

下記コマンドにより Qt の開発ツールをインストールすることができます。

- Qt5 の場合

```
$ sudo apt install qtbase5-dev qt5-qmake
```

- Qt6 の場合

```
$ sudo apt install qt6-base-dev qmake6
```

Qt5 と Qt6 の両方をインストールすることができますが、その場合、Qt を使用する OpenCV のビルドでは Qt6 が使用されません。

3.2.2. OpenCV

OpenCV は、画像処理を扱うためのマルチプラットフォームライブラリです。Linux 版 SDK ではキャプチャソフトの表示部分に OpenCV ライブラリを使用します。

ここでは、OpenCV をソースコードからビルドします。もし Qt ライブラリを使用する場合、Qt ライブラリを OpenCV のビルドより前にインストールしておく必要があります。OpenCV 4.11.0 のインストール方法を以下に示します。

1. ソースコードをダウンロードして展開します。

```
wget https://github.com/opencv/opencv/archive/4.11.0.zip
```

```
unzip 4.11.0.zip
```

2. ビルドディレクトリを作成して移動します。

```
mkdir -p build && cd build
```

3. Makefile を生成します。

- a) Qt を使用したい場合

```
cmake -D WITH_QT=ON -D WITH_OPENMP=ON ../opencv-4.11.0
```

- b) Qt を使用したくない場合

```
cmake -D WITH_OPENMP=ON ../opencv-4.11.0
```

4. ビルドします。

```
make -j4
```

5. インストールします。

```
sudo make install
```

3.2.3. libv4l

libv4l は Video4Linux ドライバ (v4l2) のためのユーティリティライブラリであり、v4l2 を使用するためのソースコードとヘッダファイルを含んでいます。SDK では v4l2Capture や libsvm ライブラリがビデオデータを直接キャプチャするために v4l2 を使用しているので、これらをビルドするために開発用の libv4l-dev をあらかじめインストールする必要があります。

```
$ sudo apt install libv4l-dev
```

3.2.4. libsdl2

サンプル v4l2SDLCapture は描画に SDL (Simple DirectMedia Layer) を使用しているため、v4l2SDLCapture をビルドするためには開発用の libsdl2-dev をあらかじめインストールする必要があります。

```
$ sudo apt install libsdl2-dev
```

3.3. libsvm ライブラリ&ソースコード

SDK_Linux

```
├── v4l2StereoCapture/
├── v4l2Capture_split/
├── v4l2Player/
├── ...
└── sdk_lib/
```

ファイル名	説明
Makefile	Makefile です。
UVCManager.cpp UVCManager.h	CExtensionUnitManager クラス / CUVCManager クラスのインプリメントです。
V4LCapture.cpp V4LCapture.h	CV4LCapture クラスのインプリメントです。

3.3.1. 説明

libsvm ライブラリは、SVM ボードの機能を Linux 環境で使用するための共有ライブラリです。このライブラリは、I2C 送受信などベンダ固有の Extension Unit 機能を使用するための CExtensionUnitManager クラス / CUVCManager クラスと、V4L (Video For Linux) ドライバを使用して OpenCV に依存しない高速画像取り込みを行う CV4LCapture クラスによって構成されています。

本ライブラリは他の SDK サンプルプロジェクト(v4l2Capture、SVMSender)でも使用しています。ライブラリの具体的な使用方法については、これらプロジェクトのソースコードを参照してください。

3.3.2. インストール方法

1. `make` により共有ライブラリとしてビルド
* libsvm.so.1.0.0 が生成されます。
2. `$ sudo make install` によりライブラリとヘッダをインストール
* デフォルトのインストール先は、ライブラリ `/usr/local/lib`、ヘッダ `/usr/local/include` となっています。

3.3.3. 関数リファレンス

* 以下は主要な関数のみ記述しています。

– CUVCManager クラス(一般的な Extension Unit を扱うためのクラス)

ヘッダファイル: “UVCManager.h”

関数	説明
CUVCManager	コンストラクタです。 引数 deviceID を指定した場合、開かれるビデオデバイスの ID を指定します。
~CUVCManager	デストラクタです。デバイスが既に開かれている場合、デストラクタ内でリソースは解放されます。
OpenDevice	ビデオデバイスを開きます。引数 deviceID によって開かれるデバイスの ID を指定します。 ここで指定するデバイス ID には、OS に対して <code>/dev/videoN/</code> で認識される番号を指定します。
CloseDevice	もしデバイスが開かれている場合、デバイスを閉じてリソースを開放します。
XUGetLength	指定された Unit ID およびコントロール番号の Extension Unit の規定データ長を返します。
XUGetValue	指定された Unit ID およびコントロール番号の Extension Unit からデータを受信します。
XUSetValue	指定された Unit ID およびコントロール番号の Extension Unit へデータを送信します。

– CExtensionUnitManager クラス

ヘッダファイル: “UVCManager.h”

SVM ボード の Extension Unit の仕様を定義したクラスです。CUVCManager クラスを継承しています。現在のバージョンでは、I2C データの送受信のみをインプリメントしています。

関数	説明
CUVCManager	コンストラクタです。 引数 deviceID によって開かれるビデオデバイスの ID を指定します。
~CUVCManager	デストラクタです。
SendI2CData	I2C データを送信します。 <u>引数</u> – int addr 7bit の I2C デバイスアドレスを指定します。 – const unsigned char* data 送信するデータの配列を指定します。 1 回に送信できるデータ長は最大 30 バイトとなります。 – int length 引数 data のデータ長 [byte] を指定します。 <u>返り値</u> 0: 成功 それ以外: 失敗 <u>備考</u> SVM ボードの FW バージョン 71 以前では、ターゲットデバイスから NACK を受信しても本関数は成功を返します。
ReceiveI2CData	I2C データを受信します。 <u>引数</u> – int addr 7bit の I2C デバイスアドレスを指定します。 – unsigned char* data 受信されたデータが格納される配列をし指定します。 1 回に受信できるデータ長は最大 30 バイトとなります。 – int length 引数 data のデータ長 [byte] を指定します。 – const unsigned char* preamble 受信レジスタアドレスなど、受信コマンドの前に送信するデータ (preamble) を指定します。 – int preambleSize

	<p>preamble のデータ長 [byte] を指定します。</p> <p>– int receiveWait = I2C_WAIT_1</p> <p>コマンド送出から受信データをボードから読み込むまでの待ち時間を指定します。</p> <p><u>返り値</u></p> <p>0: 成功</p> <p>それ以外: 失敗</p> <p><u>備考</u></p> <p>引数 preamble は後述する「I2C パケットフォーマット」図中の Register Address に相当します。</p> <p>SVM ボードの FW バージョン 71 以前では、ターゲットデバイスから NACK を受信しても本関数は成功を返します。</p>
GetFPGARegister	<p>SVM ボードの FPGA のレジスタ値を取得します。</p> <p>FPGA レジスタマップは通常公開していませんが、カスタム対応等で一部のみにユーザに開示することがあります。</p> <p><u>引数</u></p> <p>– unsigned int addr</p> <p>32bit の FPGA レジスタアドレスを設定します。</p> <p>– unsigned int* value</p> <p>レジスタ値を格納する変数へのポインタを指定します。</p> <p><u>返り値</u></p> <p>0: 成功</p> <p>それ以外: 失敗</p>
SetFPGARegister	<p>SVM ボードの FPGA のレジスタ値を設定します。</p> <p><u>引数</u></p> <p>– unsigned int addr</p> <p>32bit の FPGA レジスタアドレスを設定します。</p> <p>– unsigned int value</p> <p>新たに設定するレジスタ値を指定します。</p> <p><u>返り値</u></p> <p>0: 成功</p> <p>それ以外: 失敗</p>

–CV4LCapture クラス

ヘッダファイル: “V4LCapture.h”

v4l2 ドライバによりビデオデバイスから映像キャプチャを行うために使用するクラスです。

関数	説明
CV4LCapture	コンストラクタです。
~CV4LCapture	デストラクタです。
Init	<p>ビデオデバイスを開きます。引数 id によって開かれるビデオデバイスの ID を指定します。</p> <p>ここで指定するデバイス ID には、OS に対して <code>/dev/videoN/</code> で認識される番号を指定します。</p> <p>また、ビデオデバイスがマルチ解像度に対応している場合、引数 width / height 指定した解像度に最も近い解像度で初期化されます。SVM-03U では width/ height は無視されます。</p> <p>デバイスのタイムアウト時間は timeout、バッファするフレーム数は queueSize 引数によって指定できます。</p>
CaptureFrame	ビデオデバイスから 1 フレームのキャプチャを行います。この関数でキャプチャを行うと、色変換の行われない生の映像データが取得できます。ビデオドライバの FIFO バッファが空の場合、次フレームがキャプチャされるまで関数は返りません。(タイムアウト処理有り)
GetVideoFormat	<p>開かれているデバイスの現在のビデオフォーマットを取得します。ビデオフォーマットは下記に定義する</p> <p>CV4LCapture::sVideoFormat 構造体を指定します。</p>
GetCameraID	開かれているデバイスの ID (<code>/dev/videoN</code> の N にあたる数字)を返します。

3.4. v4l2Capture プロジェクト

SDK_Linux

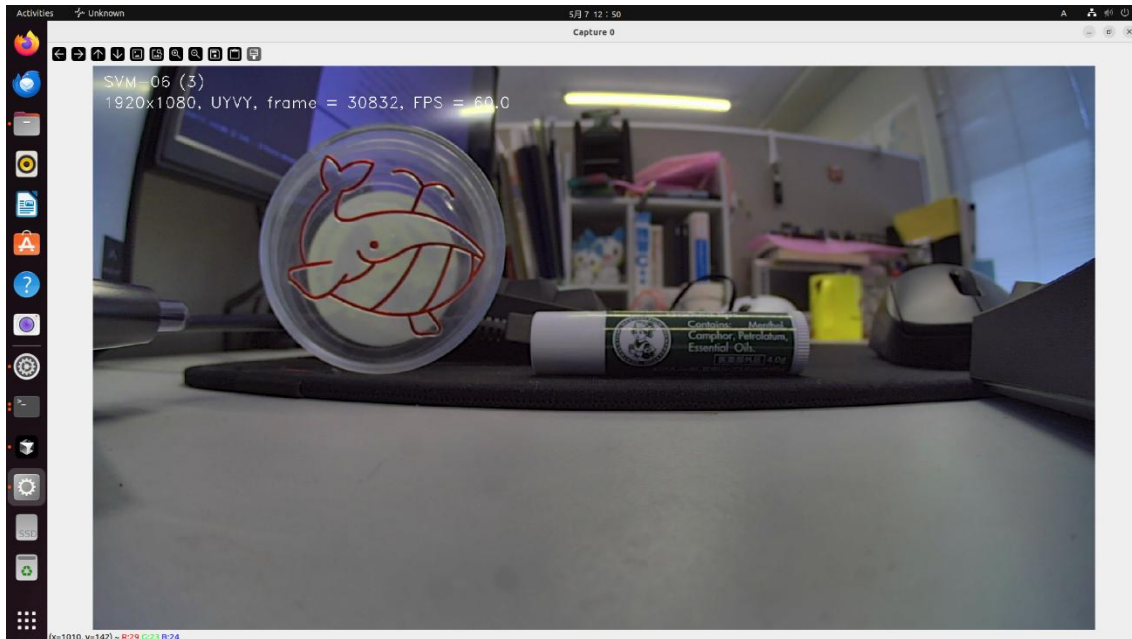
```

|—— v4l2StereoCapture/
|—— v4l2Capture_split/
|—— v4l2Player/
|—— ...
|—— v4l2Capture/

```

ファイル名	説明
CMakeLists.txt	cmake ツール用設定ファイルです。
CommandParser.h	コマンドライン解析用クラスです。
FaceDetector.cpp	顔認識クラスです。
FrameConverter.h	YUV/RGB 変換を行うためのクラスです。
main.cpp main.h	main 関数を含むプログラムの主要部分のソースコードです。
TimeConsManager.h	消費時間計測用クラスです。 (デバッグ・開発用)
UserFilter.cpp UserFilter.h	ユーザ定義フィルタを記述するファイルです。 UserFilter で定義された関数のため、OpenCV を使用した画像処理を記述するためのテンプレートになっています。 ユーザが画像処理を挿入する場合、このファイルにプログラムを記述すると便利です。
Capture.cpp Capture.h	main 関数から呼び出される関数の一部をクラス化したものです。

3.4.1. 説明



マルチチャンネルカメラキャプチャ用ソフトウェア「v4l2Capture」は、SVM ボードで取り込まれた映像を表示、保存する機能を持つ、SVM ボード動作サンプル用 GUI ソフトウェアです。最大 4 チャンネルまでのデバイスを同時に動作させることができます。本ソフトウェアは C++ 言語ソースコードの形で提供されます。

主な処理の流れは、main.cpp の main() 関数を参照してください。

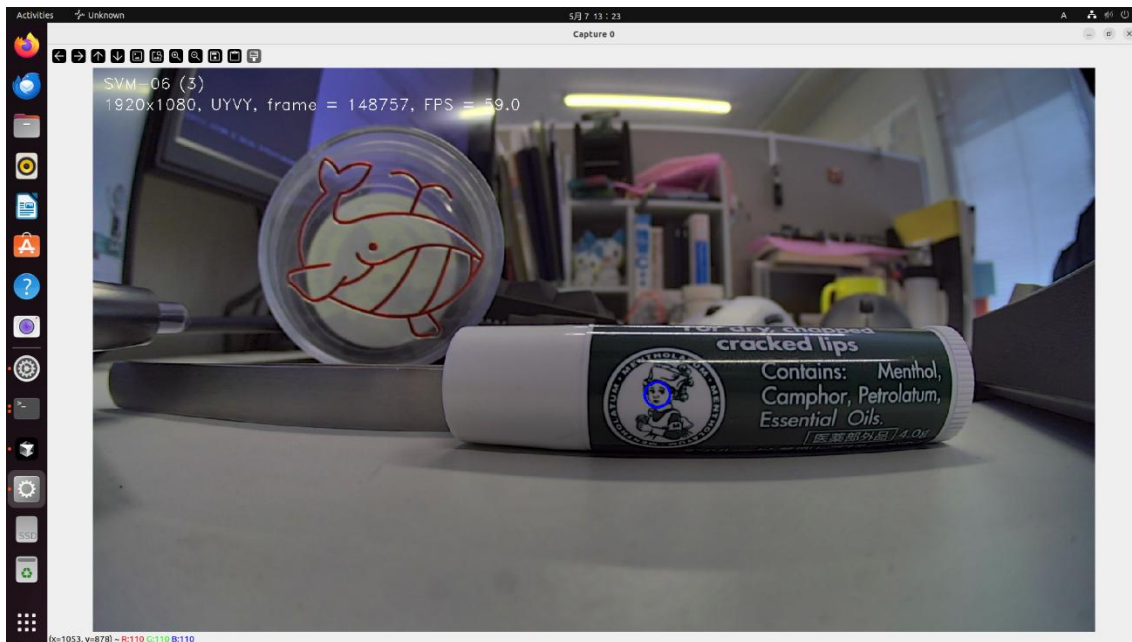
main() 関数のメインループ (MainLoop() 関数) では、

1. デバイスからバッファに画像取り込み (生データ)
2. ファイル出力
3. YUV → RGB 変換
4. ユーザ定義のフィルタ処理
5. 情報のオーバーレイ表示
6. 画面表示
7. FPS 計算
8. キー入力検出

という流れになっています。

画面表示は OpenCV ライブラリによって行っています。デバイスからの映像取り込みは OpenCV 等ライブラリの機能を用いても可能ですが、このソフトウェアでは生のデータを処理する応用を考えて、組み込みの YUV→RGB の変換をスルーするために、libsvm ライブラリを使用して Linux UVC ドライバ (V4L) を直接呼び出しています。このため、v4l2Capture のビルドには libsvm ライブラリのインストールが必要です。

YUV→RGB の変換は、2 段階に分けて行います。422 → 444 変換はソフトウェアで、YUV→RGB 変換は OpenCV のライブラリ関数で行います。これらの処理は main.cpp の ConvertFrame 関数内で行われています。



また、OpenCV を使った応用を示すために、v4l2Capture では Haar 特徴量ベースの Cascade 識別器による顔認識を行うサンプルを組み込んでいます (ON/OFF 切り替え可能)。上図は顔認識の結果例で、認識結果が青い丸で示されています。(もちろん、イラスト以外からの認識も可能です。) アルゴリズムの詳細は以下を参照してください。

https://docs.opencv.org/4.x/db/d28/tutorial_cascade_classifier.html

(補足) v4l2 の UVC モジュールはキューを使った受信バッファによる構成になっています。したがって、4ch 同時キャプチャを行う場合、USB デバイスを接続した直後にはキューの状態が接続タイミング等に依存するため、そのままではキャプチャするフレームのタイミングに少しのずれが生じる場合があります。外部配線を使用したフレーム同期(接続詳細はボードのハードウェア仕様書等を参照)のサンプルとして、v4l2StereoCapture サンプルも用意しています。

3.4.2. ビルド

1. `cmake .` により Makefile を生成
2. `make` により v4l2Capture をビルド
3. (オプション) `sudo make install` により v4l2Capture をインストール

*標準のインストール先は `/bin/` です。

3.4.3. 操作方法

映像表示中にキーボードを押下することで操作を行います。

キー	説明
q	プログラムを終了します。
i	情報表示の ON/OFF を切り替えます。
r	録画の開始/終了を制御します。 (録画ファイル名: "rec_(日付)_(時間).rec") データフォーマットは後述
f	顔認識の ON/OFF を切り替えます。
t	映像をぐるぐる回転させます。
l	映像を 180 度回転させて表示します。

3.4.4. コマンドラインオプション

「v4l2Capture」は、以下のコマンドラインオプションに対応しています。

オプション	説明
(数字)	開くビデオデバイスのデバイス ID を指定します。 たとえば <code>/v4l2Capture 2</code> として起動すると、デバイス ID 2 のデバイスのみを開き、キャプチャを開始します。デバイス ID を指定しない場合、最大 4 台までの認識されているビデオデバイスを開き、キャプチャを開始します。
<code>-simple_window</code>	キャプチャした映像を 1 つのウィンドウにまとめて表示します。このオプションを指定しない場合、OS に認識されているビデオデバイスの数だけ表示ウィンドウが作られます。
<code>-simple_gui</code>	後述の <code>qt</code> を使用するオプションで OpenCV をビルドした場合、このオプションを指定することで、シンプルな GUI のウィンドウを生成します。
<code>-half_resolution</code> または <code>-half_window</code>	このオプションを指定すると、解像度を半分に変換したうえで RGB 変換、画面表示を行います。録画はオリジナルの解像度のままで行われます。このオプションを指定すると表示に関わる処理時間が減少するため、マルチチャンネルの高速取り込みに有効です。

3.4.5. Qt ライブラリとの併用

Qt ライブラリがインストール済みであり、OpenCV が Qt を使用するようにビルドされている場合、画面の拡大縮小、画像キャプチャボタン、マウス操作、ステータスバーなどのユーザインタフェースが追加されます。この場合、本ソフトウェアの表示ウィンドウは下図のようになります。

qt ライブラリ使用時の画面構成					
<div><div><div><div><div>①</div></div><div><div><div><div>Capture 0</div><div><div><div><div>←</div><div>→</div><div>↑</div><div>↓</div><div>⏏</div><div>🔍</div><div>🔍</div><div>🏠</div><div>🔊</div></div></div><div><div>(x=777, y=542) ~ R:76 G:68 B:68</div></div></div><div><div>Capture 1</div><div><div><div>←</div><div>→</div><div>↑</div><div>↓</div><div>⏏</div><div>🔍</div><div>🔍</div><div>🏠</div><div>🔊</div></div></div><div><div>(x=438, y=603) ~ R:141 G:146 B:149</div></div></div><div><div>Capture 2</div><div><div><div>←</div><div>→</div><div>↑</div><div>↓</div><div>⏏</div><div>🔍</div><div>🔍</div><div>🏠</div><div>🔊</div></div></div><div><div>(x=1008, y=765) ~ R:139 G:143 B:144</div></div></div><div><div>Capture 3</div><div><div><div>←</div><div>→</div><div>↑</div><div>↓</div><div>⏏</div><div>🔍</div><div>🔍</div><div>🏠</div><div>🔊</div></div></div><div><div>(x=505, y=6) ~ R:111 G:114 B:99</div></div></div></div></div><div>(-simple_window オプションが指定されている場合)</div><table><tr><td>①画像ウィンドウ</td><td>取り込まれている映像を表示するウィンドウです。 オプションに応じて、1 - 4 つのウィンドウが表示されます。 SVM 基板に 1-4 の ID を割り振った場合、画面レイアウトは以下の通りに調整されます。 [1] [2] [3] [4]</td></tr><tr><td>②ツールバー</td><td>拡大縮小などの操作を行うための GUI です。 Qt を使用する場合表示されます。</td></tr></table></div></div></div></div>		①画像ウィンドウ	取り込まれている映像を表示するウィンドウです。 オプションに応じて、1 - 4 つのウィンドウが表示されます。 SVM 基板に 1-4 の ID を割り振った場合、画面レイアウトは以下の通りに調整されます。 [1] [2] [3] [4]	②ツールバー	拡大縮小などの操作を行うための GUI です。 Qt を使用する場合表示されます。
①画像ウィンドウ	取り込まれている映像を表示するウィンドウです。 オプションに応じて、1 - 4 つのウィンドウが表示されます。 SVM 基板に 1-4 の ID を割り振った場合、画面レイアウトは以下の通りに調整されます。 [1] [2] [3] [4]				
②ツールバー	拡大縮小などの操作を行うための GUI です。 Qt を使用する場合表示されます。				

③ステータスバー	マウス位置の画素情報を表示します。 Qt を使用する場合表示されます。
④デバイス名	デバイス名 (“SVM-03U(1)” etc.) を表示します。
⑤情報表示	以下のような情報を表示します。 (1) ソフトウェアに取り込まれたフレーム番号 (2) 画像に埋め込まれたフレーム番号 (3) 実 FPS (直近フレームの時刻から計算)

3.4.6. 出力動画ファイルフォーマット

「v4l2Capture」で出力される動画ファイルのフォーマットを以下に示します。本ソフトウェアはファイル書き出しにのみ対応しています。この SDK にはこのフォーマットに対応した表示用ソフトウェア「v4l2Player」が付属しています。

ファイルヘッダ (8 bytes)	(char[4]) フォーマット ('v', '4', 'l', 'c')
	(int) カメラ数
カメラ情報ヘッダ 0 (134 bytes x カメラ数)	デバイス名 (NULL 終端文字列、128 bytes)
	(int) width: カメラの画像幅 [pix.]
	(int) height: カメラの画像高さ [pix.]
	(int) bits: ピクセルあたりのビット数
	(float) fps: フレームレート
	(char[4]) pixelFormat: FourCC
	(int) padding: "0"
カメラ情報ヘッダ 1 ...	
フレーム #0 (width x height x bits x channels / 8 [bytes])	カメラ #1 のフレームデータ (左上から右下方向)
	カメラ #2 のフレームデータ
	カメラ #3 のフレームデータ
	カメラ #4 のフレームデータ
フレーム #1	カメラ #1 のフレームデータ
	...
...	

※ 弊社の SVI シリーズ標準動画フォーマット FRM 形式や AVI 形式には対応していません。

3.4.7. 備考

このプロジェクトでは libsvm ライブラリを使用しているため、ビルドするためにはあらかじめ libsvm ライブラリのインストールが必要です。

複数台のデバイスを使用する場合には、ラップトップ PC の内蔵カメラが有効になっている場合、内蔵カメラも SVM ボード同様のデバイスとして OS に認識されるため、正常に動かない可能性があります。

3.5. v4l2Capture_split プロジェクト

SDK_Linux

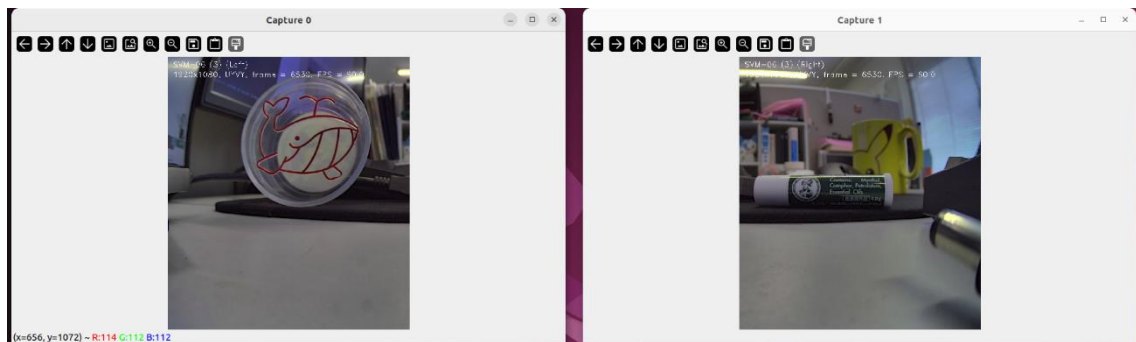
|—— v4l2StereoCapture/
|—— **v4l2Capture_split/**
|—— v4l2Player/
|—— ...

ファイル名	説明
CMakeLists.txt	cmake ツール用設定ファイルです。
CommandParser.h	コマンドライン解析用クラスです。
main.cpp main.h	main 関数を含むプログラムの主要部分のソースコードです。 v4l2Capture プロジェクトと比較して、画像分割処理が追加されています。
TimeConsManager.h	消費時間計測用クラスです。 (デバッグ・開発用)
UserFilter.cpp UserFilter.h	ユーザ定義フィルタを記述するファイルです。 UserFilter で定義された関数はため、OpenCV を使用した画像処理を記述するためのテンプレートになっています。 ユーザが画像処理を挿入する場合、このファイルにプログラムを記述すると便利です。

3.5.1. 説明

v4l2Capture_split は v4l2Capture プロジェクトをベースにしたソフトウェアで、入力画像を左右に 2 分割して独立ウィンドウに表示する処理が挿入されています。下図のように 1 つのデバイスからの映像が 2 つのウィンドウに表示され、UserFilter.cpp を編集することで画面の左右に対し独立した処理を行うことができます。Side-by-side 構成のステレオカメラでの使用時や、OpenCV を使った Mat 行列処理のリファレンスに使用してください。

(通常のカメラを用いた実行例)



(Side-by-side 構成のステレオカメラを用いた実行例)



3.5.2. 備考

このプロジェクトでは libsvm ライブラリを使用しているため、ビルドするためにはあらかじめ libsvm ライブラリのインストールが必要です。

3.6. v4l2SDLCapture プロジェクト

v4l2Capture と同じですが、表示部分を SDL に変更しています。多くの環境で v4l2Capture より高速に表示できますが、OpenCV 版で実装された映像処理や情報表示機能は削除しています。

3.7. v4l2Player プロジェクト

SDK_Linux

```
├── v4l2StereoCapture/
├── v4l2Capture_split/
├── v4l2Player/
└── ...
```

ファイル名	説明
CMakeLists.txt	cmake ツール用設定ファイルです。
CommandParser.h	コマンドライン解析用クラスです。
main.cpp	main 関数を含むプログラムの主要部分のソースコードです。
main.h	v4l2Capture プロジェクトと比較して、画像分割処理が追加されています。
FrameConverter.h	YUV/RGB 変換を行うためのクラスです。

3.7.1. 説明

「v4l2Player」は、前述の「v4l2Capture」で録画された映像ファイルを再生するためのソフトウェアです。コマンドラインオプションで読み込みファイルを指定することで、最大 4ch までの SVM ボードによって取り込まれた録画データを、1 つのファイルから同時に再生することができます。ファイル末尾まで再生されると、再び先頭に戻り、繰り返し再生が行われます。

3.7.2. ビルド

1. `cmake .` により Makefile を生成
2. `make` により v4l2Player をビルド
3. (オプション) `sudo make install` により v4l2Player をインストール

*標準のインストール先は /bin/ です。

3.7.3. コマンドラインオプション

「v4l2Player」は、以下のコマンドラインオプションに対応しています。

オプション	説明
-play [filename]	再生するファイルを指定します。ファイル名を省略した場合、「capture.cap」という名前のファイルを読み込みます。

3.8. ArroundView プロジェクト

SDK_Linux

```

├── v4l2StereoCapture/
├── v4l2Capture_split/
├── v4l2Player/
├── ArroundView/
└── ...

```

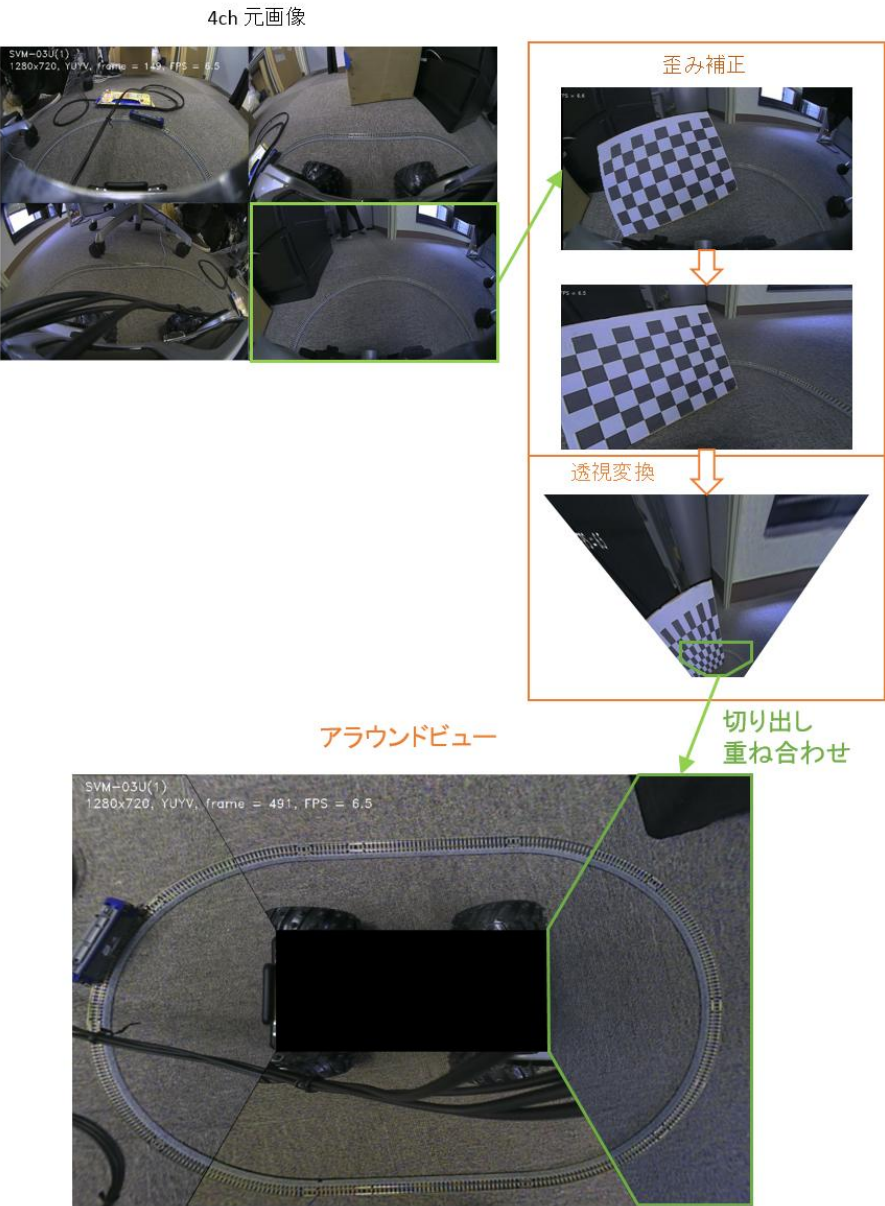
ファイル名	説明
CMakeLists.txt	cmake ツール用設定ファイルです。
ArroundView.cpp ArroundView.h	アラウンドビューのメイン処理の書かれたクラスです。
avparam.txt	各カメラの重ね合わせパラメータです。
FrameConverter.h	YUV/RGB 変換クラスです。
main.cpp main.h	main 関数を含むプログラムの主要部分のソースコードです。
SettingFileReader.cpp SettingFileReader.h	設定ファイル読み込み用クラスです。
settings	カメラ共通のひずみ補正パラメータです。
settings0 settings1 settings2 settings3	各カメラ固有のひずみ補正パラメータです。
UVCManager.cpp UVCManager.h V4LCapture.cpp V4LCapture.h	libsvm ライブラリ参照

3.8.1. 説明

4 台のカメラから 1 枚のトップビュー画像を生成する「アラウンドビュー」のデモプロジェクトです。v4l2Capture プロジェクトをベースとしており、OpenCV ライブラリによって画像処理を行っています。各カメラのひずみ補正、透視変換を行うことで、次ページの図のように、広角カメラの映像からひずみの少ないトップビュー画像をリアルタイムに表示することができます。

カメラひずみ補正パラメータの推定は、SDK 付属の undistort_init プロジェクトによって行います。

*** 本プロジェクトの動作はノンサポートとします。**



3.9. SVMSender プロジェクト

SDK_Linux

- v4l2StereoCapture/
- v4l2Capture_split/
- v4l2Player/
- SVMSender/
- ...

ファイル名	説明
CMakeLists.txt	cmake ツール用設定ファイルです。

CommandParser.h	コマンドライン解析用クラスです。
main.cpp main.h	main 関数を含むプログラムの主要部分のソースコードです。
SenderFunc.cpp SenderFunc.h	I2C 等のコマンド送信と I2C データファイル解析を行う関数を含むソースコードです。
TextFileReader.cpp TextFileReader.h	テキストファイル読み込み用ヘルパクラスです。

3.9.1. 説明

SVMSender は、I2C 送受信や FPGA レジスタ変更(オプション)などを行うための CUI ソフトウェアです。SVM ボードでは、USB Video Class に規定された Extension Unit を経由して、デバイスへの I2C 送受信や設定の送信を行うことができます。SVMSender は前述のライブラリ「libsvm」を使用することで、Extension Unit 経由で SVM ボードにコマンドを送信します。

本ツールの操作はすべて起動時のコマンドラインオプションによって指定されます。コマンドラインオプションの詳細については後述します。

(I2C データ送信時)

```

a@dynamabook: ~/Desktop/sdk_sample/v4l2Capture_split
a@dynamabook:~/Desktop/sdk_sample/v4l2Capture_split$ SVMSender 0 -send ../SVMSender/ov5642_720p_30fps_vs_neg_uyvy.dat
Device name = SVM-03U
Sending 598/598 ...
Send completed.
a@dynamabook:~/Desktop/sdk_sample/v4l2Capture_split$

```

3.9.2. ビルド

1. `cmake` により Makefile を生成
2. `make` により SVMSender をビルド
3. (オプション) `sudo make install` により SVMSender をインストール

*標準のインストール先は /bin/ です。

3.9.3. コマンドラインオプション

標準版 SVM ボードで使用される、「SVMSender」のコマンドラインオプションを以下に示します。

オプション	説明
[DeviceID] -get_name	指定したデバイス ID の SVM ボードから、ボード名(例: SVM-03U の ボード ID が 2 のとき「SVM-03U(2)」となる)を取得します。
[DeviceID] -si [Address] [Data]	指定したデバイス ID の SVM ボードに対し、I2C データを送信します。[Address] には I2C デバイスアドレス、[Data] にはカンマ区切りの 16 進コマンド列を指定します。
[DeviceID] -send_i2c_data [Address] [Data]	
[DeviceID] -ri [Address] [Size]	指定したデバイス ID の SVM ボードから、I2C データを受信します。 (beta release)
[DeviceID] -receive_i2c_data [Address] [Size]	
[DeviceID] -send [SettingFile]	指定したデバイス ID の SVM ボードに対し、I2C シーケンスを送信します。[SettingFile] には I2C シーケンスの書かれたファイル名を指定します。ファイルフォーマットの詳細については、同梱されている Windows 版 SVMctl のマニュアルを参照してください。
[DeviceID] -send_i2c_file [SettingFile]	
[DeviceID] -set_resolution [DeviceID] [Width] [Height] ([ColorSpace])	指定したデバイス ID の SVM ボードに対し、UVC 設定の変更を行います。

3.9.4. 備考

このプロジェクトでは libsvm ライブラリを使用しているため、ビルドするためにはあらかじめ libsvm ライブラリのインストールが必要です。

3.10. SVMCtlLinux プロジェクト

SDK_Linux

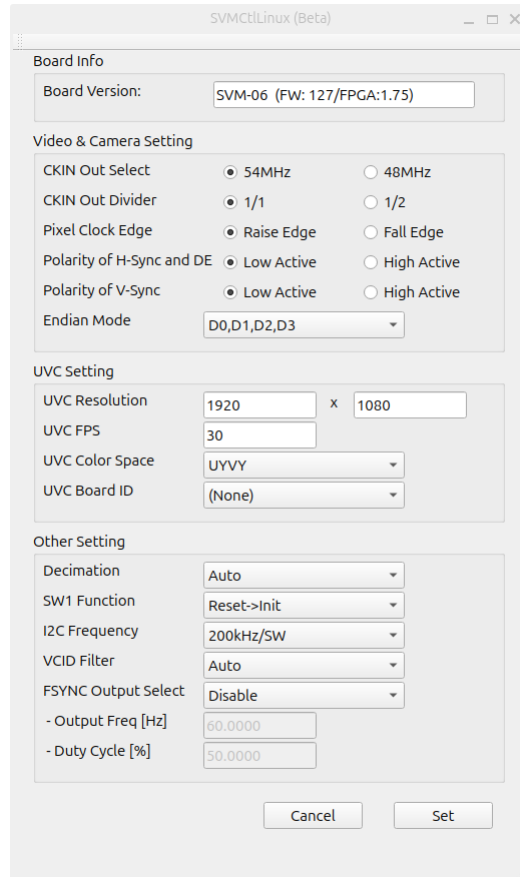
```

|—— v4l2StereoCapture/
|—— v4l2Capture_split/
|—— v4l2Player/
|—— SVMCtlLinux/
|—— ...

```

ファイル名	説明
SVMCtlLinux/SVMCtlLinux.pro	Qt Editor 用プロジェクトファイルです。
SVMCtlLinux/boardselectdialog.cpp SVMCtlLinux/boardselectdialog.h	ボード選択ダイアログのインプリメントです。
SVMCtlLinux/boardselectdialog.ui	ボード選択ダイアログの UI 定義ファイルです。
SVMCtlLinux/mainwindow.cpp SVMCtlLinux/mainwindow.h	設定ダイアログのインプリメントです。
SVMCtlLinux/mainwindow.ui	設定ダイアログの UI 定義ファイルです。

3.10.1. 説明



「SVMCtlLinux」は、SVM ボードの入力フォーマットや UVC 解像度等の設定を行うための GUI ユーティリティソフトです。ダイアログの操作方法については、Windows 版ユーティリティ「SVMCtl」の「SVM Setting」画面と同様です。「Set」ボタンを押すと、設定内容が SVM ボードの SPI-ROM に設定され、次回起動時以降に設定が反映されます。

「SVMCtlLinux」は GUI ダイアログを扱うために Qt を使用しており、Qt Creator のプロジェクトの形で提供しています。アプリケーションのビルドには別途 Qt のインストールが必要になります。

3.10.2. ビルド

1. Qt5 を使用する場合は `qmake` 、Qt6 を使用する場合は `qmake6` により Makefile を生成
2. `make` により SVMCtlLinux をビルド

3.10.3. 備考

このプロジェクトでは libsvm ライブラリを使用しているため、ビルドするためにはあらかじめ libsvm ライブラリのインストールが必要です。

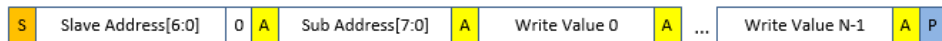
I2C 通信フォーマット

凡例

S Start
 P Stop
 A ACK(Slave)
 A ACK(Master)
 N NACK(Master)

I2C Write

Word Address: Unchecked

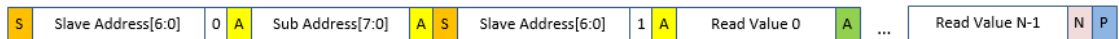


Word Address: Checked

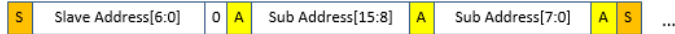


I2C Read (Restart Condition チェックあり)

Word Address: Unchecked



Word Address: Checked



I2C Read (Restart Condition チェックなし)

Word Address: Unchecked



Word Address: Checked

