情報計測学基礎最終レポート

中京大学大学院 工学研究科 機械システム工学専攻
T11703M　市川 智裕

目次

① OpenCV を使うパッケージの作成方法

1．フォルダとその下に src ファイルをつくる．
$ mkdir -p my_program/src

2．my_program/src の中にパッケージをつくる．
$cd my_program/src
$catkin_create_pkg my_opencv sensor_msgs cv_bridge roscpp std_msgs image_transport

3．my_opencv/src の中ソースフォルダをつくる．
$cd my_opencv/src
$gedit optical_flow.cpp

4．プログラムを書く
プログラムの内容は4章を参照

5．CMakeList.txt の編集
$cd ..
$gedit CMakeList.txt
139行目付近に　add_executable(optical_flow src/optical_flow.cpp)
155行目付近に　target_link_libraries(optical_flow ${catkin_LIBRARIES})
を加える．

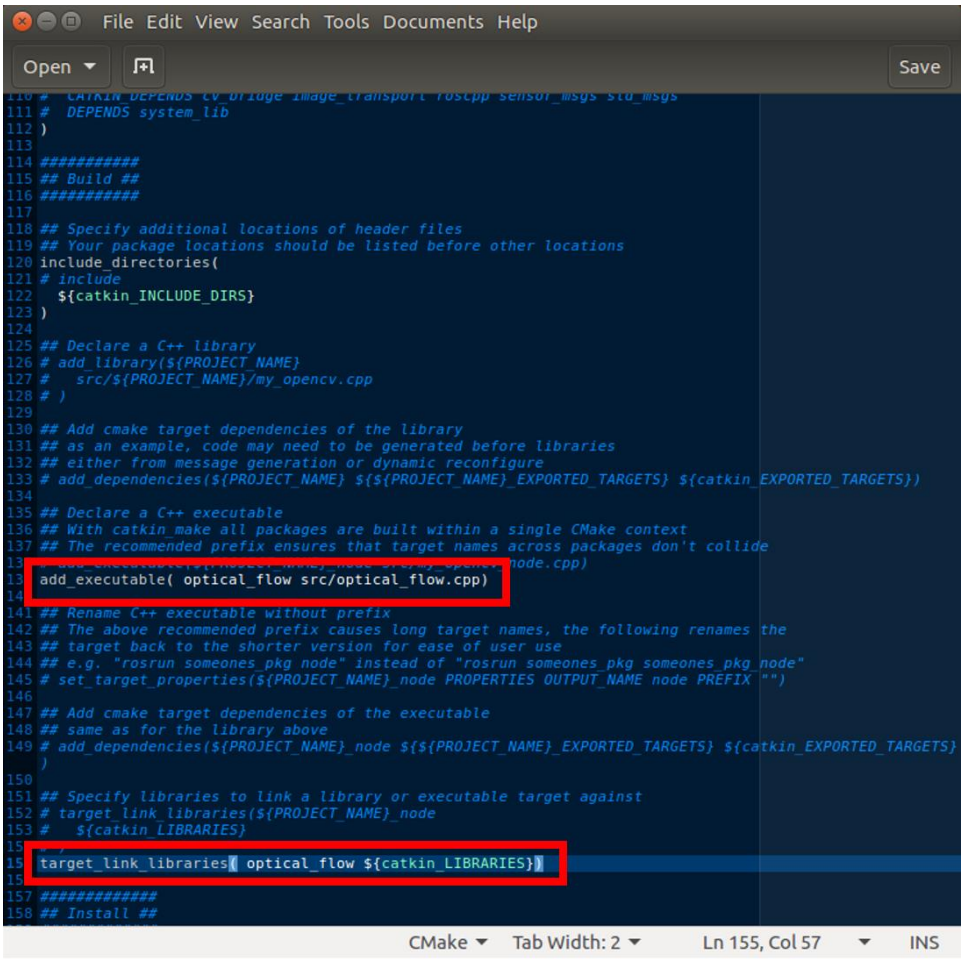


図１．CMakeList.txt の編集箇所

６．CMakeList.txt をつくる．
もし my_program/src の中に CMakeList.txt があったら消す
my_program/src の中で
$catkin_init_workspace
のコマンドを打つと CMakeList.txt ができる．
この中には，個々の PC のアドレスが書かれているため，他の PC で使う際は再度作り直す必要がある．

７．ビルドする．
my_program の階層で
$catkin_make

以上の流れで OpenCV を使ったプログラムのビルドを通すことができる．
下記にターミナルの実行結果を示す．

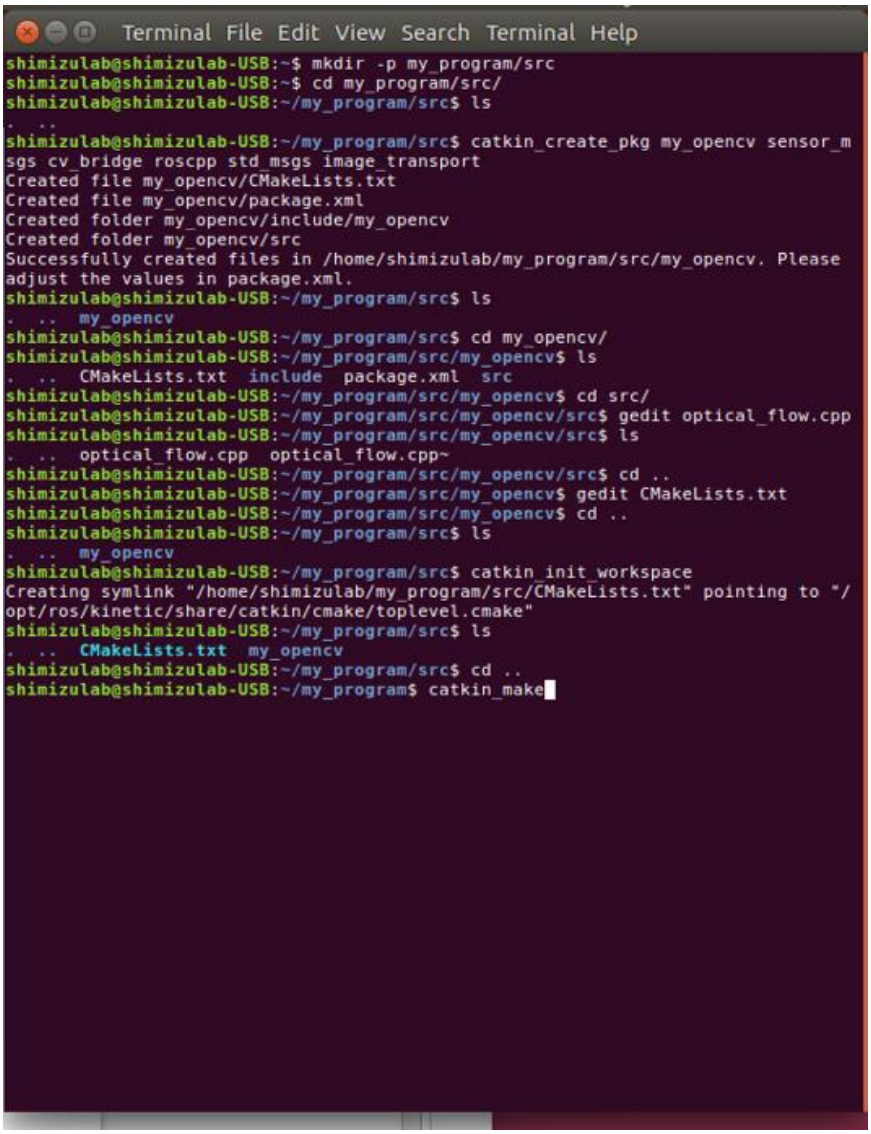


図２．ターミナルへの記述内容 その１

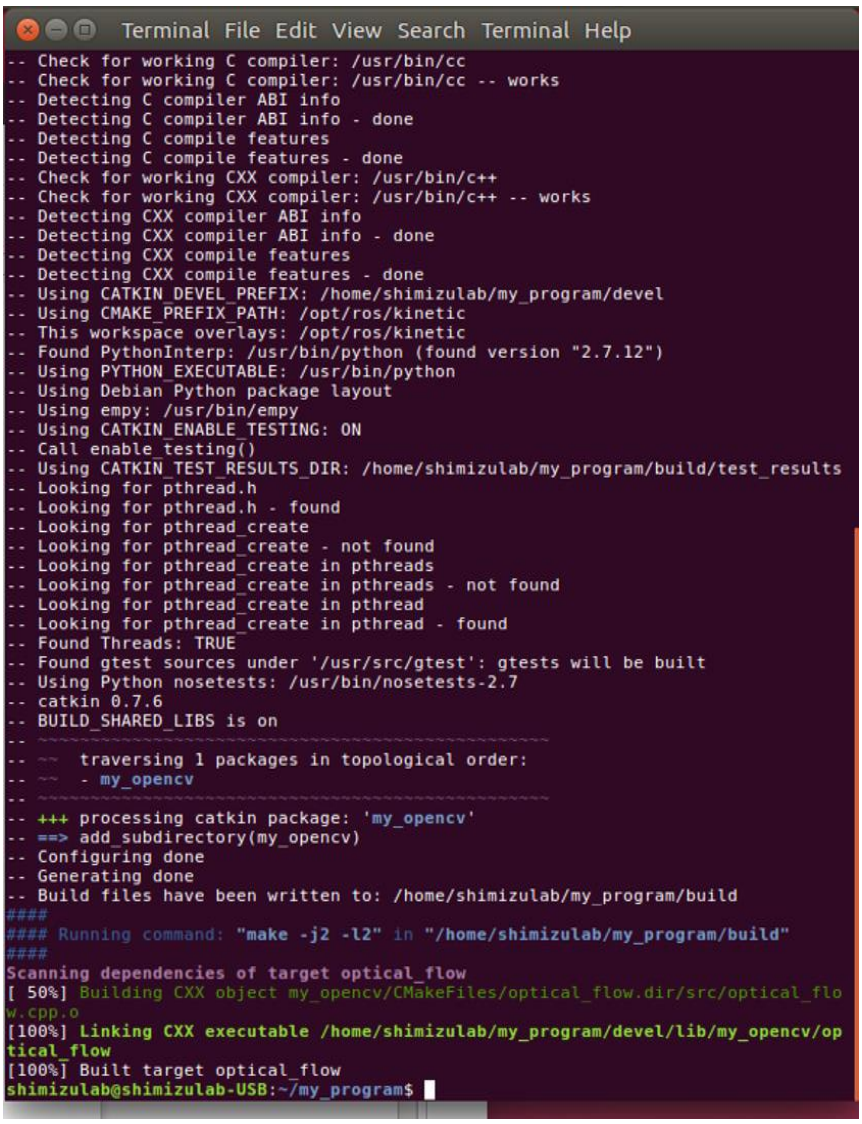


図３．ターミナルへの記述内容 その２

② USB カメラを使えるようにする方法

１．使用するプログラムがある src ファイルに移動
$cd my_program/src

２．github のサイトから ROS で USB カメラを使えるソフトをインストールする．
$git clone https://github.com/ros-drivers/usb_cam

３．ビルドする．
$cd ..
$catkin_make

以上の流れで USB カメラが使えるようになる．

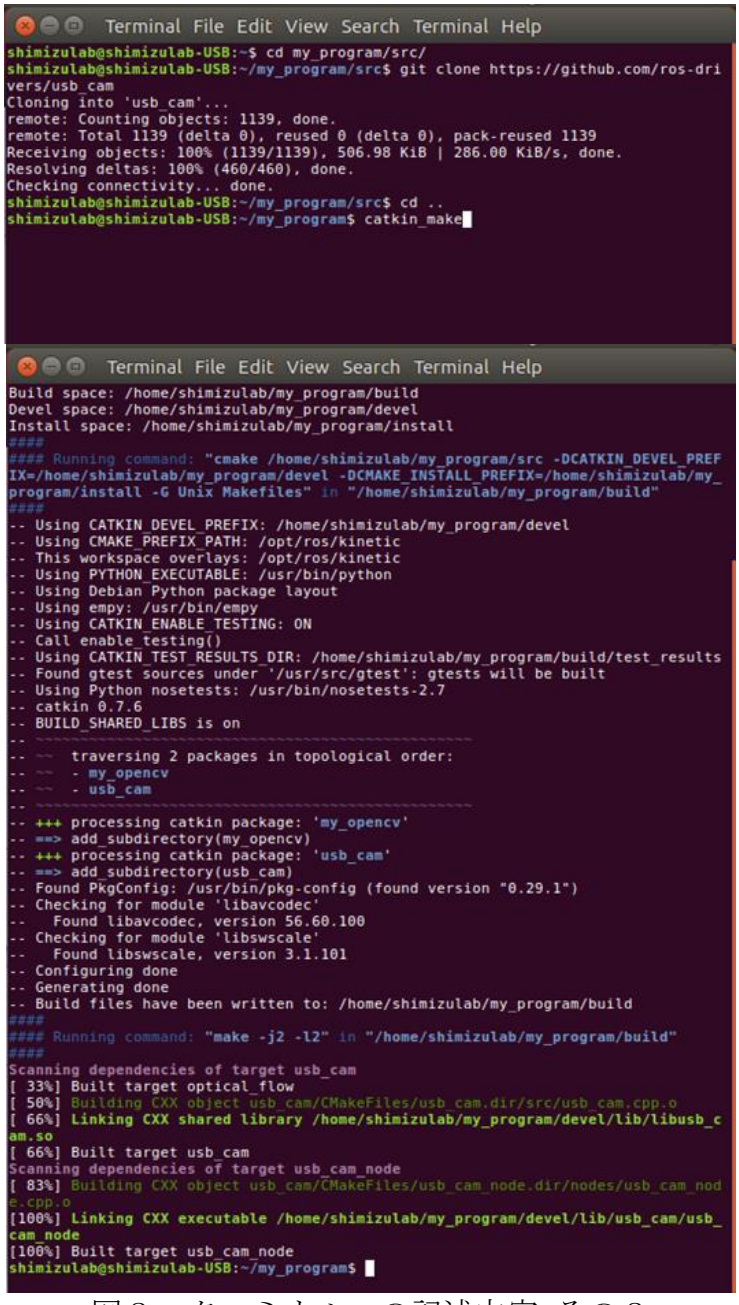


図３．ターミナルへの記述内容 その３

③ OpenCV と USB カメラ，ROS を用いたプログラムの実行方法

実行には３つのターミナルを用いる．

ターミナル１：ROS を動かす

$roscore

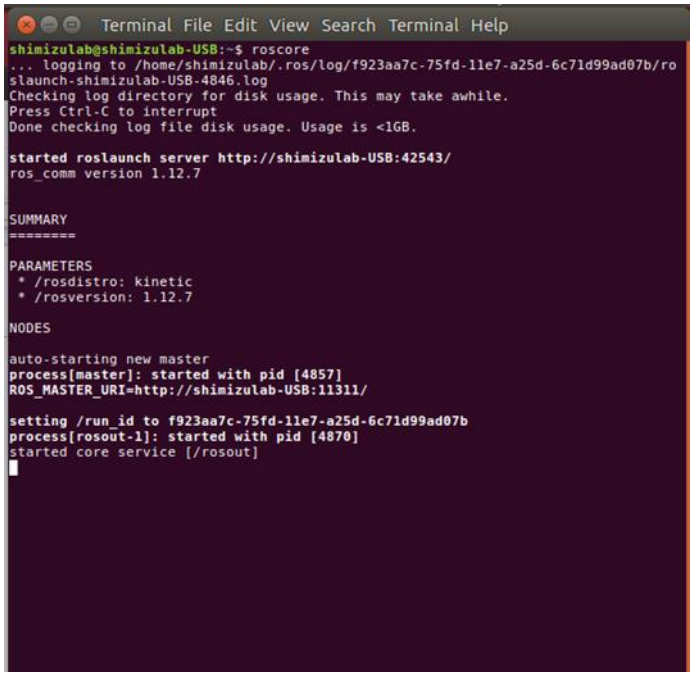


図４．ターミナル１への記述内容

ターミナル２：USB カメラを使えるようにする．

$cd my_program

$source devel/setup.bash

($rosparam set usb_cam/pixel_format yuyv)カメラによってはフォーマットを指定する必要がある．

$rosrun usb_cam usb_cam_node

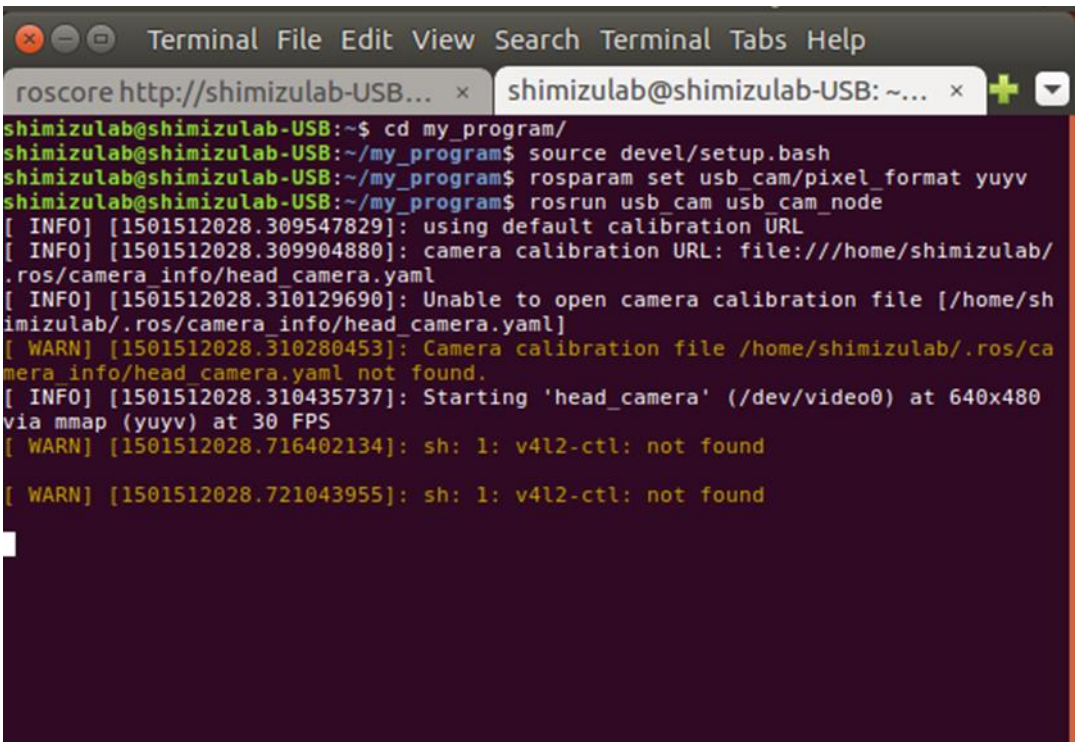


図５．ターミナル２への記述内容

ターミナル３：OpenCV で動かす

$cd my_program

$source devel/setup.bash

$rosrun my_opencv optical_flow

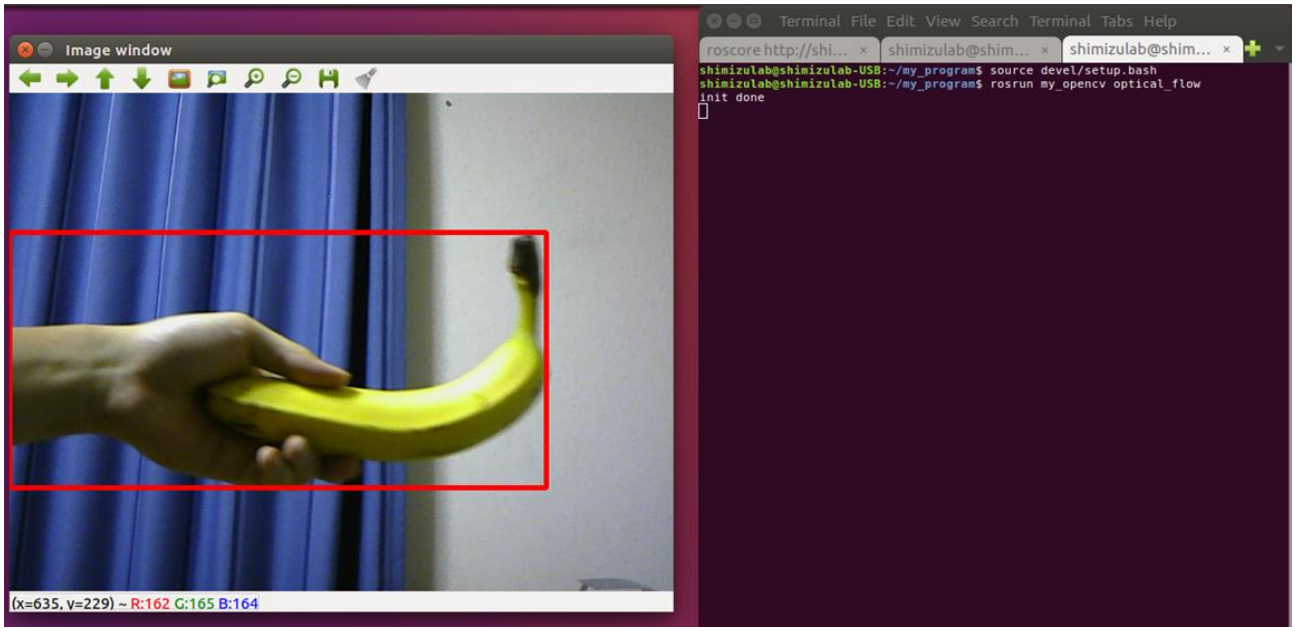
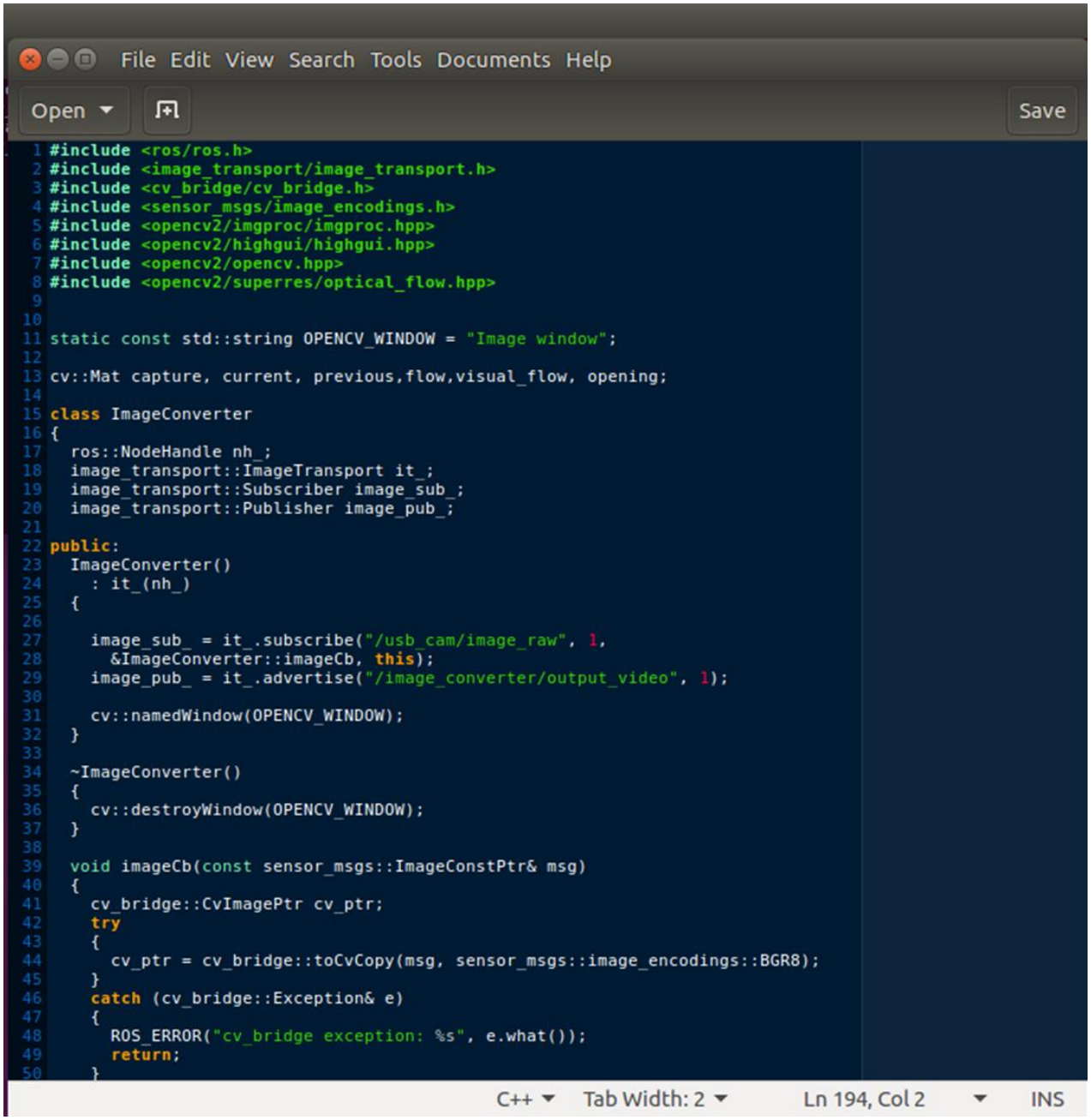
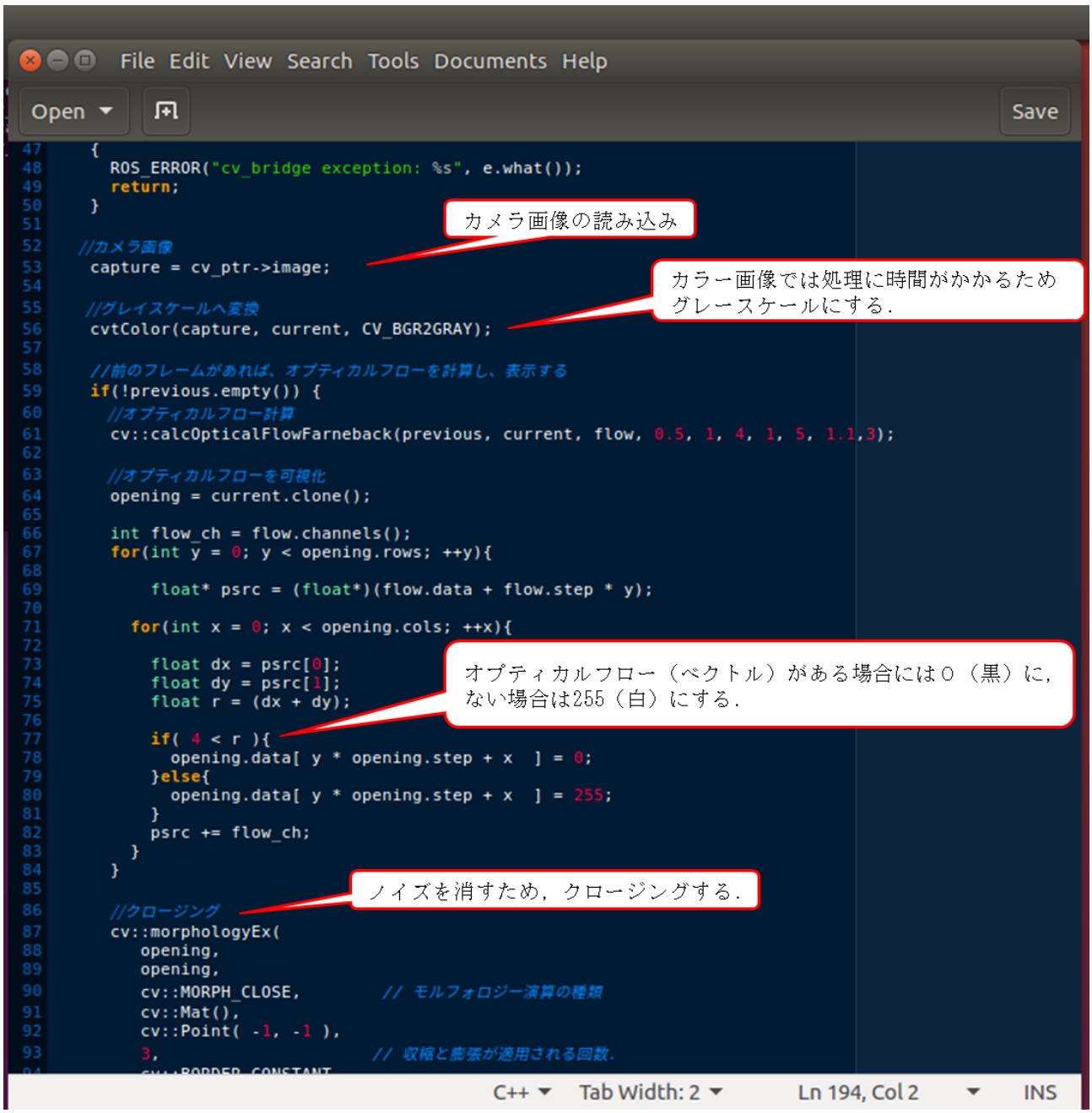


図６．ターミナル３への記述内容と実行結果

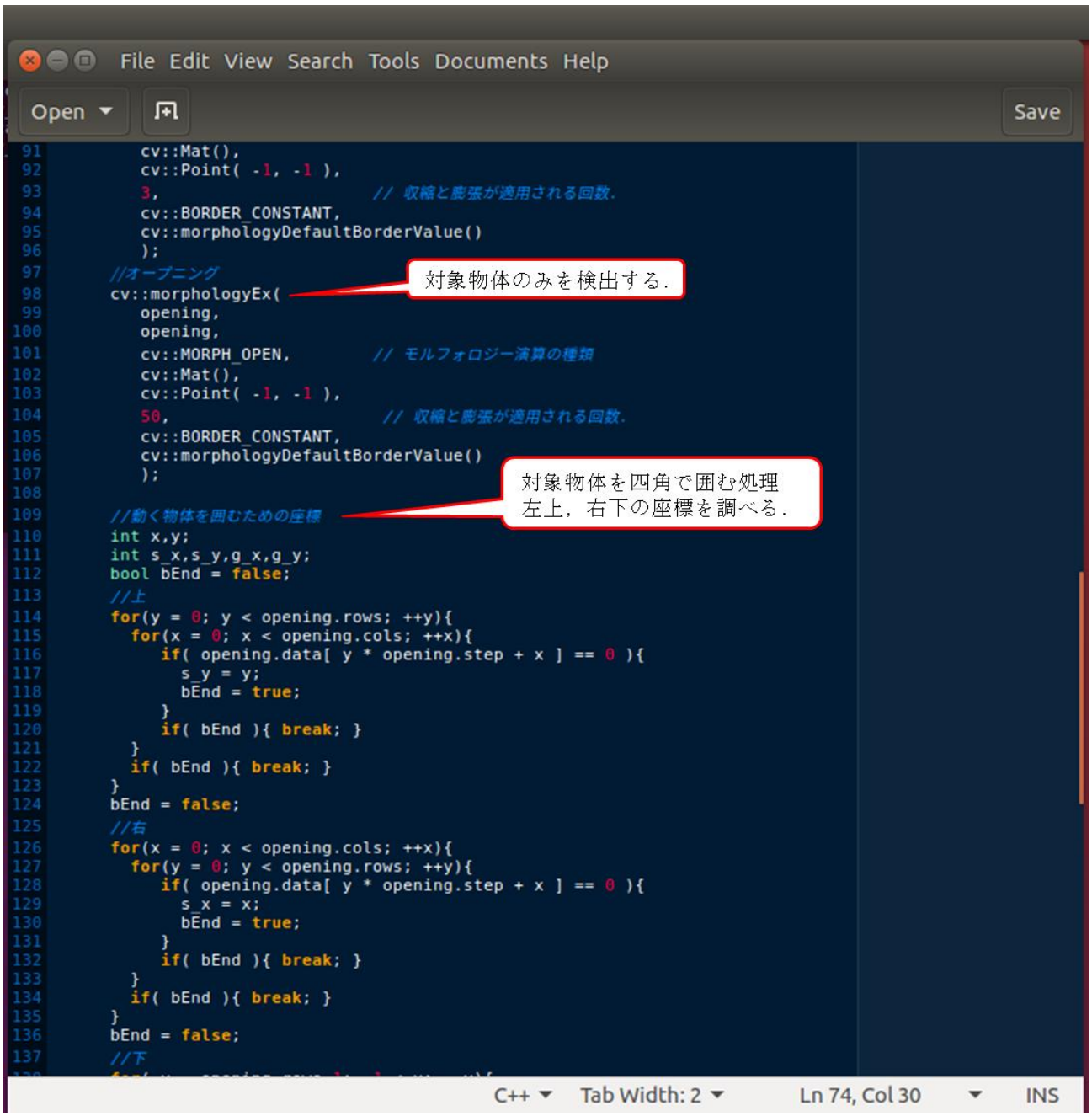
④ プログラムの概要

授業で扱った，カメラ画像を表示するプログラムを元に作成した.

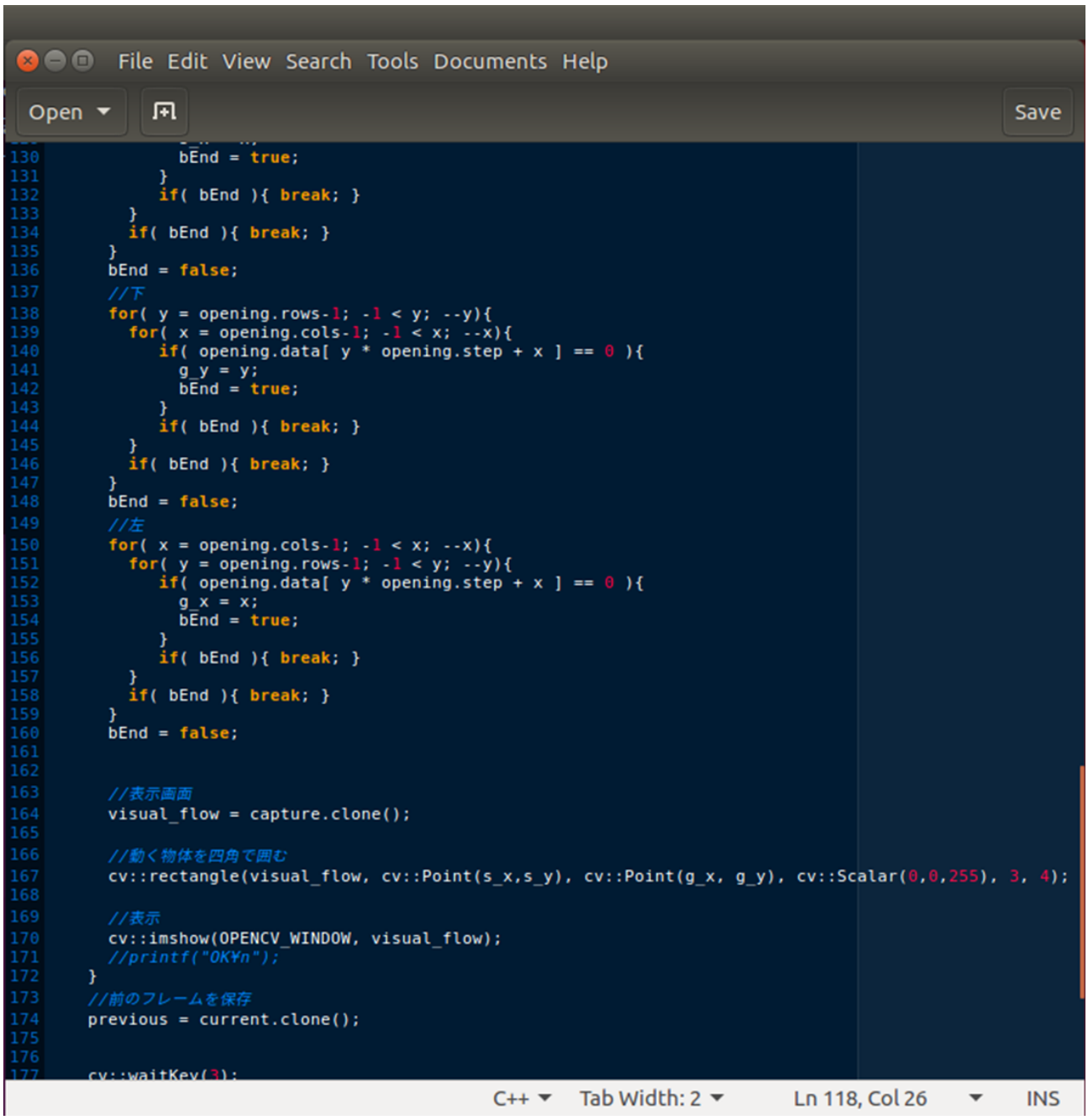





```
47    {
48      ROS_ERROR("cv_bridge exception: %s", e.what());
49      return;
50    }
51
52   //カメラ画像
53    capture = cv_ptr->image;
54
55   //グレイスケールへ変換
56    cvtColor(capture, current, CV_BGR2GRAY);
57
58    //前のフレームがあれば、オプティカルフローを計算し、表示する
59    if(!previous.empty()) {
60      //オプティカルフロー計算
61      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.
```




```
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      //オープニング
98      cv::morphologyEx(
99          opening,
100         opening,
101         cv::MORPH_OPEN,             // モルフォロジー演算の種類
102         cv::Mat(),
103         cv::Point( -1, -1 ),
104         50,                         // 収縮と膨張が適用される回数.
105         cv::BORDER_CONSTANT,
106         cv::morphologyDefaultBorderValue()
107         );
108
109     //動く物体を囲むための座標
110     int x,y;
111     int s_x,s_y,g_x,g_y;
112     bool bEnd = false;
113     //上
114     for(y = 0; y < opening.rows; ++y){
115       for(x = 0; x < opening.cols; ++x){
116         if( opening.data[ y * opening.step + x ] == 0 ){
117           s_y = y;
118           bEnd = true;
119         }
120         if( bEnd ){ break; }
121       }
122       if( bEnd ){ break; }
123     }
124     bEnd = false;
125     //右
126     for(x = 0; x < opening.cols; ++x){
127       for(y = 0; y < opening.rows; ++y){
128         if( opening.data[ y * opening.step + x ] == 0 ){
129           s_x = x;
130           bEnd = true;
131         }
132         if( bEnd ){ break; }
133       }
134       if( bEnd ){ break; }
135     }
136     bEnd = false;
137     //下
```



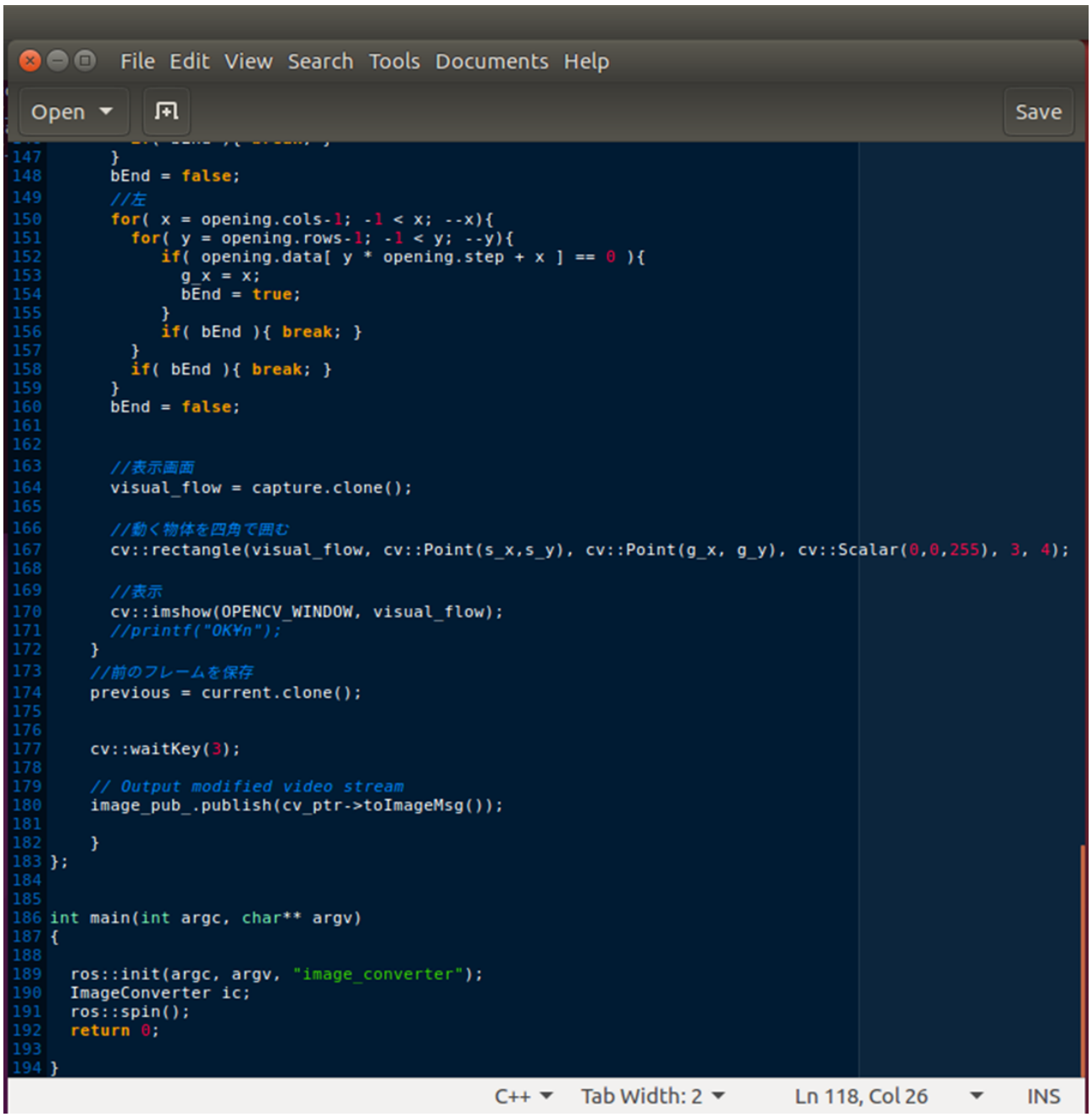

```
            bEnd = true;
          }
          if( bEnd ){ break; }
        }
        if( bEnd ){ break; }
      }
      bEnd = false;
      //下
      for( y = opening.rows-1; -1 < y; --y){
        for( x = opening.cols-1; -1 < x; --x){
          if( opening.data[ y * opening.step + x ] == 0 ){
            g_y = y;
            bEnd = true;
          }
          if( bEnd ){ break; }
        }
        if( bEnd ){ break; }
      }
      bEnd = false;
      //左
      for( x = opening.cols-1; -1 < x; --x){
        for( y = opening.rows-1; -1 < y; --y){
          if( opening.data[ y * opening.step + x ] == 0 ){
            g_x = x;
            bEnd = true;
          }
          if( bEnd ){ break; }
        }
        if( bEnd ){ break; }
      }
      bEnd = false;


      //表示画面
      visual_flow = capture.clone();

      //動く物体を四角で囲む
      cv::rectangle(visual_flow, cv::Point(s_x,s_y), cv::Point(g_x, g_y), cv::Scalar(0,0,255), 3, 4);

      //表示
      cv::imshow(OPENCV_WINDOW, visual_flow);
      //printf("OK¥n");
    }
    //前のフレームを保存
    previous = current.clone();


    cv::waitKey(3);
```


File Edit View Search Tools Documents Help
Open
Save
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 //前のフレームを保存
174 previous = current.clone();
175
176
177 cv::waitKey(3);
178
179 // Output modified video stream
180 image_pub_.publish(cv_ptr->toImageMsg());
181
182 }
183 };
184
185
186 int main(int argc, char** argv)
187 {
188
189 ros::init(argc, argv, "image_converter");
190 ImageConverter ic;
191 ros::spin();
192 return 0;
193
194 }
C++
Tab Width: 2
Ln 118, Col 26
INS