VPF (Vision Processing Framework)
機能仕様書

第1版

Contents

1. 概要
2. 機能一覧・画面構成
3. 各機能仕様
   3.1 プラグイン管理機能
   3.2 画像処理フロー管理
   3.3 ストリーミング制御
   3.4 静止画保存
   3.5 ログ表示

Appendix

1. 概要

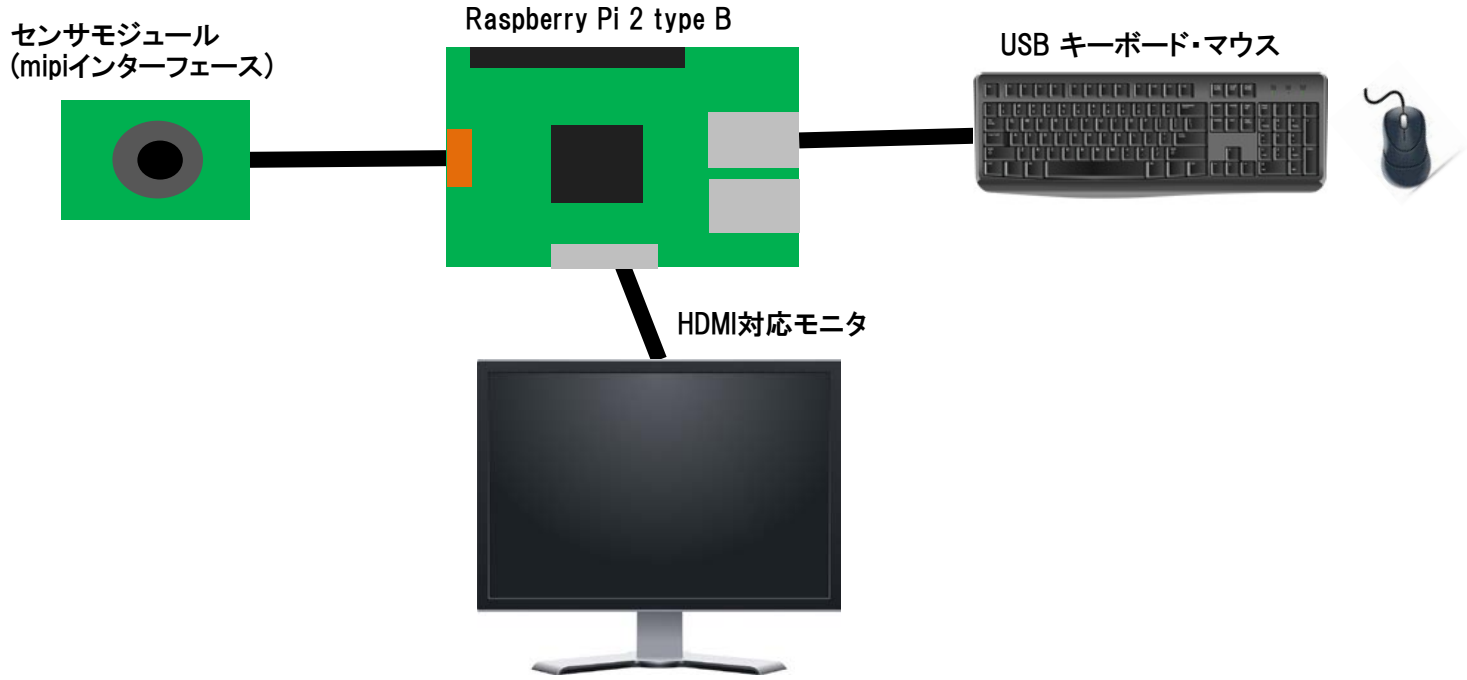
1.1 VPF概要

VPF(Vision Processing Framework)は、Raspberry Pi上で画像処理を行うためのアプリケーションフレームワークである。各種画像処理はプラグイン(共有ライブラリ)として実装し、VPFは画像処理プラグイン開発のためのプラグイン・インタフェースを提供する。(画像処理プラグインの作成方法については別資料 「3_プラグイン作成方法.pptx」参照。)

1. 概要

1.2 動作環境構成

以下にVPFの動作環境構成を示す。



以下にVPFで動作可能なセンサモジュールおよびOSを示す。

センサモジュール ： IMX219, IU233, IMX408
OS ： Raspbian Jessie (Debian 8.0)

2. 機能一覧・画面構成

2.1 機能一覧(1)

VPFの機能一覧と概要を以下に示す。各機能の詳細は3章に記載する。

機能	小項目	概要
プラグイン管理	ロード/リロード	ロード　:VPF起動時、所定のフォルダに配置されているプラグインをロードする。 リロード:VPF起動後、所定のフォルダに配置されているプラグインを再読み込みする。
	バージョンチェック	プラグインのロード(リロード)時、各プラグインが持つプラグイン・インタフェースのバージョンチェックを行い、実行可能なプラグインのみロードする。
画像処理フロー管理	構築	プラグインの接続関係を制御する事ができる。プラグインの接続関係を「画像処理フロー」と呼ぶ。ユーザは、プラグインの追加、削除、挿入、分岐、置換操作により任意の画像処理フローを構築する事ができる。画像処理フローの先頭は、画像ファイル読み込みやセンサモジュール制御などを行う入力プラグインのみ配置することができる。
	保存/読み込み	保存　　:画像処理フローおよびプラグインが持つ設定値をファイルに保存する。 読み込み:保存したファイルを読み出し、画像処理フローおよび各プラグインの設定値情報を反映する。
	接続妥当性チェック	構築した画像処理フローおよび各プラグインの設定をもとに、画像処理が実行可能か否かを判断する。

次ページに続く。

2. 機能一覧・画面構成

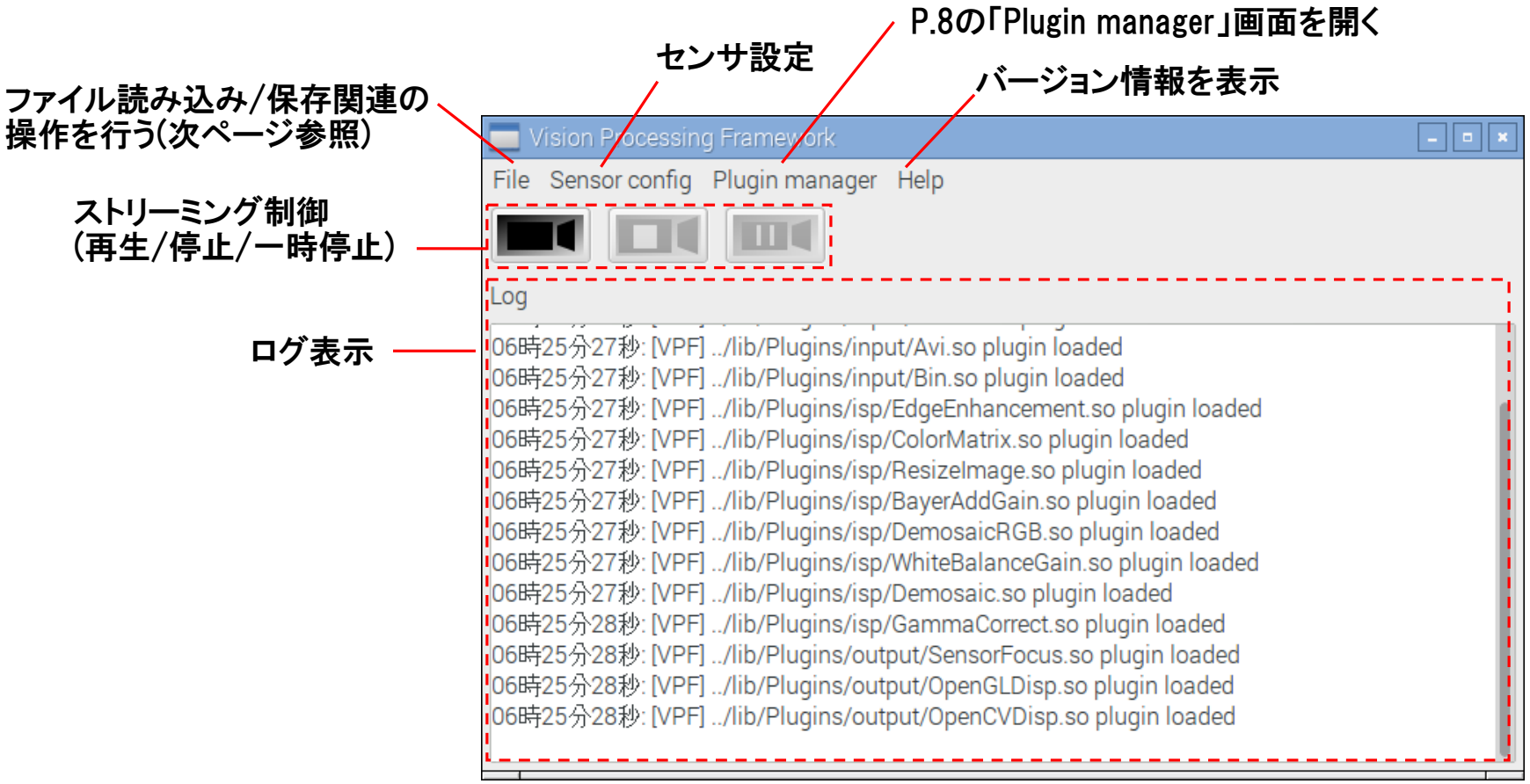
2.1 機能一覧(2)

VPFの機能一覧と概要を以下に示す。

機能	小項目	概要
ストリーミング制御	再生	構築した画像処理フローに基づき、画像処理プラグインの各種処理を実行する。
	一時停止	画像処理を一時停止する。
	停止	画像処理を停止する。画像処理プラグインの終了処理を実行するため、画像処理結果を表示するディスプレイウィンドウなどは閉じられる(プラグインの実装による)。
RAW画像読み込み	ー	Binプラグインの設定ウィンドウを表示し、Raw画像保存によって保存した画像データを読み込むことができる。 詳細は別資料「3_プラグインの説明.pptxの3-2.Binプラグイン」参照。
Avi読み込み	ー	Aviプラグインの設定ウィンドウを表示し、非圧縮AVIファイルを読み込むことができる。 詳細は別資料「3_プラグインの説明.pptxの3-3.Aviプラグイン」参照。
静止画保存	Raw画像保存	入力プラグインが出力する画像データを保存することができる。
	JPEG/PNG/BMP保存	画像処理フローの終端プラグインが出力する画像データを保存することができる。 画像処理フローが分岐している場合、保存したいプラグインを選択することができる。また、保存形式はJPEG, PNG, BMPから選択することができる。
ログ表示	ー	VPF本体と各プラグインのログを指定の書式に基づき表示する。
センサ設定	ー	Sensorプラグインの設定ウィンドウを表示し、Raspberry Piに接続しているセンサモジュールの初期化設定、アナログ/デジタルゲイン、露光時間等を設定できる。 詳細は別資料「3_プラグインの説明.pptxの3-1.Sensorプラグイン」参照。

2. 機能一覧・画面構成

2.2 メインウィンドウのUI構成 (1)



2. 機能一覧・画面構成

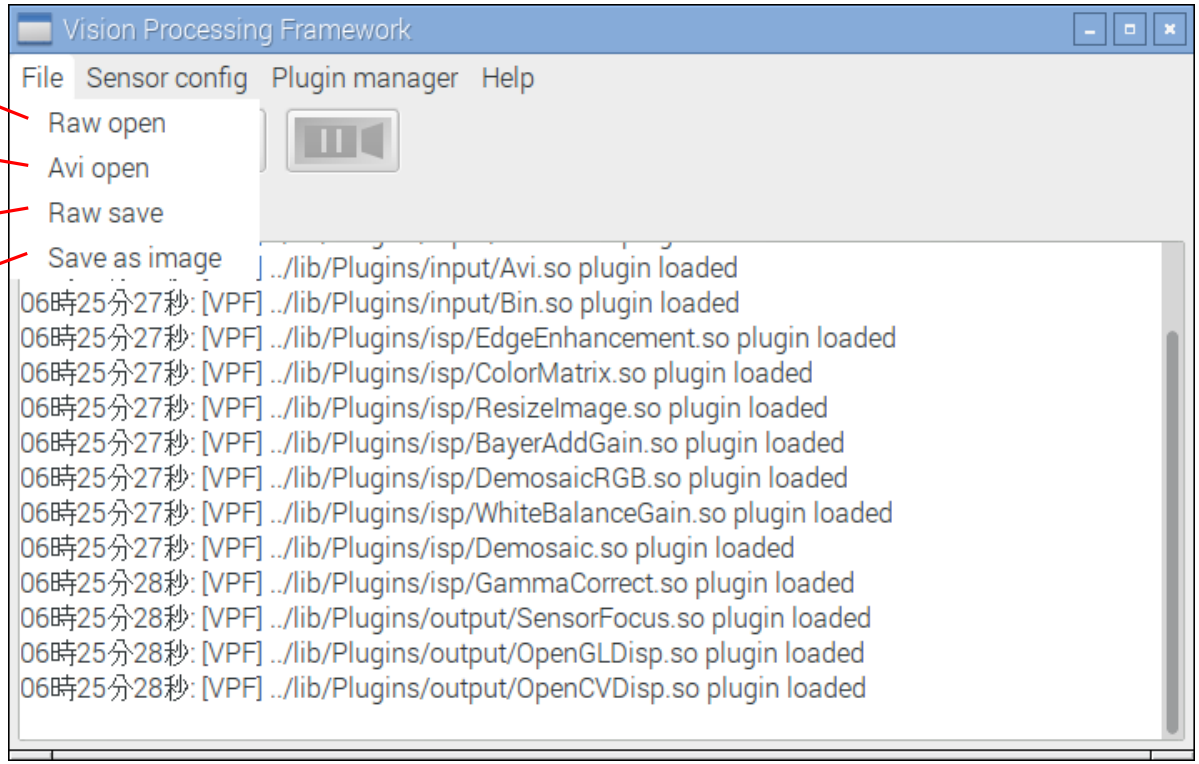
2.2 メインウィンドウのUI構成 (2)

RAW画像読み込み

AVI読み込み

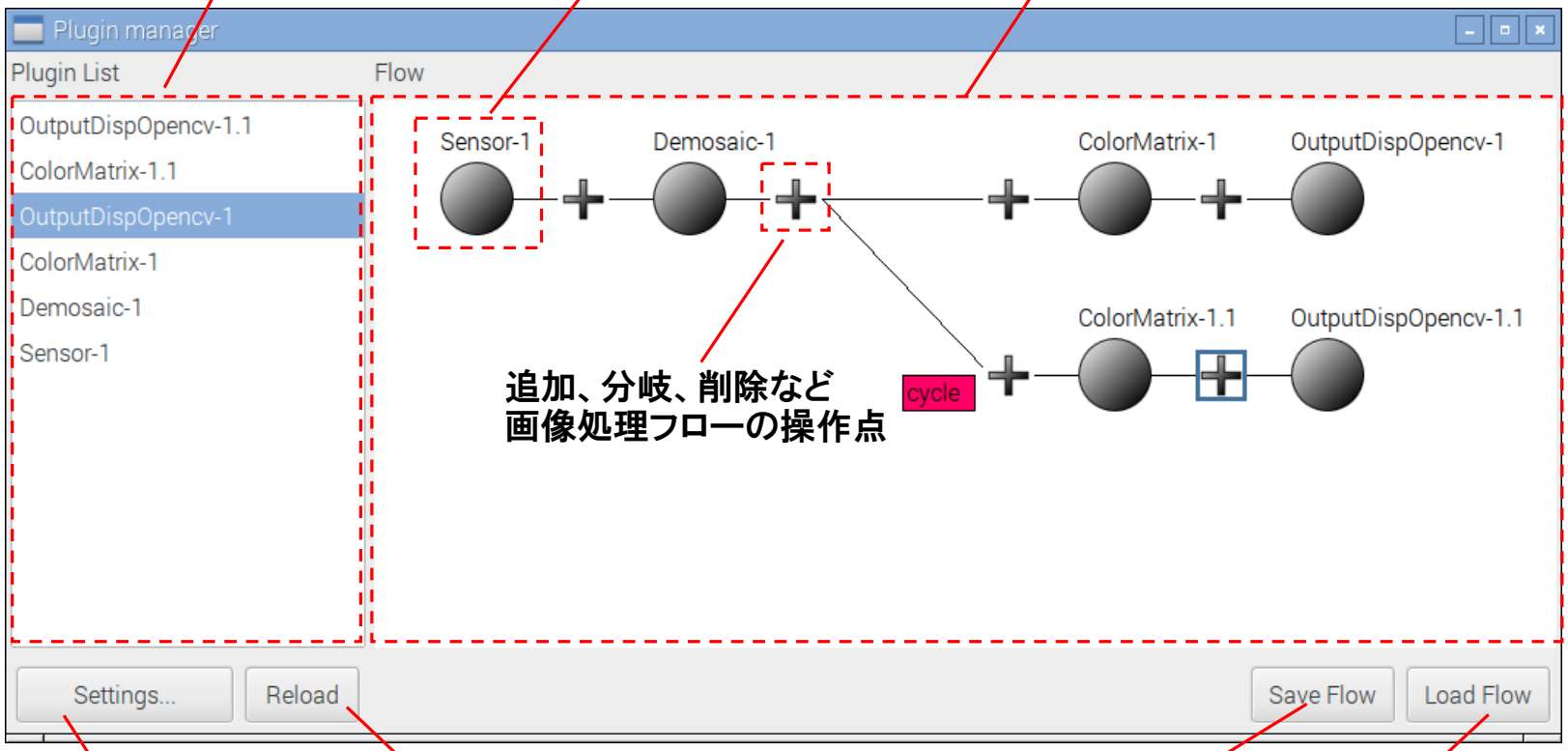
RAW画像保存

JPEG/PNG/BMP保存



2. 機能一覧・画面構成

2.3 Plugin managerのUI構成



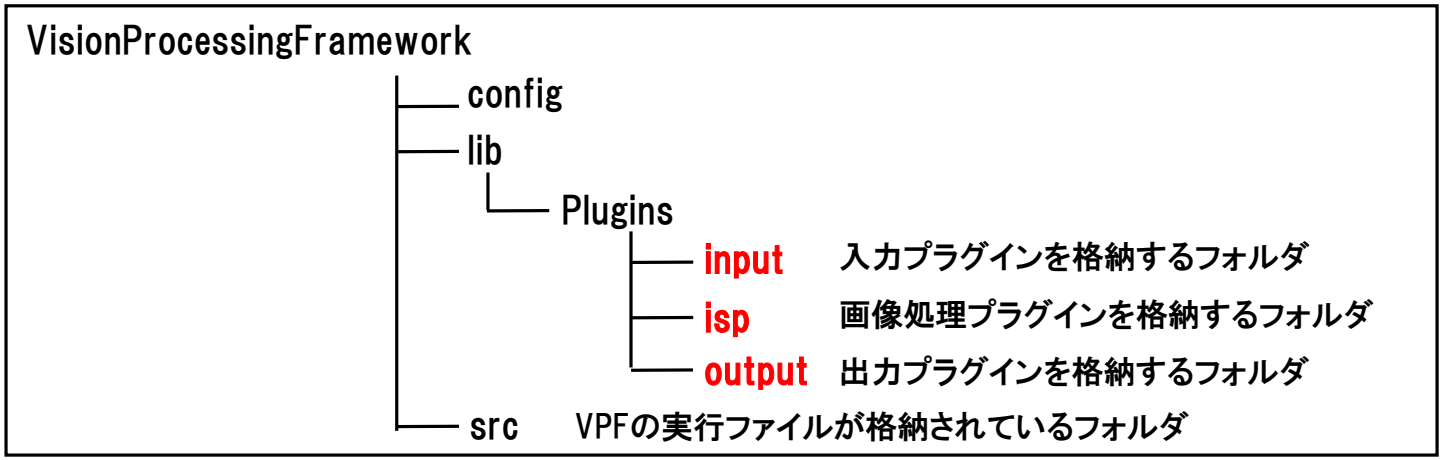
3. 各機能仕様

3.1 プラグイン管理機能

3.1.1 ロード/リロード

・ロード

VPF起動時、以下に示すディレクトリに格納されているプラグインをロードする。プラグインは種別フォルダごとに格納されており、拡張子は.soとする。



プラグイン・インタフェースは「プラグイン名」、「プラグイン・インタフェースバージョン」取得のAPIを定義しており、VPFはロードしたプラグインのプラグイン名とプラグイン・インタフェースバージョンを取得し、それらを組み合わせた名称によってプラグインを一元管理する。同一のプラグイン名とプラグイン・インタフェースバージョンを持つプラグインを複数検出した場合、当該プラグインのロードはスキップする。

・リロード

Plugin manager画面の「Reload」ボタン押下時、ロード済みプラグインを一旦アンロードした後、再度指定フォルダに格納されているプラグインをロードする。

3. 各機能仕様

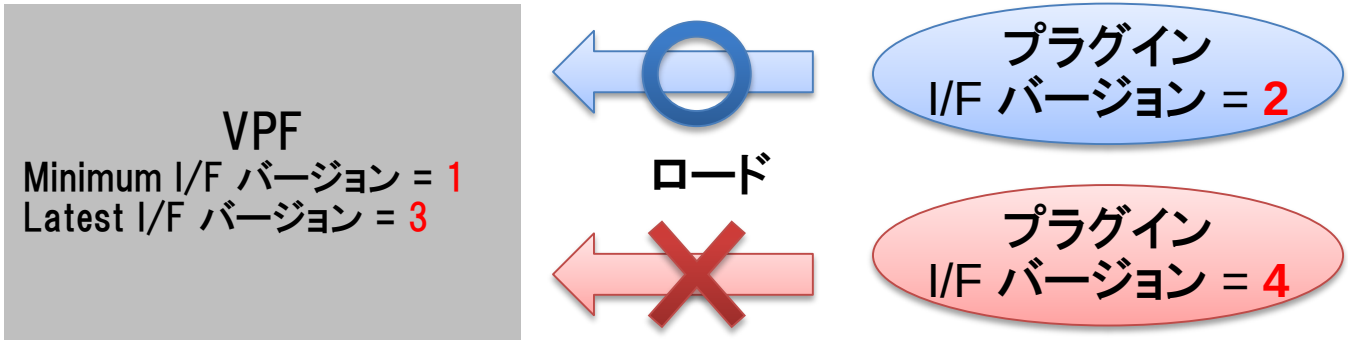
3.1 プラグイン管理機能

3.1.2 バージョンチェック

バージョンチェックの方法

・VPF本体は、以下に示すプラグイン・インタフェースバージョンに関する定義を持つ。
Minimum I/F バージョン：VPFが動作保障可能な最も古いプラグイン・インタフェースバージョン
Latest I/Fバージョン　：VPFが認識している最も新しいプラグイン・インタフェースバージョン

・プラグインロード(リロード)時、当該プラグインのプラグイン・インタフェースバージョンを取得し、以下の条件を満たす場合、実行可能なプラグインと判断する。
Minimum I/F バージョン <= 当該プラグインのI/Fバージョン <= Latest I/F バージョン

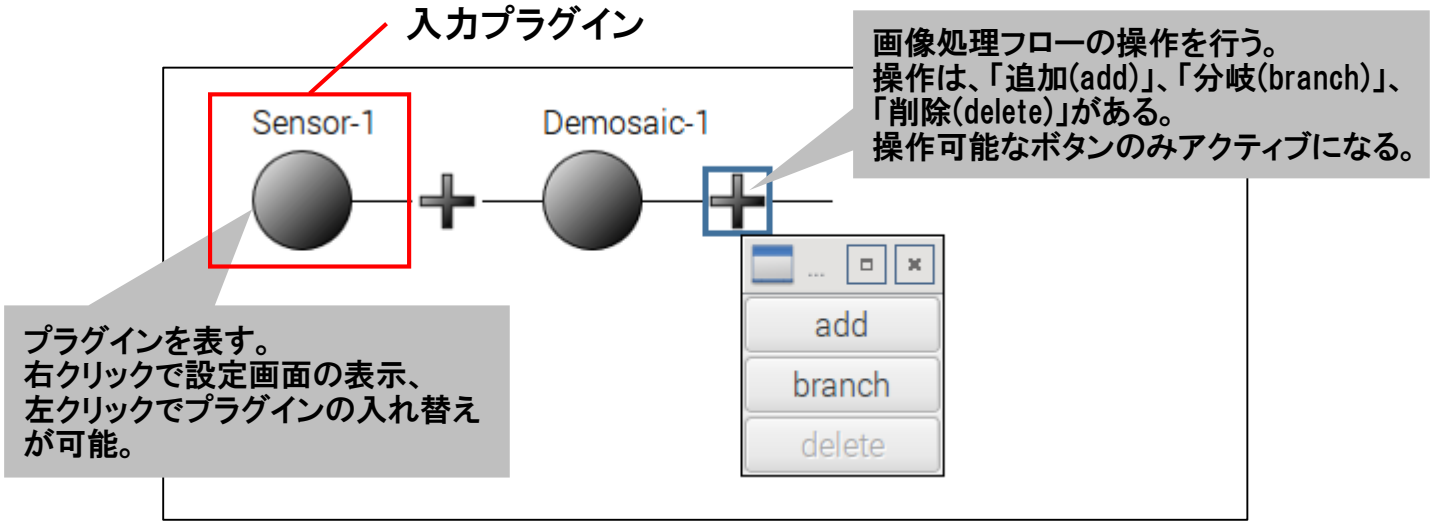


3. 各機能仕様

3.2 画像処理フロー管理

3.2.1 画像処理フロー構築

画像処理フローの構築は「Plugin manager」画面で行う。以下に構築時の操作画面を示す。フローの先頭は入力プラグインのみ配置可能とする。初期状態は、フローの先頭にロードした入力プラグインのいずれかが配置されている。入力プラグインがロードされていない場合、フロー構築エリアは空欄となり、追加等の操作は出来ない。



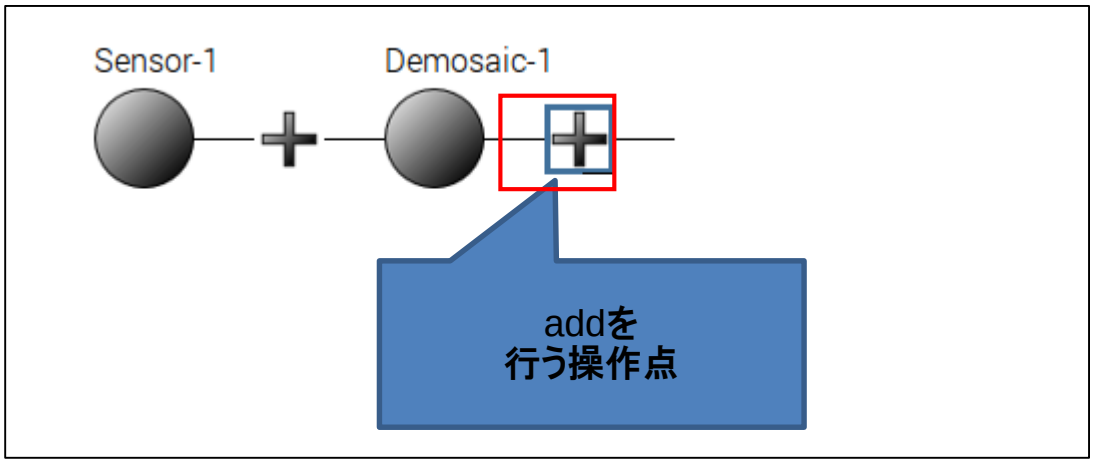
3. 各機能仕様

3.2 画像処理フロー管理

3.2.1 画像処理フロー構築

・プラグイン追加

プラグインを追加したい操作点で「add」を押下する。その際、選択した操作点に対して追加可能なプラグインの候補が表示される(次頁参照)。その中からプラグインを選択することで、フロー上に配置される。なお、前段プラグインと同名のプラグインは候補として表示されない。



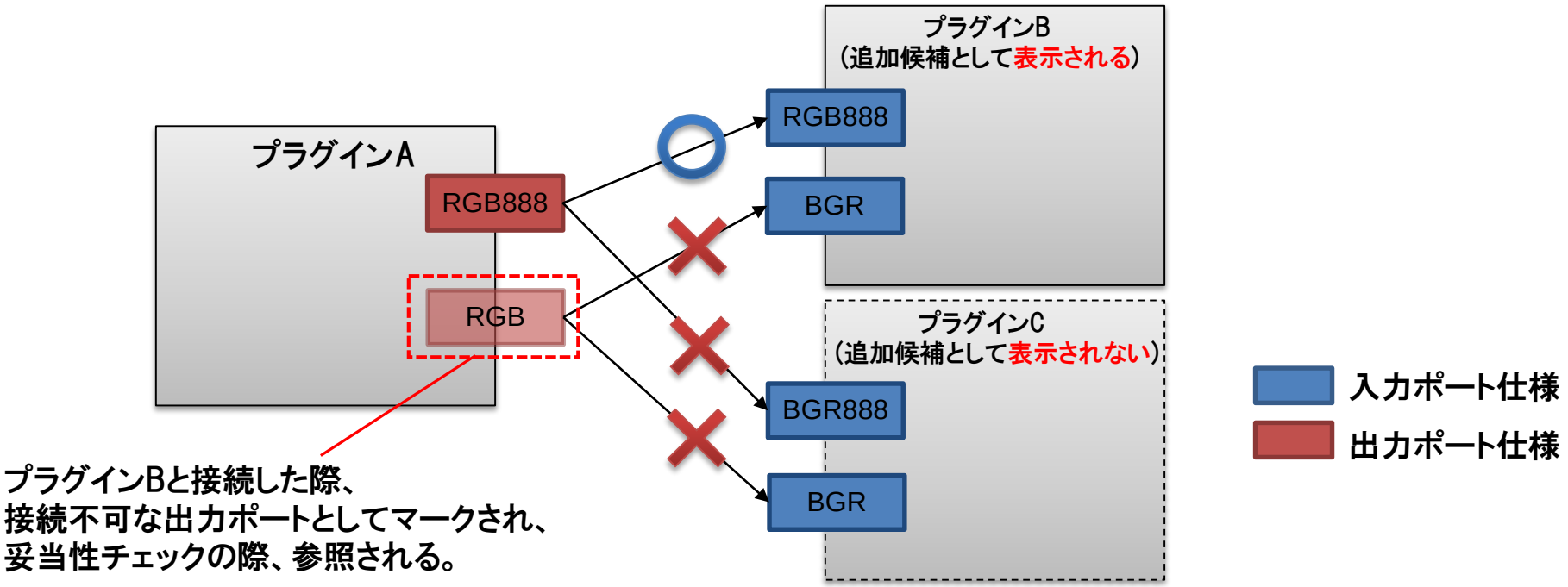
3. 各機能仕様

3.2 画像処理フロー管理

3.2.1 画像処理フロー構築

・プラグイン追加

※追加可能なプラグインの候補表示
各プラグインは「入力ポート仕様」、「出力ポート仕様」を定義している。
あるプラグインをフローに追加する際、前段プラグインの出力仕様と追加するプラグインの入力仕様が1つでも合致すれば、追加可能なプラグインの候補として表示される。



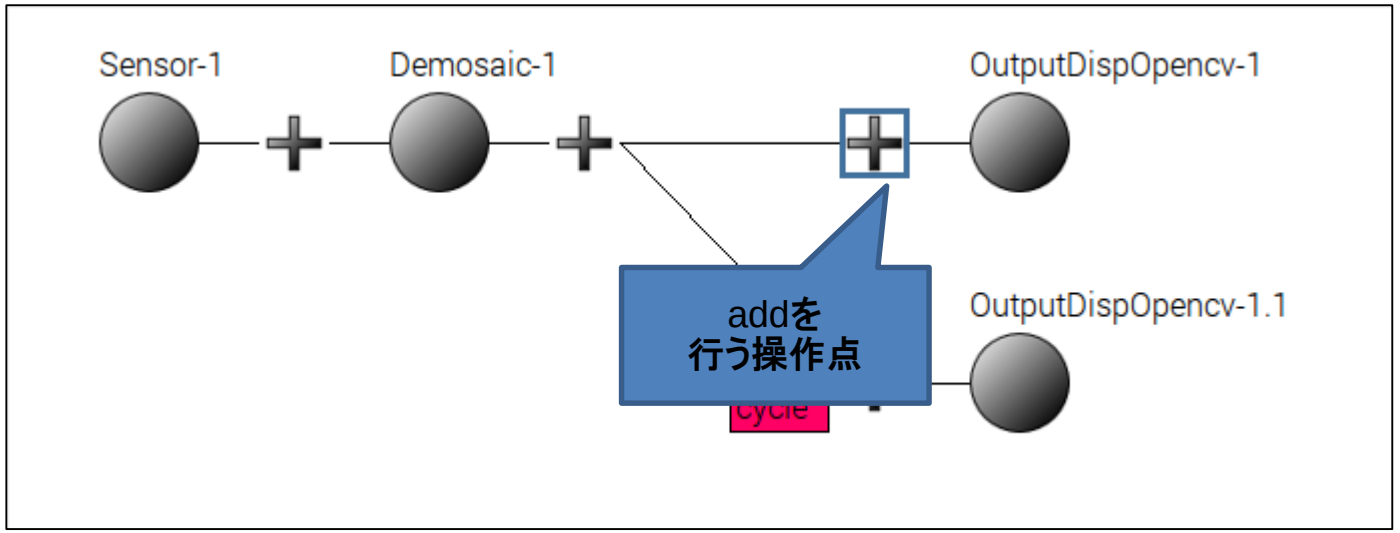
3. 各機能仕様

3.2 画像処理フロー管理

3.2.1 画像処理フロー構築

・プラグイン挿入

プラグインを挿入したい位置の操作点で「add」を押下する。挿入可能なプラグインの候補が表示されるため、その中から挿入したいプラグインを選択する。挿入する位置の前後のプラグインと同名のプラグインは表示されない。また、前後のプラグインの入出力仕様と合致しないプラグインは表示されない。



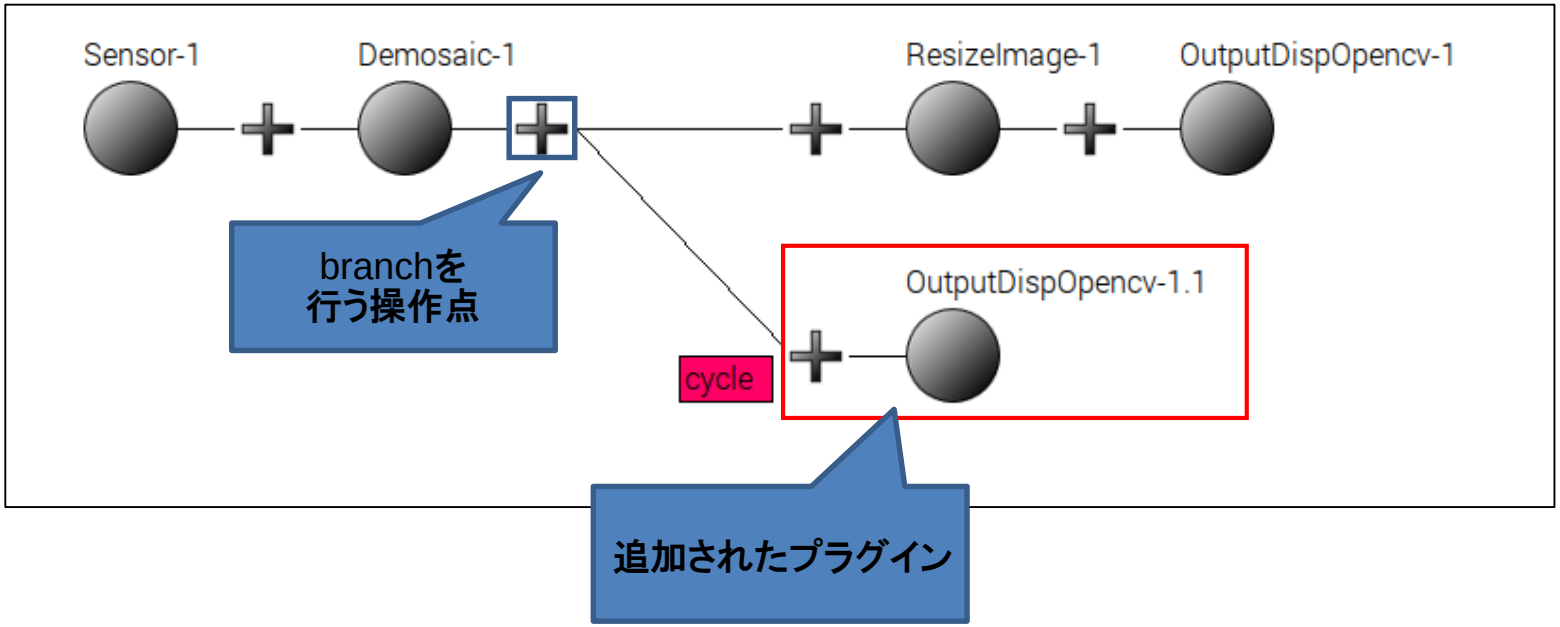
3. 各機能仕様

3.2 画像処理フロー管理

3.2.1 画像処理フロー構築

・プラグイン分岐

分岐したい操作点で「branch」を押下する。分岐先に接続可能なプラグインの候補が表示されるため、その中から配置するプラグインを選択する。フローの分岐数は最大5つとする。5つ分岐している状態で、操作点をクリックすると「branch」が非活性化して表示される。



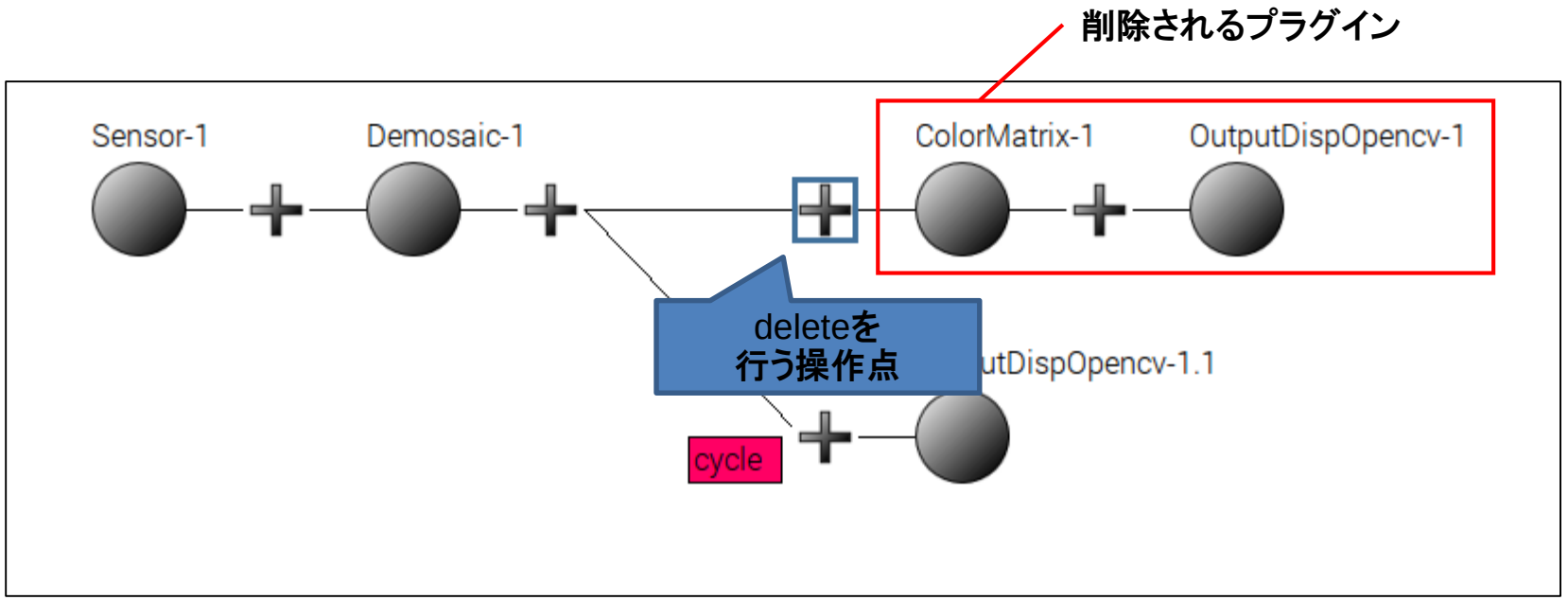
3. 各機能仕様

3.2 画像処理フロー管理

3.2.1 画像処理フロー構築

・プラグイン削除

削除したい操作点で「delete」を押下する。操作点以降のプラグインがフローから削除される。



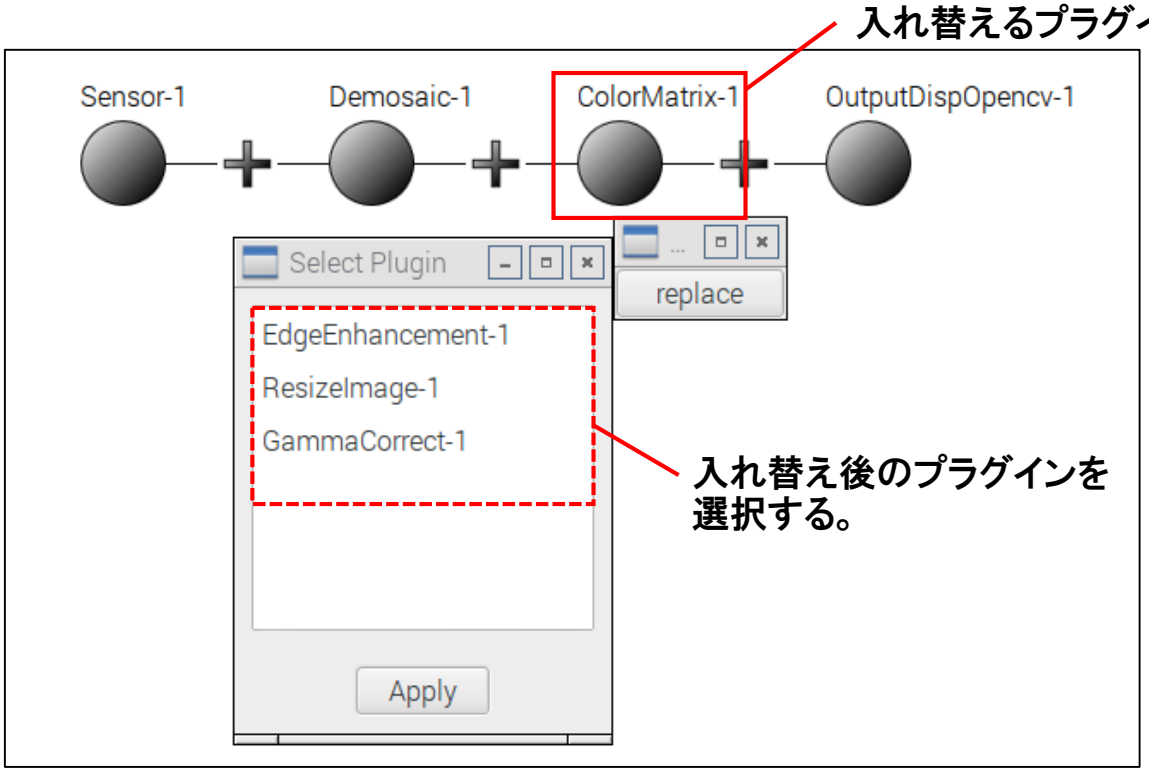
3. 各機能仕様

3.2 画像処理フロー管理

3.2.1 画像処理フロー構築

・プラグイン入れ替え

入れ替えたいプラグインを左クリックし、「replace」を押す。入れ替え可能なプラグインが候補として表示されるため、その中からプラグインを選択する。



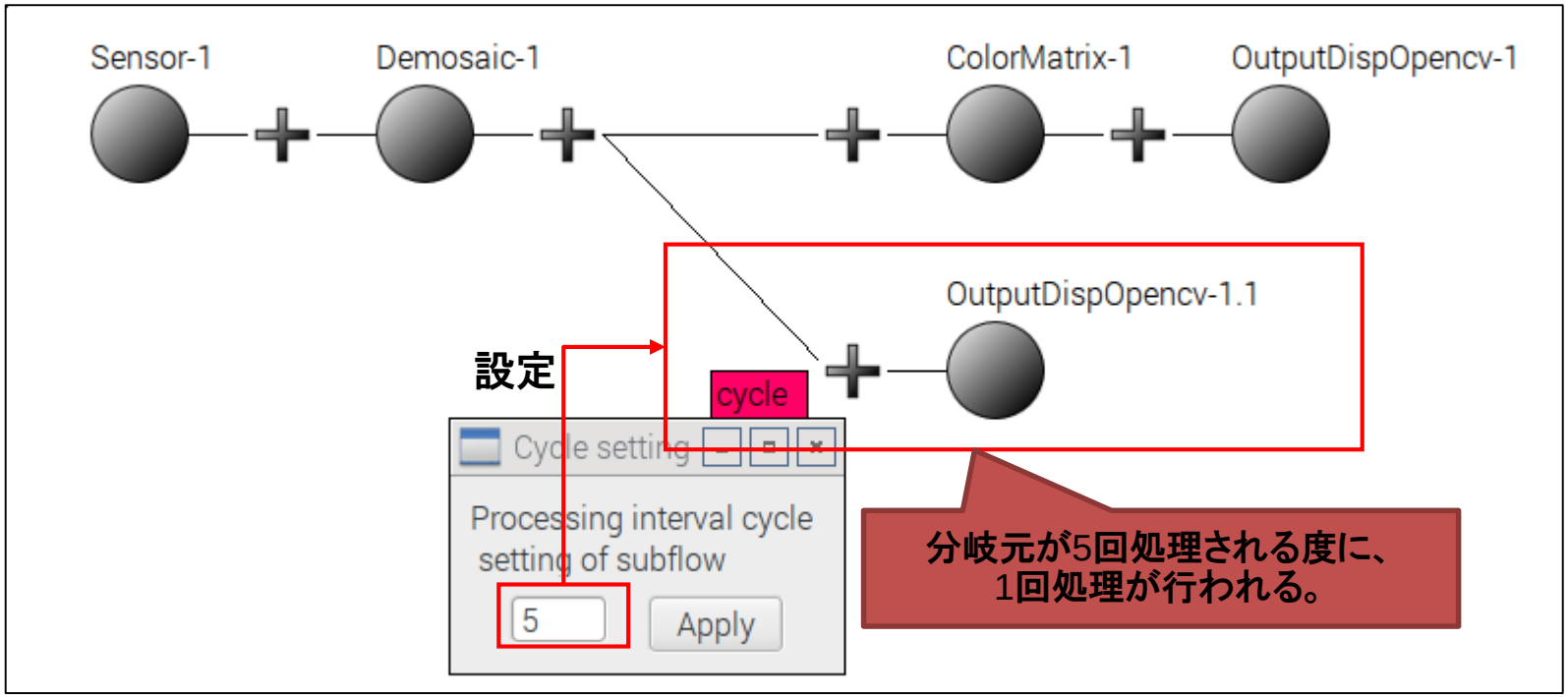
3. 各機能仕様

3.2 画像処理フロー管理

3.2.1 画像処理フロー構築

・サイクル設定

フローを分岐した場合、分岐元に対する分岐先の処理頻度(1以上の整数を指定)を指定することができる。毎フレーム処理を実行しなくてもよいプラグインは、分岐点のサイクルを設定することでフロー全体の処理速度向上を図ることができる。



3. 各機能仕様

3.2 画像処理フロー管理

3.2.2 画像処理フロー保存/読み込み

・画像処理フロー保存

Plugin manager画面で構築したフローおよびフローに配置されている各プラグインの設定値をファイルとして保存する。保存時にはファイル選択ダイアログが表示され、保存先とファイル名を指定する。ファイルの拡張子は「.flow」となる。ファイルの記載形式はAppendix3参照。

・画像処理フロー読み込み

画像処理フロー保存で作成したファイルを読み込み、フローを再構築する。また、各プラグインの設定値を反映する。読み込んだファイルに記載されているプラグインがVPFにロードされていない場合は読み込みに失敗し、その旨をエラーログに表示する。

3. 各機能仕様

3.2 画像処理フロー管理

3.2.3妥当性チェック

・概要

再生ボタン押下時、画像処理フロー全体が実行可能な接続になっているかどうかをチェックし、実行不可の接続を検出した場合は、画像処理の実行を行わずエラーダイアログを表示する。

画像処理フロー構築時は、前段プラグインと接続対象プラグインの入出力ポート仕様をチェックして接続可能なプラグインのみ配置する仕様となっているが、 前後関係の仕様チェックだけでは不十分なケースがあるため、再生開始時に本チェックを行う。次頁に例を示す。

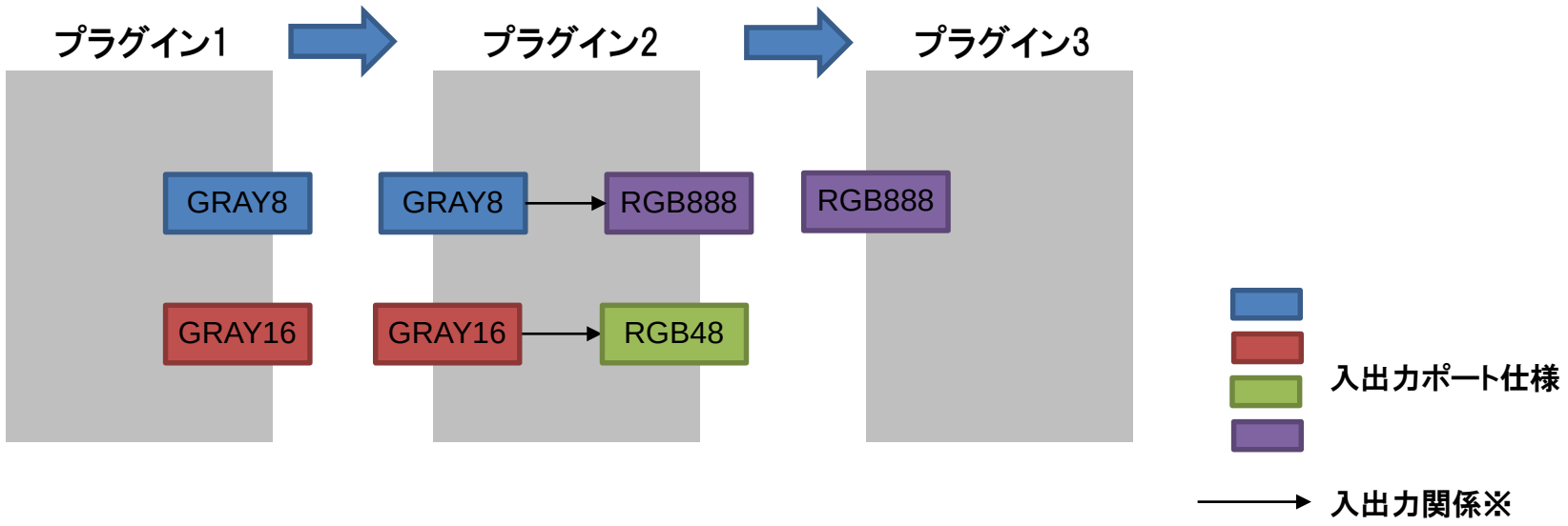
3. 各機能仕様

3.2 画像処理フロー管理

3.2.3妥当性チェック

・妥当性チェックの例

以下に示す3つのプラグインをプラグイン1, 2, 3の順に接続したときの、妥当性チェックを説明する。



※プラグイン開発時、入力に対する出力の関係も定義する。
別資料「4_プラグイン作成方法.pptx」参照。

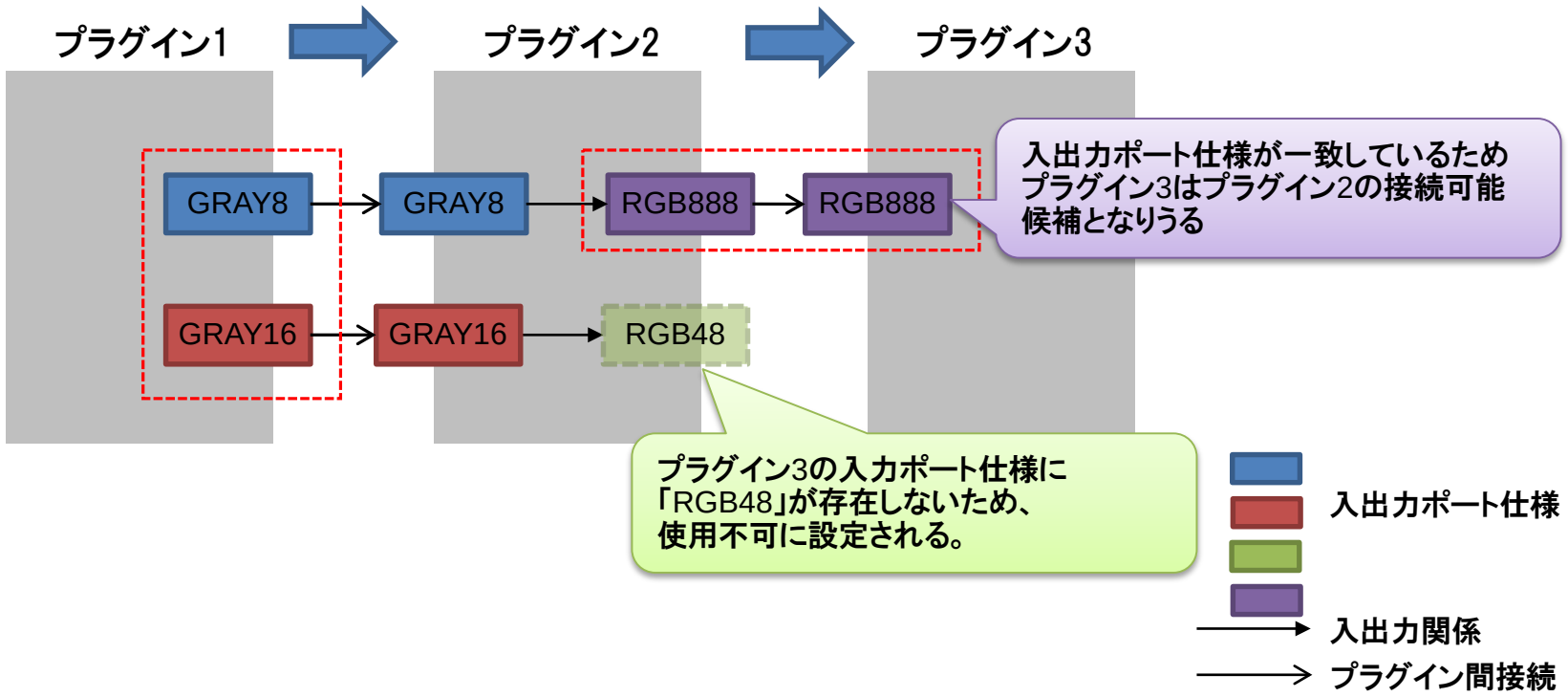
3. 各機能仕様

3.2 画像処理フロー管理

3.2.3妥当性チェック

・画像処理フロー構築完了時点

画像処理フロー構築完了の時点ではプラグイン1が「GRAY8」または「GRAY16」を出力するか確定していない。



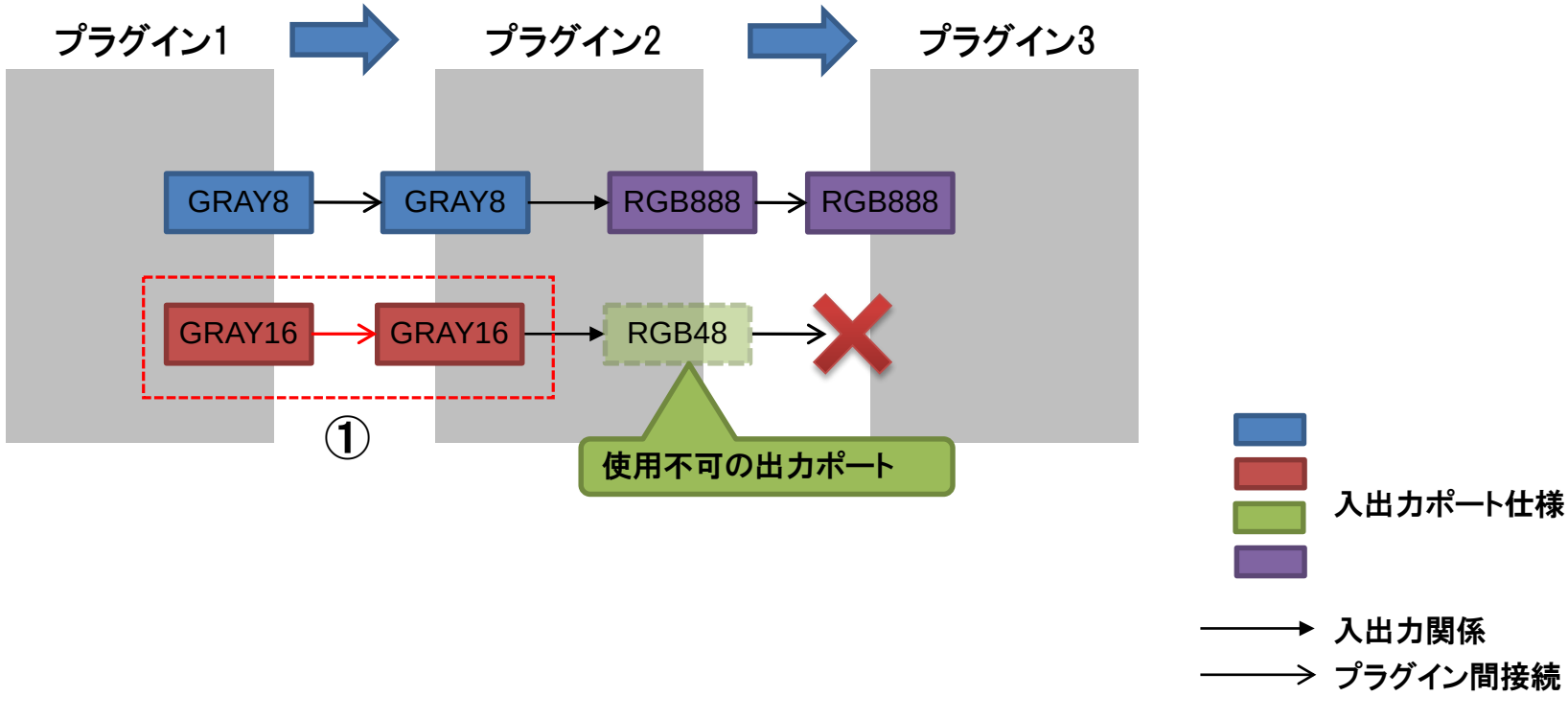
3. 各機能仕様

3.2 画像処理フロー管理

3.2.3妥当性チェック

・再生ボタン押下時

プラグイン1が「GRAY16」で出力することが確定した場合、プラグイン1とプラグイン2の接続が確定される。(下図①)
プラグイン2は入出力関係上、出力が「RGB48」に決定する。しかし、「RGB48」は使用不可に設定されているため、妥当性チェックはNGと判定する。



3. 各機能仕様

3.3 ストリーミング制御

・再生

構築した画像処理フローに基づき妥当性チェックを行い、その後画像処理を開始する。
妥当性チェックでNGと判断された場合、エラーダイアログを表示し、画像処理を実行せず停止状態にする。

・一時停止

画像処理を一時停止する。再生ボタンを押下すると、画像処理を再開する。

・停止

画像処理を停止する。画像処理プラグインの終了処理を実行するため、
画像処理結果を表示するディスプレイウィンドウなどは閉じられる(プラグインの実装による)。

・各状態における実行可能機能

以下に、再生/一時停止/停止時における実行可能機能を示す。(○:実行可能、×:実行不可)
プラグインの設定値変更は各プラグインの実装次第となる。

	再生	一時停止	停止
画像処理フロー構築	×	×	○
画像処理フロー保存	×	×	○
画像処理フロー読み込み	×	×	○
プラグインリロード	×	×	○
静止画保存	○	○	×

3. 各機能仕様

3.4 静止画保存

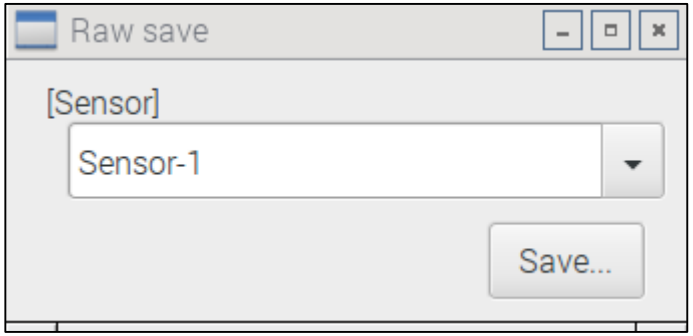
VPFでは、以下に示す2種類の静止画を保存することができる。

1. 画像処理フローの先頭プラグインが出力するデータ

画像処理フローの先頭プラグインが出力するデータを非圧縮データ(RAW)として保存する。
保存したRAWデータをBinプラグインで読み込めるように、ヘッダ(画像のサイズ情報)を付与する。

・操作方法

画像処理実行中または一時停止中に、VPF本体の「File」から「Raw save」を押下すると
以下のダイアログが表示される。
「Save...」を押下すると、ファイル保存ダイアログが表示されるため、保存先とファイル名を
指定する※。



※補足：画像処理実行中に保存を実行した場合、保存後自動的に画像処理を再開する。
一時停止中に保存を実行した場合、保存後も一時停止を継続する。

3. 各機能仕様

3.4 静止画保存

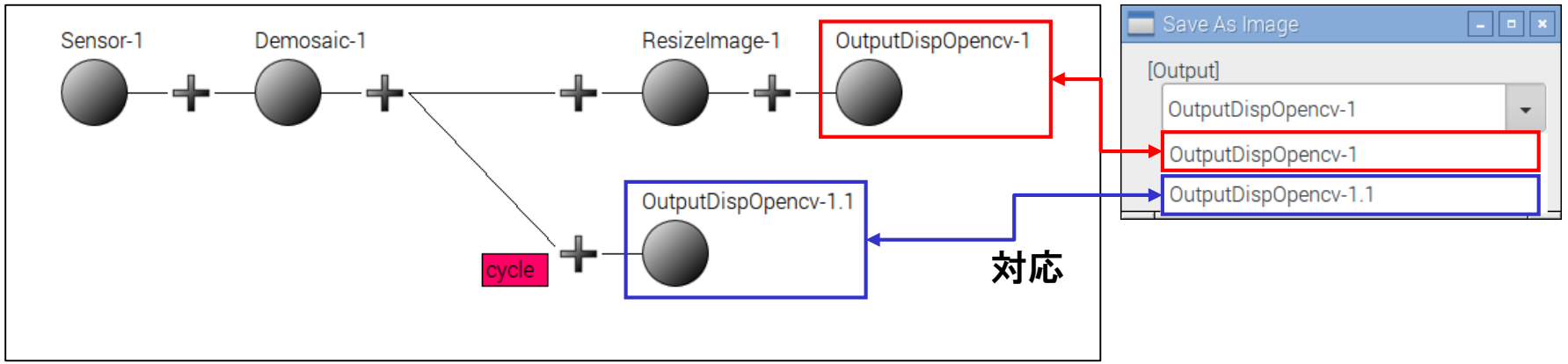
（前頁からの続き）

2. 画像処理フローの終端プラグインが出力するデータ

画像処理フローの終端プラグインが処理するデータをJPEG, BMP, PNG(保存時に1つ選択)に変換して保存する。

・操作方法

画像処理実行中または一時停止中に、VPF本体の「File」から「Save as image」を押下すると以下のダイアログが表示される。
画像処理フローが分岐している場合、保存したいプラグインを選択する(下図参照)。
「Save...」を押下すると、ファイル保存ダイアログが表示されるため、保存先、ファイル名、ファイル形式を指定する※。



※補足：画像処理実行中に保存を実行した場合、保存後自動的に画像処理を再開する。
一時停止中に保存を実行した場合、保存後も一時停止を継続する。

3. 各機能仕様

3.5 ログ表示

VPF本体または各プラグインが出力するログをVPF本体のログ表示エリアに表示する。
ログ表示行数については制限を設けていない。

・ログ出力形式
以下に示す形式でログを表示する。
<date>:<Level>:<[AppName/PluginName]>:<Message>

<date>	: HH:mm:ss形式で時刻を表示。
<Level>	: 異常系ログの場合、ログレベル([Warning]/[Error])を表示。
<AppName/PluginName>	: アプリ本体名(VPF)またはプラグイン名を表示。
<Message>	: メッセージ本文。(最大256文字)

Appendix

1. 起動方法

Raspberry Pi上のコマンドラインでVPFの実行ファイルがあるフォルダに移動し、
以下のコマンドを実行します。

```
pi@raspberryPi ~
$ sudo ./VisionProcessingFramework
```

Appendix

2. VPFが定義する画像のデータ形式

以下に、各プラグインの入出力定義で用いる画像のデータ形式※をまとめる。

#	定義値	チャンネル数	1ピクセルあたりのビット数
1	GRAY8	1	8
2	GRAY16	1	16
3	RGB444	3	12
4	BGR444	3	12
5	RGB555	3	15
6	BGR555	3	15
7	RGB565	3	16
8	BGR565	3	16
9	RGB888	3	24
10	BGR888	3	24
11	ARGB	4	32
12	RGBA	4	32
13	ABGR	4	32
14	BGRA	4	32
15	RGB48	3	48
16	BGR48	3	48
17	RGBA64	4	64
18	BGRA64	4	64

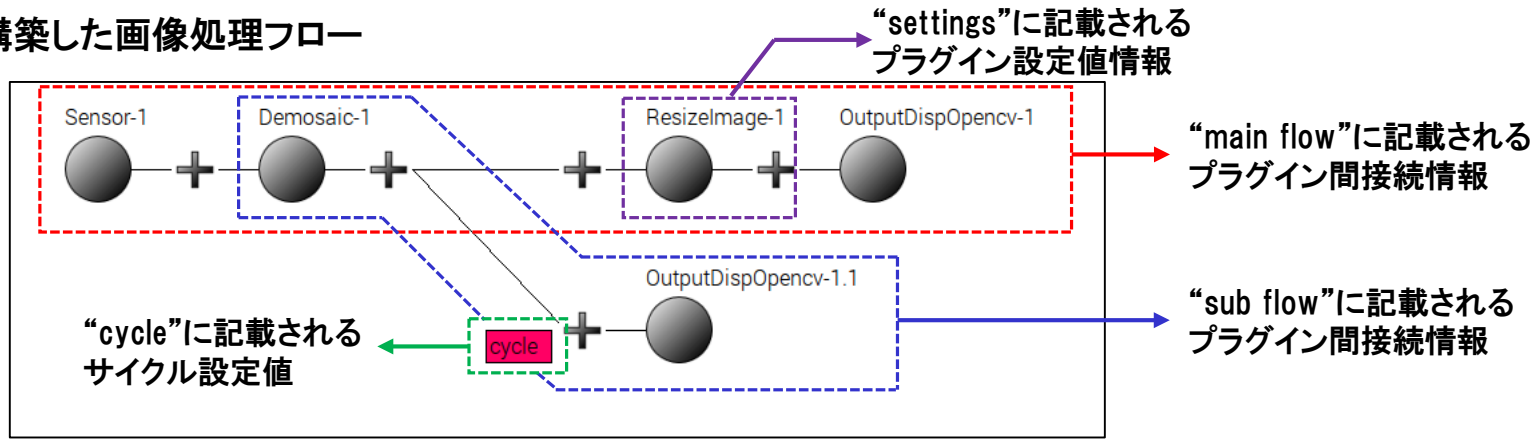
※データ形式の定義は右記を参照。/src/plugin_manager.cpp

Appendix

3. 画像処理フロー保存ファイルの記載形式

以下にPlugin manager画面で構築した画像処理フローとそれに対応する画像処理フロー保存ファイルを示す。

・構築した画像処理フロー



・出力される画像処理フロー保存ファイル

```
[main_flow]
Sensor-1,0,Demosaic-1,0,
Demosaic-1,0,ResizeImage-1,0,
ResizeImage-1,0,OutputDispOpencv-1,0,
OutputDispOpencv-1,0,,0,
[sub_flow]
Demosaic-1,0,OutputDispOpencv-1.1,1,
OutputDispOpencv-1.1,1,,0,
[settings]
name=ResizeImage-1
1
2
160
[cycle]
Demosaic-1,OutputDispOpencv-1.1,1,
```

Appendix

3. 画像処理フロー保存ファイルの記載形式(続き)

以下に画像処理フロー保存ファイルの構成を示す。