共通カメラインタフェースによるカメラコンポーネント群

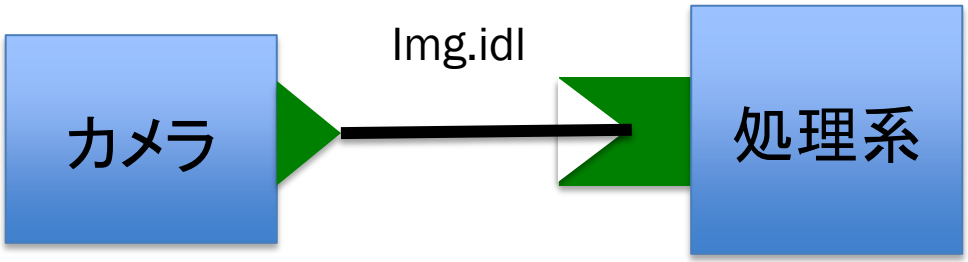
名城大学　大原賢一

共通カメラインタフェースの目的

コンポーネントの再利用性を高めるための，共通（基本）となるコンポーネント間のインタフェースの規定．NEDO知能化プロジェクトの推奨インタフェースとして位置づけ．

2014年7月に，RTM国際標準化委員会で標準認定．
OpenRTM-aist-1.1.1から正式採用

- カメラと処理系の間のインタフェースを規定



インタフェースを規定することで，異なるインタフェースによる開発を抑制し，開発したRTCの蓄積を促進する．

共通カメラインタフェース

単眼，複眼を視野に入れたカメラインタフェース

画像ファイル向け

```
#ifndef IMG_IDL
#define IMG_IDL

#include <BasicDataType.idl>

module Img {

/* vector and matrix type */
typedef double Vec3[3];
typedef double Mat44[4][4];

/* image */
enum ColorFormat
{
  CF_UNKNOWN, CF_GRAY, CF_RGB
};

struct ImageData
{
  long width;
  long height;

  ColorFormat format;
  sequence<octet> raw_data;
};
```

単眼カメラ向け

```
/* camera image */
struct CameraIntrinsicParameter
{
  double matrix_element[5];
  sequence<double> distortion_coefficient;
};

struct CameraImage
{
  RTC::Time captured_time;
  ImageData image;
  CameraIntrinsicParameter intrinsic;
  Mat44 extrinsic;
};

struct TimedCameraImage
{
  RTC::Time tm;
  CameraImage data;
  long error_code;
};
```

複眼カメラ向け

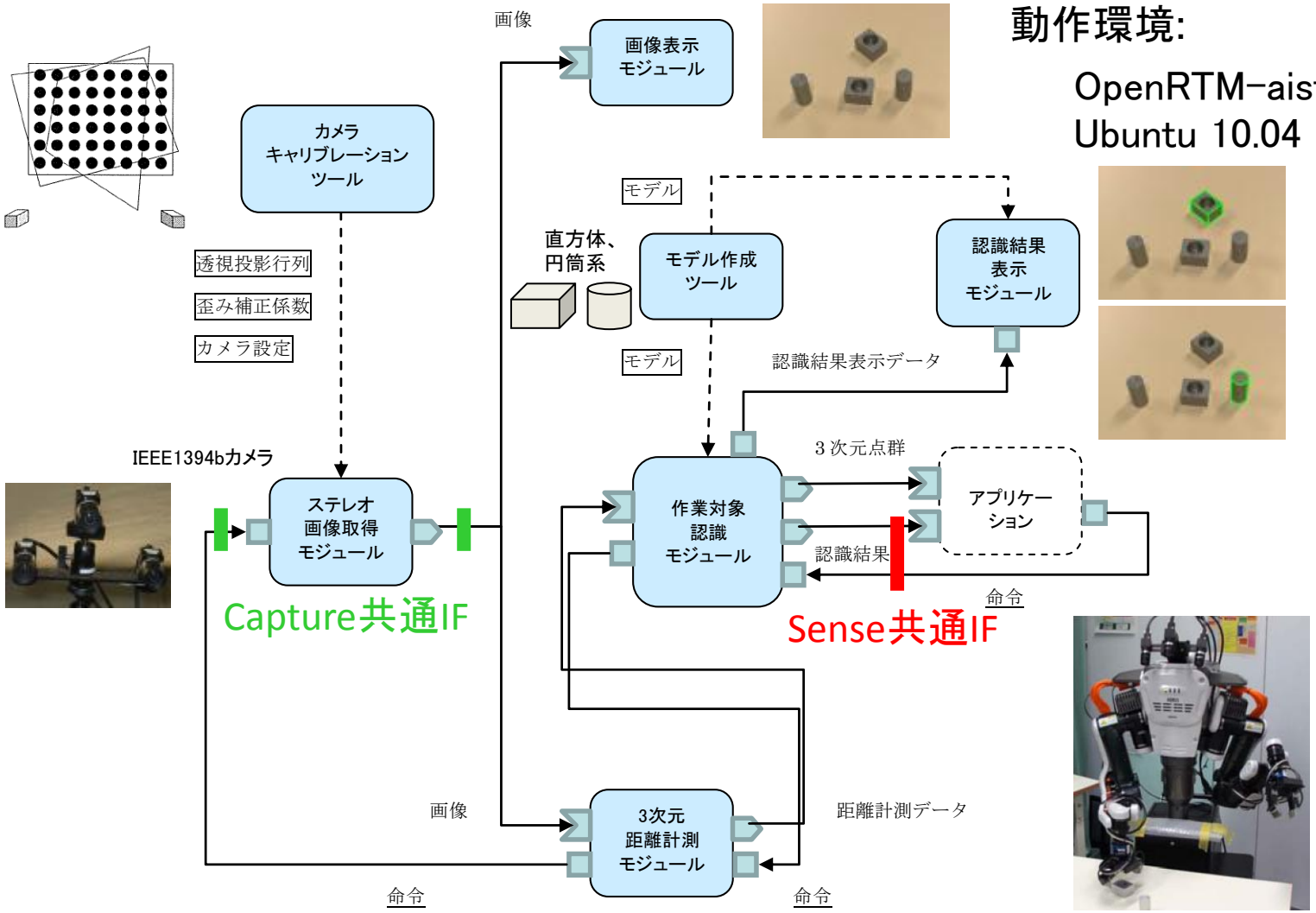
```
/* stereo camera image */
struct MultiCameraImage
{
  sequence<CameraImage> image_seq;
  long camera_set_id;
};

struct TimedMultiCameraImage
{
  RTC::Time tm;
  MultiCameraImage data;
  long error_code;
};
```

共通カメラインタフェースの特徴

- 複眼→単眼→画像ファイルというように階層化が行われており，メモリアクセスがわかりやすい．
- OpenCVとの親和性を考慮している．
  - 内部パラメータや歪みパラメータのフォーマット
- 内部パラメータを保有していることから，接続する処理コンポーネントにおいてカメラの動的な変更にも柔軟に対応可能
- カメラの使用用途に応じて，他のコンポーネントから動作モードを変更可能

対応コンポーネントの紹介

オープンソース版作業対象物体認識モジュール群(OpenVGR)



動作環境:

OpenRTM-aist-1.0.0[C++]

Ubuntu 10.04

対応コンポーネントの紹介

画像取得コンポーネント

- CaptureCamera : WebCamera用
- FireWireCamera_Img : FireWireCamera用
- LoadPicture : 画像ファイル用

表示コンポーネント

- Viewer : 画像表示

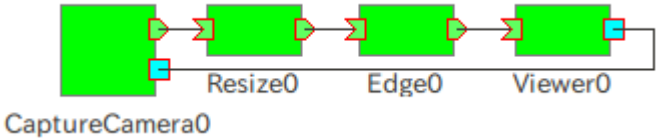
画像処理系コンポーネント

- Threshold : 画像の2値化
- DilationErosion : バイナリイメージの膨張・収縮
- Edge : エッジの検出
- Findcontour : 輪郭の検出
- Flip : 画像の上下左右回転
- HoughCircles : 円検出
- Houghline : 直線検出
- Perspective : 遠近投影
- Rotate : 画像角度回転
- Smooth : 画像の平滑化
- Translate : 映像の平行移動
- Template : パターン認識

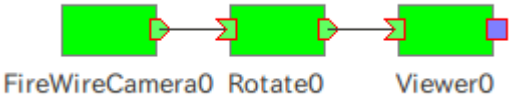
- OpenCVの機能をRTM上で利用できるようにラッピング.
- 各コンポーネントをつなぐだけで処理結果を見ることができる.
- 画像処理の学習用にも利用可能
- ソースを公開しているため，RTコンポーネントのコーディングの理解にも利用可能

対応コンポーネントの紹介

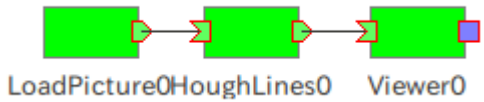
WebCamera



FireWireCamera



LoadPicture



対応コンポーネントの紹介

概要:
参照画像から得られるSIFT特徴量を用いた物体検出にGPUを用いることで、CPUのみを用いた手法と比較して、高速な目標物体の位置及び姿勢を提供する機能を実現する。

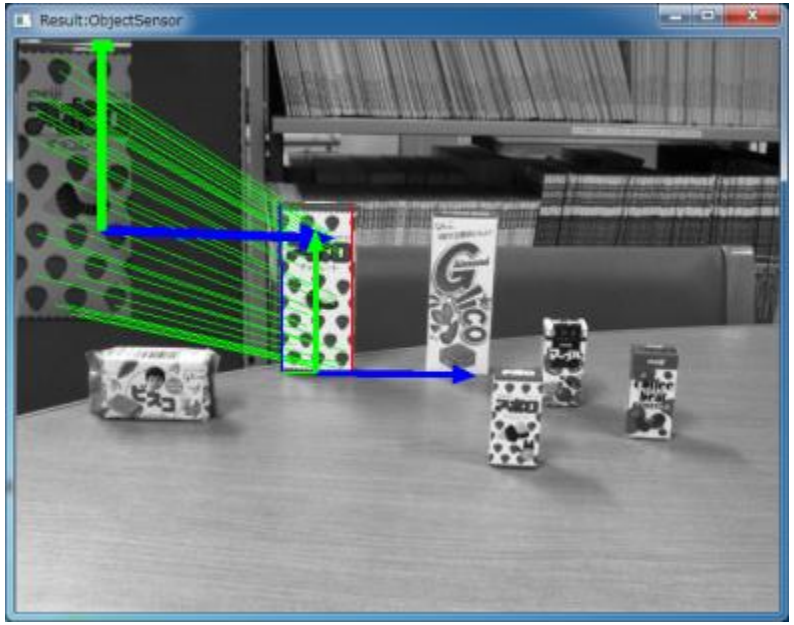
特徴:

- ◆1枚の参照画像のみで、位置姿勢を推定できる
- ◆SIFT特徴によるロバストな推定
- ◆GPUによる高速化
- ◆共通カメラインタフェースimg.idlに対応

動作環境:

OS	Ubuntu 10.04
OpenRTM	1.0.0
依存ライブラリ	OpenCV2.0以上, CUDAなど

ライセンス(公開条件):
修正BSDライセンス



動作イメージ

公開URL
http://www-arai-lab.sys.es.osaka-u.ac.jp/CameraIF/

現在1．1への対応中
少々おまちください．．．

システム構築事例

OpenHRIと共通カメラI/F対応コンポーネントを用いています.

コンポーネントの利用方法

1. https://github.com/rsdlab にアクセス.
2. WebCameraRTCとImageViewerRTCをダウンロード
3. CmakeでVSプロジェクトファイルをリリース
4. 各プロジェクトをコンパイル
5. コンパイル済みプロジェクトを実行

OpenCVの最新版を利用するために

- FlipCompのサンプルをベースにCMakeListsを編集しているものとします.
  - http://openrtm.org/ja/node/5286

- OpenCVの最新版を手軽に利用するためには，OpenCVをダウンロード後，OpenCVの環境変数を変えてあげる必要があります.
  - OpenCV_DIRという環境変数のパスを編集する.
    - 例： c:¥opencv¥build

これで，最新版のOpenCVを用いてRTCを開発することができます.

カメラを使ったRTCの運用にあたって

- 基本的な処理については，用意したRTCで可能
- ただし，対象物までの距離の計測は難しい．
- RTCをたくさんつなぐと処理速度が落ちるので，運用形態は検討が必要．

おわりに

- 画像処理系RTCについては，対応カメラも含めて鋭意増強しています（が，追いついてない．．．）
- サマーキャンプ中にも着々と増えていくかと思います
- ぜひ，みなさんも共通カメラIF対応のRTCを作って公開してみてください！