情報計測基礎 1

T11706M
七野　樺穂

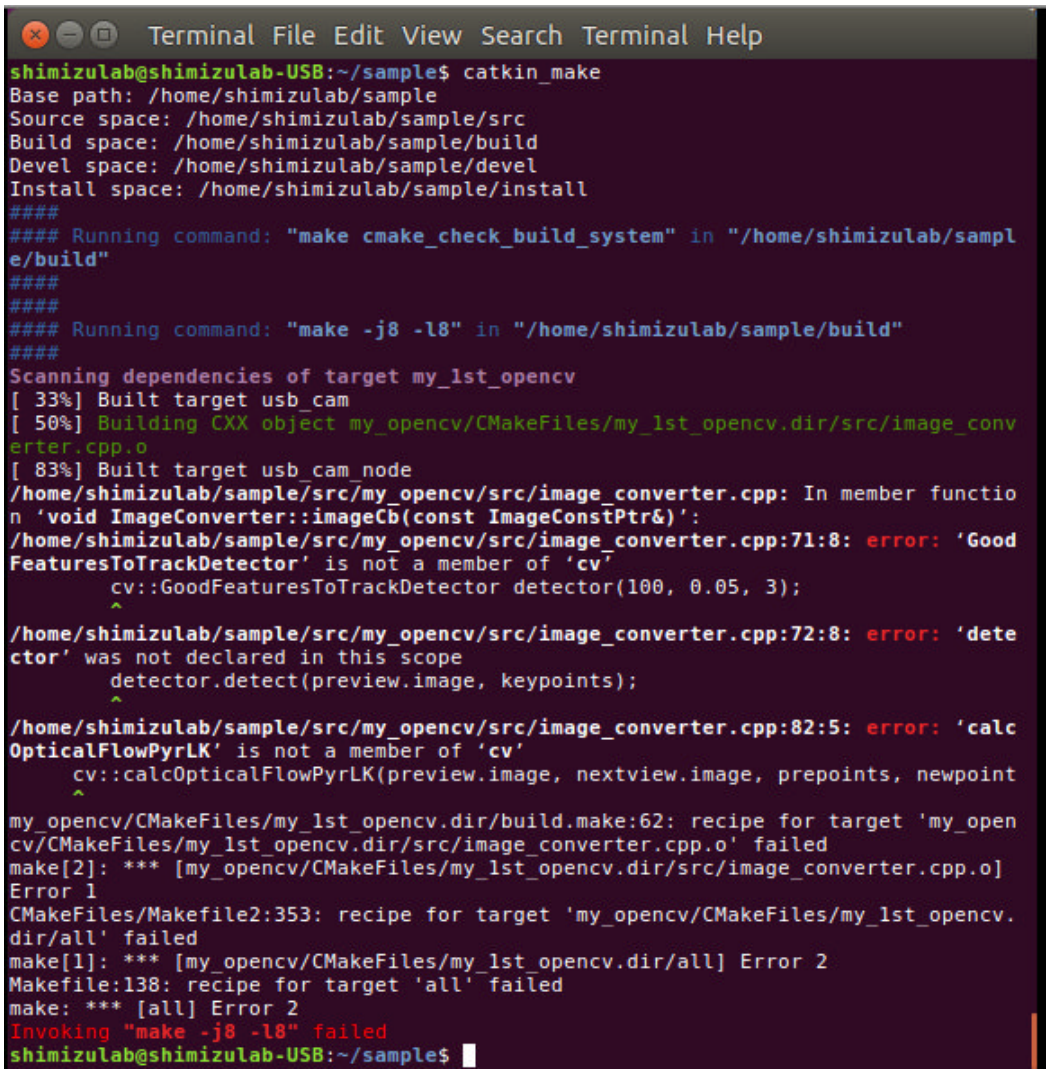
講義で作成したプログラムに加えた変更点と実装過程

講義中に実装した Imageconverter.cpp に、オプティカルフローを算出するプログラムを書き加える方法で、動体検出するプログラムの作成を目指した。下記の(1)、(2)を参考に、Imageconverter.cpp の 59～100 行目にオプティカルフローを算出するプログラムを下記のように書き加え、変更した変数名を書き換えた。

image_converter.cpp　　CMakeLists.txt

```
50    // Draw an example circle on the video stream
51    if (cv_ptr->image.rows > 60 && cv_ptr->image.cols > 60)
52      cv::circle(cv_ptr->image, cv::Point(50, 50), 10, CV_RGB(255,0,0));
53
54      //cv::Mat gray;
55      //cv_bridge::CvImage  gray;
56      //cv::cvtColor(cv_ptr->image, gray.image, CV_BGR2GRAY);
57
58  ////////////////////////////////////////////////////
59      cv_bridge::CvImage preview, nextview;
60
61      cv::cvtColor(cv_ptr->image, nextview.image, CV_BGR2GRAY);
62
63      //2color white/black
64      //cv::threshold(gray.image, gray.image, 127, 255, cv::THRESH_BINARY);
65
66      if(&preview.image != 0){
67
68        std::vector<cv::KeyPoint>  keypoints;
69        std::vector<cv::Point2f>   prepoints;
70
71        cv::GoodFeaturesToTrackDetector detector(100, 0.05, 3);
72        detector.detect(preview.image, keypoints);
73
74        for(std::vector<cv::KeyPoint>::iterator itk = keypoints.begin(); itk != keypoints.end(); ++itk){
75          prepoints.push_back(itk->pt);
76        }
77
78        std::vector<cv::Point2f>  newpoints;
79        std::vector<unsigned char>  status;
80        std::vector<float>   errors;
81
82        cv::calcOpticalFlowPyrLK(preview.image, nextview.image, prepoints, newpoints, status, errors,
  cv::Size(21,21), 3, cvTermCriteria(CV_TERMCRIT_ITER | CV_TERMCRIT_EPS, 20, 0.05), 0);
83        }
84
85      cv::cvtColor(nextview.image, preview.image, CV_BGR2GRAY);
86  ////////////////////////////////////////////////////
87
88    // Update GUI Window
89    //cv::imshow(OPENCV_WINDOW, cv_ptr->image);
90    cv::imshow(OPENCV_WINDOW, nextview.image);
91    cv::waitKey(3);
92
93    // Output modified video stream
94    //image_pub_.publish(cv_ptr->toImageMsg());
95
96     //When you with publish color
97     //image_pub_.publish(gray.toImageMsg());
98
99     //When you with publish mono
100    image_pub_.publish(cv_bridge::CvImage(std_msgs::Header(),"mono8",nextview.image).toImageMsg());
101
102
103
104   }
105 };
```

上記のように書き換えた後、catkin_make を実行してみると、



上記のようなエラーがでる。1、3 つ目のエラーは、「cv のメンバーにそのような名前はいない」という内容で、2 つ目のエラーは、1 つ目のエラーが原因で変数が宣言できなかったと考えられる。

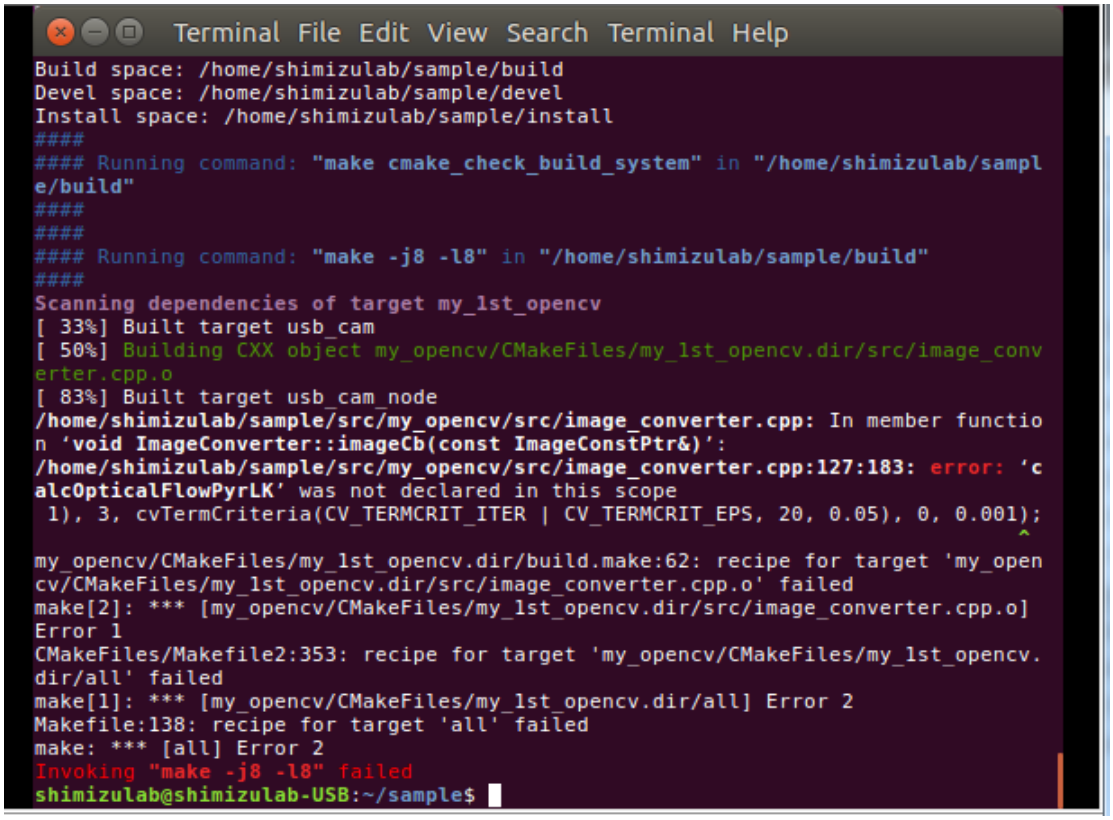
エラー内容・講義内で Opencv の実装時に利用した(4)のサイトを見直し、GoodFeatureToTrack や、calcOpticalFlowPyrLK を使うには、他に必要なパッケージがあると考えた。調べてみたが、Windows 向けのものや、Python で書かれたものしか見つからず、動体検出するプログラムの作成には至らなかった。
そこで、GoodFeatureToTrack や、calcOpticalFlowPyrLK の実装例(6),(7)を参考にプログラムを下記のように書き換えた。

```
static const std::string OPENCV_WINDOW = "Image window";

/// Global variables

int maxCorners = 23;
int maxTrackbar = 100;

class ImageConverter
{
  ros::NodeHandle nh_;
  image_transport::ImageTransport it_;
  image_transport::Subscriber image_sub_;
  image_transport::Publisher image_pub_;

public:
  ImageConverter()
    : it_(nh_)
```

```
  void goodFeaturesToTrack_Demo( int, void* ,cv_bridge::CvImage  gray, std::vector<cv::Point2f> corners)  ///[1]
{
  if( maxCorners < 1 ) {
    maxCorners = 1;
  }

  /// Parameters for Shi-Tomasi algorithm
  //std::vector<cv::Point2i> corners;       ///[2]
  double qualityLevel = 0.01;
  double minDistance = 10;
  int blockSize = 3;
  bool useHarrisDetector = false;
  double k = 0.04;

  /// Copy the source image


  /// Apply corner detection
  goodFeaturesToTrack( gray.image,
               corners,
               maxCorners,
               qualityLevel,
               minDistance,
               cv::Mat(),                    ///[3]
               blockSize,
               useHarrisDetector,
               k );


  /// Draw corners detected
 // cout<<"** Number of corners detected: "<<corners.size()<<endl;
 // int r = 4;
  //for( size_t i = 0; i < corners.size(); i++ )
```

```
90
91      // Draw an example circle on the video stream
92      if (cv_ptr->image.rows > 60 && cv_ptr->image.cols > 60)
93        cv::circle(cv_ptr->image, cv::Point(50, 50), 10, CV_RGB(255,0,0));
94
95        //cv::Mat gray;
96        //cv_bridge::CvImage  gray;
97        //cv::cvtColor(cv_ptr->image, gray.image, CV_BGR2GRAY);
98
99        cv_bridge::CvImage preview, nextview;
100       std::vector<cv::Point2f>  points;
101       std::vector<cv::Point2f>  newpoints;
102
103       cv::cvtColor(cv_ptr->image, nextview.image, CV_BGR2GRAY);
104
105      // goodFeaturesToTrack_Demo( 0, 0, nextview );
106
107       //2color white/black
108       //cv::threshold(gray.image, gray.image, 127, 255, cv::THRESH_BINARY);
109
110 ////////////////////////////////////////////////////
111       if(&preview.image != 0){
112
113         //std::vector<cv::KeyPoint>  keypoints;
114        // std::vector<cv::Point2f>   prepoints;
115
116         //cv::GoodFeaturesToTrackDetector detector(100, 0.05, 3);
117         //detector.detect(preview.image, keypoints);
118
119         //for(std::vector<cv::KeyPoint>::iterator itk = keypoints.begin(); itk != keypoints.end(); ++itk){
120           //prepoints.push_back(itk->pt);
121         //}
122
123         std::vector<unsigned char>  status;
124         std::vector<float>   errors;
125
126
127         calcOpticalFlowPyrLK(preview.image, nextview.image, points, newpoints, status, errors, cv::Size(21,21), 3,
    cvTermCriteria(CV_TERMCRIT_ITER | CV_TERMCRIT_EPS, 20, 0.05), 0, 0.001);
128
129
130       }
131
132       cv::cvtColor(nextview.image, preview.image, CV_BGR2GRAY);
133       goodFeaturesToTrack_Demo( 0, 0, preview, points );
134
135
136 ////////////////////////////////////////////////////
137
138     // Update GUI Window
139     //cv::imshow(OPENCV_WINDOW, cv_ptr->image);
140     cv::imshow(OPENCV_WINDOW, nextview.image);
141     cv::waitKey(3);
142
143     // Output modified video stream
144     //image_pub_.publish(cv_ptr->toImageMsg());
```

上記のように書き換えた後、catkin_make を実行してみると、



上記のようなエラーが出た。調べてもわからなかったため、最終的にオプティカルフローの算出を実装することはできなかったが、オプティカルフローの算出に関わる部分をコメントアウトし、GoodFeatureToTrack の動作確認を行ったところ、特徴抽出は問題なくできていた。calcOpticalFlowPyrLK を動かせるようになれば、オプティカルフローを算出し、動体検出が出来るようになると考えられる。

【参考にしたサイト】

(1)C++版 OpenCV のオプティカルフローで物体追跡

http://opencv.blog.jp/cpp/opticalflow_lucaskanade

検索ワード：オプティカルフロー　c++

(2)OpenCV 備忘録：OpenCV で OpticalFlow を試してみた

https://iwaki2009.blogspot.jp/2012/12/opencvopticalflow.html

検索：(1)のリンク「OpenCV で OpticalFlow を試してみた」から

(3)モーション解析と物体追跡—opencv2.2

http://opencv.jp/opencv-2svn/cpp/motion_analysis_and_object_tracking.html#cv-calcopticalflowpyrlk

検索：(1)のリンク「cv::calcOpticalFlowPyrLK 日本語ドキュメント」から

(4)cv_bridge/Tutorials/UsingCvBridgeToConvertBetweenROSImagesAndOpenCVImages – ROS Wiki
http://wiki.ros.org/cv_bridge/Tutorials/UsingCvBridgeToConvertBetweenROSImagesAndOpenCVImages
検索ワード：ros opencv

(5)opencv_apps – ROS Wiki
http://wiki.ros.org/opencv_apps
検索ワード：ros opencv opticalflow

(6) opencv/ goodFeaturesToTrack_Demo.cpp at master ・opencv/opencv・GiyHub
https://github.com/opencv/opencv/blob/master/samples/cpp/tutorial_code/TrackingMotion/goodFeaturesToTrack_Demo.cpp
検索：(5)の 5.Motion Analysis Nodes の 5.1goodfeature_track にあるリンク goodFeaturesToTrack_Demo.cpp から

(7)opencv/lkdemo.cpp at 2.4 ・opencv/opencv・GiyHub
https://github.com/opencv/opencv/blob/2.4/samples/cpp/lkdemo.cpp
検索：(5)の 5.Motion Analysis Nodes の 5.4lk_flow にあるリンク lkdemo.cpp から