OpenCV 5차시

ㅇ 클래스 설계와 구현

다음과 같이 사용될 수 있는 클래스 MyOpenCV를 설계하고 구현하시오.
중요 구현사항은 다음과 같다.
1. .cpp와 .hpp를 구분해서 구현한다.
2. 이미지 이름을 string 인수로 입력받는 생성자를 가지고 있다.
3. 크기가 조절된 이미지를 반환하는 함수 getResizedImg( )는 인수로써 크기조절 배율을 입력받는다.

```
1   #include "MyOpenCV.hpp"
2
3   int main(int argc, char* argv[])
4   {
5       if (argc < 2)
6       {
7           cout << "Image file name is missing" << endl;
8       }
9
10      MyOpenCV* m = new MyOpenCV(argv[1]);
11
12      imshow("Result", m->getResizedImg(0.2));
13
14      waitKey(0);
15
16  }
```

ㅇ 구현 예

위의 조건을 만족시키는 클래스 MyOpenCV의 구현 예 (MyOpenCV.hpp, MyOpenCV.cpp)는 다음과 같다.

```cpp
#include <opencv2/opencv.hpp>
#include <iostream>

using namespace cv;
using namespace std;

class MyOpenCV
{
private:
    Mat myOrgImg;
    Mat myProcessedImg;

public:
    MyOpenCV(Mat img);
    MyOpenCV(string fname);
    ~MyOpenCV();
    Mat& getResizedImg(float _s);
};
```

```cpp
#include "MyOpenCV.hpp"

// Constructor
MyOpenCV::MyOpenCV(Mat img)
{
    img.copyTo(myOrgImg);
}

MyOpenCV::MyOpenCV(string fname)
{
    myOrgImg = imread(fname, IMREAD_UNCHANGED);
    if (myOrgImg.empty())
    {
        cout << "Unambe to read image: " << fname << endl;
        return;
    }
}

// Destrunctor
MyOpenCV::~MyOpenCV()
{

}

Mat& MyOpenCV::getResizedImg(float _s)
{
    resize(myOrgImg, myProcessedImg, Size(), _s, _s);
    return myProcessedImg;
}
```

ㅇ 상속

클래스 MyOpenCV를 상속하여 새로운 클래스 MyCV를 만들 때, 아래 main( ) 함수에서 처럼 사용가능하도록 하시오.

함수 getEdgeDetectedImage( )는 이미지 크기를 0.5 비율로 축소시킨 후, canny edge detect한 결과를 반환한다.

```
#include "MyCV.hpp"

int main(int argc, char* argv[])
{
    if (argc < 2)
    {
        cout << "Image file name is missing" << endl;
    }

    MyCV* m = new MyCV(argv[1]);

    imshow("Canny", m->getEdgedDetectedImage());
    waitKey(0);

}
```

구현 예

```
#include "MyOpenCV.hpp"

class MyCV : MyOpenCV
{
private:
    MyOpenCV * p;
public:
    MyCV(string fname);
    MyCV();
    Mat& getEdgedDetectedImage();
};
```

```
#include "MyCV.hpp"

MyCV::MyCV(string fname)
{
    p = new MyOpenCV(fname);
}

MyCV::MyCV()
{

}

Mat& MyCV::getEdgedDetectedImage()
{
    Mat& m = p->getResizedImg(0.5);
    cvtColor(m, m, COLOR_BGR2GRAY);
    Canny(m, m, 100, 200);
    return m;
}
```

실습: OpenCV API를 이용하여 다음을 구현해 보자. 단, 모든 구현은 Class 기반으로 구현하여, 다른 프로그램에서 재사용할 수 있는 형태로 구현할 것.
1. 블랙박스 동영상의 이미지 프레임의 가로 길이와 세로 길이를 알아보자.
2. 블랙박스 동영상의 정중앙 부분에 빨간색 사각형 (크기는 가로,세로 30)이 표시되도록 해 보자. * OpenCV API를 검색하여 구현
3. 블랙박스 동영상을 Gray로 표시하되, 정중앙 사각형은 빨간색으로 표시하시오.
4. 블랙박스 동영상에서 **번호판이 있을 확률이 높은 곳을 찾아서, 빨간색 사각형으로 표시**하시오. 단, 알고리즘은 자율적으로 구현하되, 그러한 알고리즘을 만든 이유를 서술.