情報計測学基礎Ⅰ

作成日:2017 年 8 月 4 日
作成者:T11705M 真川智也

■ 目的

カメラから取得された動体を検出し，枠で囲む．

■ 実施内容

➢ **ROS におけるワークスペースの作成及びビルド**

My_test というワークスペースを作成する．ワークスペースとなるディレクトリを作成後，ソースディレクトリ（src）を作成．“catkin_init_workspace”コマンドによりワークスペースを作成する．

```
$ mkdir my_test
$ cd my_test
$ mkdir src
$ cd src
$catkin_init_workspace
```

“catkin_init_workspace”コマンドを実行すると src ディレクトリに「CMakeList.txt」が生成される．このファイルは“catkin_make”する際に必要なファイルである．my_test ディレクトリで“catkin_make”を実行する．

```
$ cd ../
$ catkin_make
```

正常にビルドが完了すると my_test ディレクトリに devel と buiild ディレクトリが作成される．これらのディレクトリの中に実行ファイルやライブラリファイルが保存される．作成された「devel/setup.bash」にワークスペースおよびパッケージの情報が記載されているため，“setup.bash”を用いてワークスペースのパスを追加する．

```
$ source devel/setup.bash
```

初めて catkin_make する場合や，あるいは他人のパッケージを実行する場合はフォルダ名がかわるため src ディレクトリに「CMakeList.txt」がないことを確認し，

“catkin_init_workspace”で再構築し，“catkin_make”でビルド，“setup.bash”でパスの追加をする必要がある.

- **パッケージの作成**

  パッケージの作成には“catkin_create_pkg”を実行する.

  $ catkin_create_pkg [パッケージ名] [依存パッケージ１　依存パッケージ２ ・・・]

  上記のように依存パッケージを追加する.

```
$ cd src
$ catkin_create_pkg my_opencv sensor_msgs cv_bridge roscpp std_msgs
image_transport
```

今回は my_opencv というパッケージに 5 つ™依存パッケージを追加した.

- プログラミング

  作成したパッケージディレクトリの src ディレクトリに動作させる cpp ファイルを作成する.（今回はオプティカルフローのサンプルコードを引用した：
  http://opencv.jp/sample/optical_flow.html）

```
$ cd src
$ pwd
/home/my_test/src/my_opencv/src
$ vim OpticalFlowUsingROSOpenCV.cpp
```

プログラムの書き換えが終了したら，ビルドするためにパッケージの中にある「CMakeList.txt」に下記 2 点を追加する.

```
add_executable( opticalflow src/OpticalUsingROSOpenCV.cpp )
target_link_libraries( opticalflow ${catkin_LIBRARIES} )
```

最上位のディレクトリに戻り“catkin_make”でビルドする.

```
$ cd ../../
/home/my_test
$ catkin_make
$ source devel/setup.bash
```

- プログラムの実行

  プログラムを実行する前に，新しい端末を開き下記のコマンドを実行する．

```
$ roscore
```

  さらに，別端末を開き usb カメラを立ち上げる．

```
$ rosrun usb_cam usb_cam_node
```

  プログラムを実行する．

```
$ rosrun my_opencv opticalflow
```

  ログラムが実行され，オプティカルフローが表示される．
  ※今回は動体を枠で囲むプログラムまで完成できなかった．

- Github に作成したファイルをアップロード

  ・git のインデックスに追加

  $ git add [file]

  $ git status

  ・変更結果をローカルリポジトリにコミットする．

  $ git commit -a -m “add new file”

  ・ローカルリポジトリをプッシュしてリモートリポジトリ反映

  $ git push origin master

  ※README には変更履歴（誰が，いつ，何を変更したか）が分かりやすいように記載する．また，読んだ人が分かりやすく，実装できるように書くことも重要である．