

春学期の情報計測学基礎セミナーでは Ubuntu、ROS、GAZEBO を使用した。レポートでは ROS のコマンドによる USB カメラと動作検出のプログラムの実行方法を示す。四角で囲まれた文はコンソールで入力するコマンドを表す。

- OpenCV の実装方法

Ctrl + Alt + T でコンソールを開き home ディレクトリにて mono ディレクトリを作成し、その中に src ディレクトリを作成する(mono ディレクトリの名前は任意)。

```
mkdir -p mono/src
```

mono の中の src ディレクトリへ移動する。src ディレクトリの中に my_opencv ディレクトリを作成し ROS の catkin 形式のパッケージとして五つのパッケージを作成する。参照サイト(1)の中段 4. An example ROS node を参照。

```
cd mono/src  
catkin_create_pkg my_opencv sensor_msgs cv_bridge roscpp std_msgs image_transport
```

my_opencv の中の src ディレクトリに移動する。image_converter.cpp ファイルを作成し、編集モードで起動する。

```
cd my_opencv/src  
gedit image_converter.cpp
```

以下 4 ページのソースを記入。

```
File Edit View Search Tools Documents Help
Open [icon]

1 #include <ros/ros.h>
2 #include <image_transport/image_transport.h>
3 #include <cv_bridge/cv_bridge.h>
4 #include <sensor_msgs/image_encodings.h>
5 #include <opencv2/imgproc/imgproc.hpp>
6 #include <opencv2/highgui/highgui.hpp>
7 #include <opencv2/opencv.hpp>
8 #include <opencv2/superres/optical_flow.hpp>
9
10
11 static const std::string OPENCV_WINDOW = "Image window";
12
13 cv::Mat capture, current, previous, flow, visual_flow, opening;
14
15 class ImageConverter
16 {
17     ros::NodeHandle nh_;
18     image_transport::ImageTransport it_;
19     image_transport::Subscriber image_sub_;
20     image_transport::Publisher image_pub_;
21
22 public:
23     ImageConverter()
24     : it_(nh_)
25     {
26
27         image_sub_ = it_.subscribe("/usb_cam/image_raw", 1,
28             &ImageConverter::imageCb, this);
29         image_pub_ = it_.advertise("/image_converter/output_video", 1);
30
31         cv::namedWindow(OPENCV_WINDOW);
32     }
33
34     ~ImageConverter()
35     {
36         cv::destroyWindow(OPENCV_WINDOW);
37     }
38
39     void imageCb(const sensor_msgs::ImageConstPtr& msg)
40     {
41         cv_bridge::CvImagePtr cv_ptr;
42         try
43         {
44             cv_ptr = cv_bridge::toCvCopy(msg, sensor_msgs::image_encodings::BGR8);
45         }
46         catch (cv_bridge::Exception& e)
47         {
48             ROS_ERROR("cv_bridge exception: %s", e.what());
49             return;
50         }
51
52         //カメラ画像
53         capture = cv_ptr->image;
54     }
```

```
51
52 //カメラ画像
53 capture = cv_ptr->image;
54
55 //グレイスケールへ変換
56 cvtColor(capture, current, CV_BGR2GRAY);
57
58 //前のフレームがあれば、オプティカルフローを計算し、表示する
59 if(!previous.empty()) {
60     //オプティカルフロー計算
61     cv::calcOpticalFlowFarneback(previous, current, flow, 0.5, 1, 4, 1, 5, 1.1, 3);
62
63     //オプティカルフローを可視化
64     opening = current.clone();
65
66     int flow_ch = flow.channels();
67     for(int y = 0; y < opening.rows; ++y){
68
69         float* psrc = (float*)(flow.data + flow.step * y);
70
71         for(int x = 0; x < opening.cols; ++x){
72
73             float dx = psrc[0];
74             float dy = psrc[1];
75             float r = (dx + dy);
76
77             if( 4 < r ){
78                 opening.data[ y * opening.step + x ] = 0;
79             }else{
80                 opening.data[ y * opening.step + x ] = 255;
81             }
82             psrc += flow_ch;
83         }
84     }
85
86     //クロージング
87     cv::morphologyEx(
88         opening,
89         opening,
90         cv::MORPH_CLOSE,          // モルフォロジー演算の種類
91         cv::Mat(),
92         cv::Point( -1, -1 ),
93         3,                        // 収縮と膨張が適用される回数.
94         cv::BORDER_CONSTANT,
95         cv::morphologyDefaultBorderValue()
96     );
97
98     //オープニング
99     cv::morphologyEx(
100         opening,
101         opening,
102         cv::MORPH_OPEN,          // モルフォロジー演算の種類
103         cv::Mat(),
104         cv::Point( -1, -1 ),
105         50,                      // 収縮と膨張が適用される回数.
106         cv::BORDER_CONSTANT,
107         cv::morphologyDefaultBorderValue()
108     );
109 }
```

```
File Edit View Search Tools Documents Help
Open ▾ [Icon]

107 //オープニング
108 cv::morphologyEx(
109     opening,
110     opening,
111     cv::MORPH_OPEN,          // モルフォロジー演算の種類
112     cv::Mat(),
113     cv::Point( -1, -1 ),
114     50,                      // 収縮と膨張が適用される回数.
115     cv::BORDER_CONSTANT,
116     cv::morphologyDefaultBorderValue()
117 );
118
119 //動く物体を囲むための座標
120 int x,y;
121 int s_x,s_y,g_x,g_y;
122 bool bEnd = false;
123 //上
124 for(y = 0; y < opening.rows; ++y){
125     for(x = 0; x < opening.cols; ++x){
126         if( opening.data[ y * opening.step + x ] == 0 ){
127             s_y = y;
128             bEnd = true;
129         }
130         if( bEnd ){ break; }
131     }
132     if( bEnd ){ break; }
133 }
134 bEnd = false;
135 //右
136 for(x = 0; x < opening.cols; ++x){
137     for(y = 0; y < opening.rows; ++y){
138         if( opening.data[ y * opening.step + x ] == 0 ){
139             s_x = x;
140             bEnd = true;
141         }
142         if( bEnd ){ break; }
143     }
144     if( bEnd ){ break; }
145 }
146 bEnd = false;
147 //下
148 for( y = opening.rows-1; -1 < y; --y){
149     for( x = opening.cols-1; -1 < x; --x){
150         if( opening.data[ y * opening.step + x ] == 0 ){
151             g_y = y;
152             bEnd = true;
153         }
154         if( bEnd ){ break; }
155     }
156     if( bEnd ){ break; }
157 }
158 bEnd = false;
159 //左
160 for( x = opening.cols-1; -1 < x; --x){
161     for( y = opening.rows-1; -1 < y; --y){
162         if( opening.data[ y * opening.step + x ] == 0 ){
163             g_x = x;
164             bEnd = true;
165         }
166         if( bEnd ){ break; }
167     }
168     if( bEnd ){ break; }
169 }
```

```
File Edit View Search Tools Documents Help
Open [icon]

135 }
136 bEnd = false;
137 //下
138 for( y = opening.rows-1; -1 < y; --y){
139     for( x = opening.cols-1; -1 < x; --x){
140         if( opening.data[ y * opening.step + x ] == 0 ){
141             g_y = y;
142             bEnd = true;
143         }
144         if( bEnd ){ break; }
145     }
146     if( bEnd ){ break; }
147 }
148 bEnd = false;
149 //左
150 for( x = opening.cols-1; -1 < x; --x){
151     for( y = opening.rows-1; -1 < y; --y){
152         if( opening.data[ y * opening.step + x ] == 0 ){
153             g_x = x;
154             bEnd = true;
155         }
156         if( bEnd ){ break; }
157     }
158     if( bEnd ){ break; }
159 }
160 bEnd = false;
161
162
163 //表示画面
164 visual_flow = capture.clone();
165
166 //動く物体を四角で囲む
167 cv::rectangle(visual_flow, cv::Point(s_x,s_y), cv::Point(g_x, g_y), cv::Scalar(0,0,255), 3, 4);
168
169 //表示
170 cv::imshow(OPENCV_WINDOW, visual_flow);
171 //printf("OK\n");
172
173 }
174 //前のフレームを保存
175 previous = current.clone();
176
177
178 cv::waitKey(3);
179
180 // Output modified video stream
181 image_pub_.publish(cv_ptr->toImageMsg());
182
183 }
184 };
185 };
186
187
188 int main(int argc, char** argv)
189 {
190     ros::init(argc, argv, "image_converter");
191     ImageConverter ic;
192     ros::spin();
193     return 0;
194 }
195
196 }
```

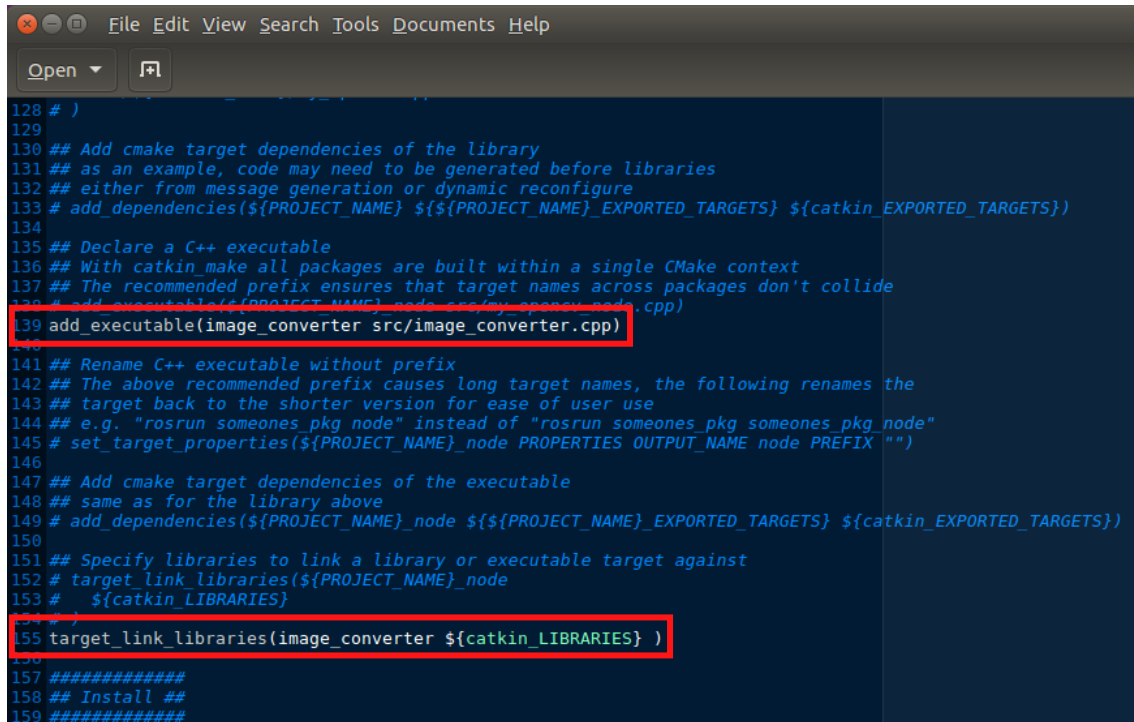
C++ Tab Width: 2 Ln 119, 0

一つ前のディレクトリへ移動し、CmakeLists.txt を編集モードで起動する。

```
cd ..  
gedit CmakeLists.txt
```

139 行目に `add_executable(image_converter src/image_converter.cpp)` を追加。

155 行目に `target_link_libraries(image_converter ${catkin_LIBRARIES})` を追加。



```
File Edit View Search Tools Documents Help  
Open [icon]  
128 # )  
129  
130 ## Add cmake target dependencies of the library  
131 ## as an example, code may need to be generated before libraries  
132 ## either from message generation or dynamic reconfigure  
133 # add_dependencies(${PROJECT_NAME} ${${PROJECT_NAME}_EXPORTED_TARGETS} ${catkin_EXPORTED_TARGETS})  
134  
135 ## Declare a C++ executable  
136 ## With catkin_make all packages are built within a single CMake context  
137 ## The recommended prefix ensures that target names across packages don't collide  
138 # add_executable(${PROJECT_NAME}_node src/image_converter.cpp)  
139 add_executable(image_converter src/image_converter.cpp)  
140  
141 ## Rename C++ executable without prefix  
142 ## The above recommended prefix causes long target names, the following renames the  
143 ## target back to the shorter version for ease of user use  
144 ## e.g. "roslaunch someones_pkg node" instead of "roslaunch someones_pkg someones_pkg_node"  
145 # set_target_properties(${PROJECT_NAME}_node PROPERTIES OUTPUT_NAME node PREFIX "")  
146  
147 ## Add cmake target dependencies of the executable  
148 ## same as for the library above  
149 # add_dependencies(${PROJECT_NAME}_node ${${PROJECT_NAME}_EXPORTED_TARGETS} ${catkin_EXPORTED_TARGETS})  
150  
151 ## Specify libraries to link a library or executable target against  
152 # target_link_libraries(${PROJECT_NAME}_node  
153 #   ${catkin_LIBRARIES})  
154 # )  
155 target_link_libraries(image_converter ${catkin_LIBRARIES})  
156  
157 #####  
158 ## Install ##  
159 #####
```

一つ前のディレクトリへ移動し、catkin のワークスペースを設定する。

```
cd ..  
catkin_init_workspace
```

一つ前のディレクトリへ移動し、catkin_make でビルドする。

```
cd ..  
catkin_make
```

- USB カメラのプログラムの実装方法

mono ディレクトリの中の src ディレクトリへ移動する。GitHub の参照サイト(2)にて usb_cam のプログラムをダウンロードする。

```
cd src
git clone https://github.com/bosch-ros-pkg/usb\_cam
```

一つ前のディレクトリへ移動し、catkin_make でビルドする。

```
cd ..
catkin_make
```

- プログラムの実行方法

mono ディレクトリにて ROS のプログラムを実行する際に初めに起動する roscore を入力する。

```
roscore
```

Ctrl + Shift + T でコンソールに二つ目のタブを追加する。新しいタブにワークスペースを配置する ROS のコマンドを入力する。さらに、usb_cam ディレクトリにある usb_cam_node のプログラムを rosrune のコマンドにより起動する。

```
source devel/setup.bash
rosrune usb_cam usb_cam_node
```

Ctrl + Shift + T でコンソールに三つ目のタブを追加する。新しいタブにワークスペースを配置する ROS のコマンドを入力する。さらに、my_opencv ディレクトリにある image_converter のプログラムを rosrune のコマンドにより起動する。

```
source devel/setup.bash
rosrune my_opencv image_converter
```

参照サイト

(1) ROS wiki OpenCV :

http://wiki.ros.org/cv_bridge/Tutorials/UsingCvBridgeToConvertBetweenROSImagesAndOpenCVImages

(2) ros-drivers/usb-cam :

https://github.com/bosch-ros-pkg/usb_cam