OPENCV_1일차_최범수

19기 예비단원 최범수

1. HW_001

가우시안 필터를 적용 한 후 색상과 외부 배경의 경계값이 흐려지는 현상을 보였다. 이를 설명하기 전에 먼저 코드에 대해서 설명하겠다. 본 코드는 ROS 패키지를 제작해 작성되었다. image_filter.hpp, image.filter.cpp로 구성되어 있다.

```
#ifndef IMAGE_FILTER_HPP
#define IMAGE_FILTER_HPP

#include <opencv2/opencv.hpp>
#include <string>

class ImageFilter {
public:
    ImageFilter(const std::string& image_path);
    void process();
    void displayImages();
    void saveImages(const std::string& original_path, const s

private:
    cv::Mat img;
    cv::Mat yellow_binary;
};

#endif // IMAGE_FILTER_HPP
```

ImageFilter와 관련된 클래스를 만들고 public 영역에 process, displayImages, saveImages와 같은 함수들을 선언했다. 각각의 역할에 대해서 설명하자면 ImageFilter는 생성자로 경로를 받아 이미지를 불러온다. process 함수는 이미지를 처리하는 함수이다. 노란색 영역을 검출해 바이너리 작업을 한다. displayImages는 처리 결과를 이미지에 적용해 출력한다. SaveImages는 이미지를 적용해 경로에 저장하는 역할이다.

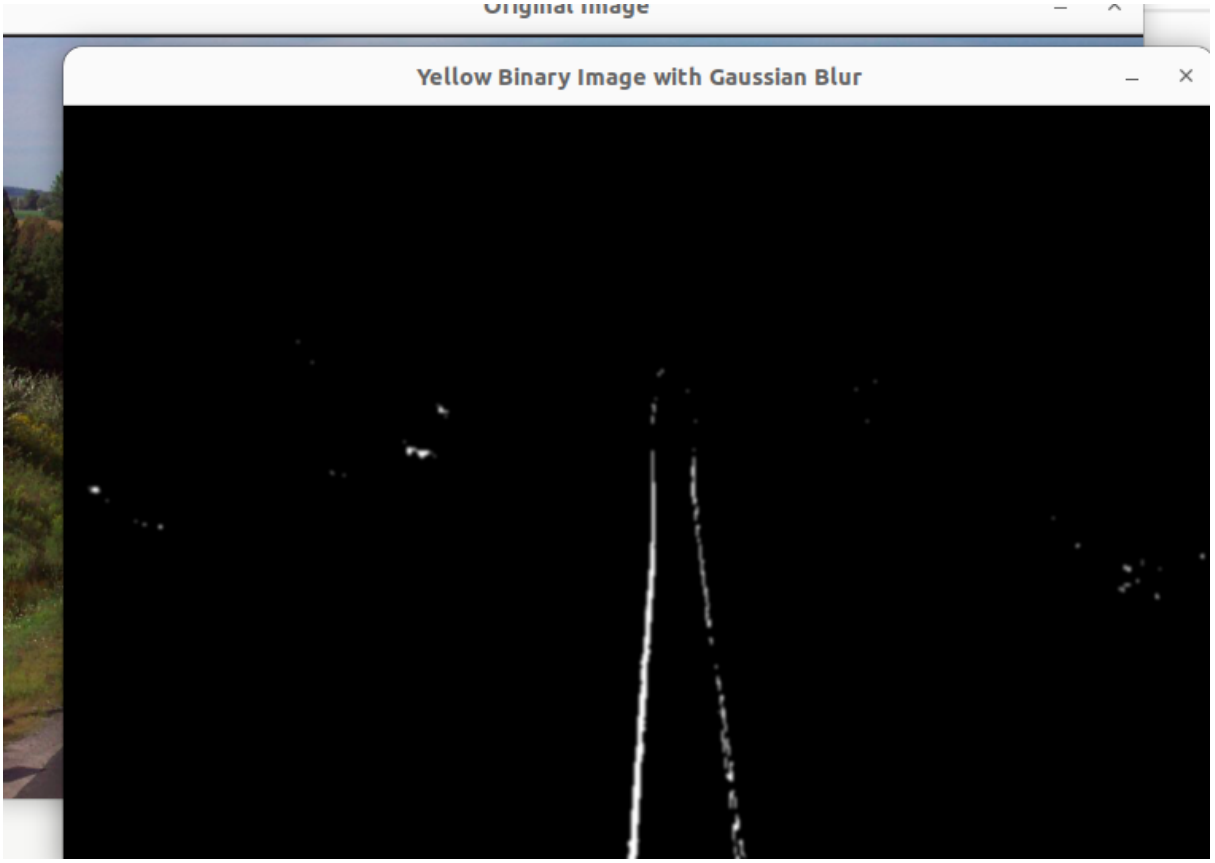
Private 멤버에는 img, yellow_binary가 있다. OpenCV에서 쓰이는 타입이며, 저장, 처리에 사용된다.

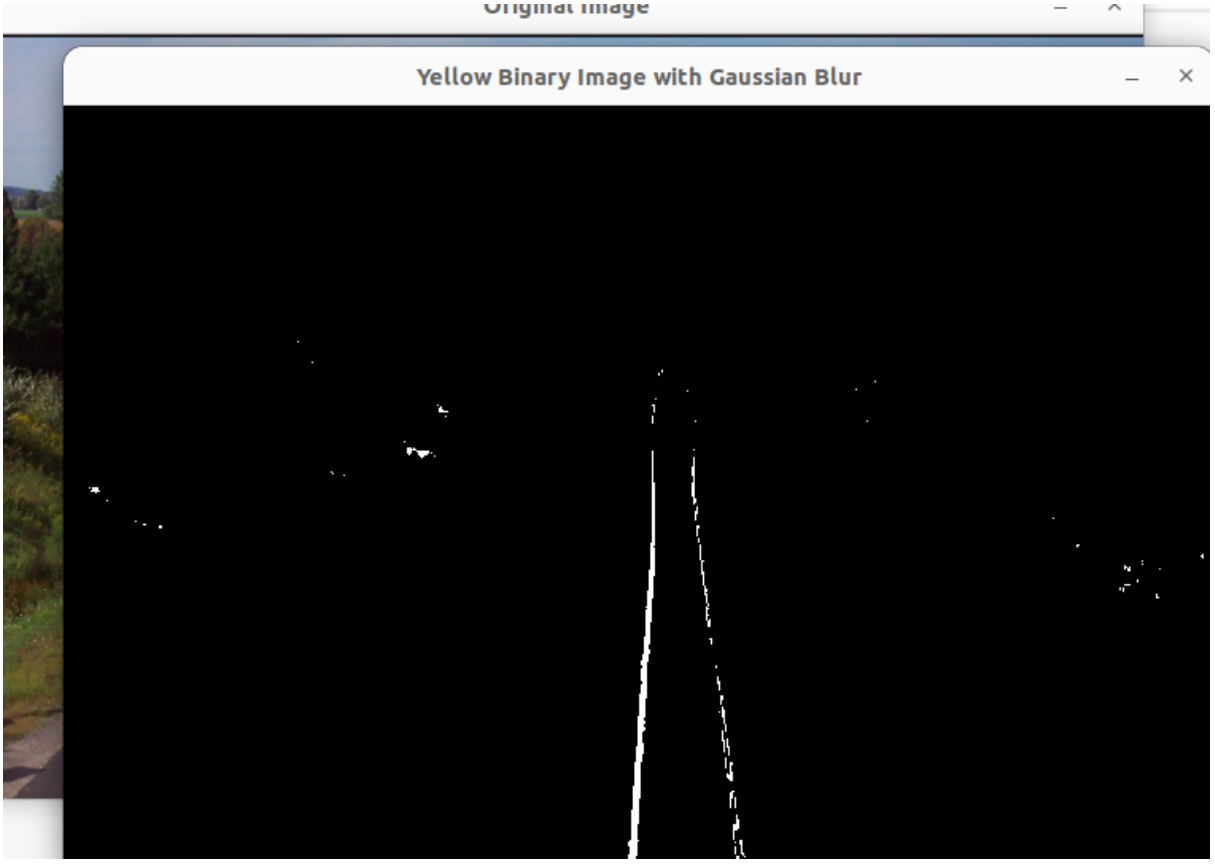
```
#include "image_package/image_filter.hpp"
#include <iostream>

ImageFilter::ImageFilter(const std::string& image_path) {
    img = cv::imread(image_path, cv::IMREAD_COLOR);
    if (img.empty()) {
        std::cerr << "이미지를 불러올 수 없습니다! 경로를 확인하세요:
    }
}

void ImageFilter::process() {
    if (img.empty()) return;

    // HSV 색상
    cv::Mat hsv_image;
    cv::cvtColor(img, hsv_image, cv::COLOR_BGR2HSV);

    // 노란색 범위 설정
    cv::Scalar lower_yellow(20, 180, 100), upper_yellow(30, 2

    // 노란색 마스크 생성
    cv::inRange(hsv_image, lower_yellow, upper_yellow, yellow_

    // 가우시안 블러 적용
    cv::GaussianBlur(yellow_binary, yellow_binary, cv::Size(5
}

void ImageFilter::displayImages() {

    // 원본과 노란색 binary 출력
    cv::imshow("Original Image", img);
    cv::imshow("Yellow Binary Image with Gaussian Blur", yell
    cv::waitKey(0);
}
```

```
void ImageFilter::saveImages(const std::string& original_path

    cv::imwrite(original_path, img);
    cv::imwrite(binary_path + "_yellow_blurred.png", yellow_b
}
```

ImageFilter는 생성자로서 이미지 파일을 받는다. process 함수는 HSV 색상으로 변환해 새로운 변수에 저장하고 지정한 노란색 값을 통해서 필터링 작업을 한 뒤 가우시안 블러를 적용한다

나머지는 단순히 저장하거나 화면에 띄우는 기능이다. 이제 실행결과를 비교하겠다.





다음 사진은 가우시안 블러를 적용한 사진과 밑은 적용하지 않은 사진이다. 오른쪽 길을 보거나 점을 보면 확연한 차이가 보인다. 적용한 사진에서 흰색 점이 사라진 부분도 있고 뿌옇게 뭉개진 부분도 있다. 가우시안 블러를 적용하는 이유는 위의 사진처럼 이미지를 부드럽게 만들 수 있기 때문이다. 밑의 사진은 이진화 작업이 처리되었는데 점과 같은 노이즈들이 보인다. 여기서 가우시안 블러를 통해 작은 노이즈들을 감소시킬 수 있다. 두번째로는 경계를 부드럽게 처리한다는 점이다. 경계가 구분이 확실하면 후의 객체 탐지 부분에서 노이즈로 인한 오류가 발생할 수 있다. 가우시안 블러를 통해 경계를 뭉개놓는다면 객체 탐지 부분에서도 더 정확한 결과를 얻을 수 있다.

위의 특징처럼 가우시안 블러는 후속 단계를 위한 초석과 같다. 가우시안 블러를 적용해 이미지를 부드럽게 처리한다면 후의 라벨링 혹은 객체탐지, 호프라인과 같은 직선을 찾는 과정에서 필요한 정보만을 노이즈 없이 출력할 수 있기에 가우시안 블러를 적용해야 한다.

2. HW_002

가우시안 필터외의 다른 필터로는 미디언 필터, 평균 필터, 샤프닝 필터가 있다. 이 필터들은 전부 openCV에서 지원하고 있는 필터이다.
첫번째, 미디언 필터이다. 미디언 필터는 커널 내의 픽셀 값을 정렬한 뒤 중앙값을 사용해서

중앙 필터 값을 대체하는 방식으로 노이즈를 줄인다. 사용 목적으로는 소금과 후추 노이즈와 같은 극단적인 픽셀 값이 있는 노이즈 제거에 유용하므로 사용한다.

```
cv::medianBlur(input_image, output_image, kernel_size);
```

미디언 필터는 가우시안 블러와 달리 이미지의 경계를 유지하는 강점이 있다. 중앙값을 평균으로 대체 하다보니 점과 같은 작은 노이즈를 수월하게 제거할 수 있다. 여기서 kernel_size는 홀수여야 한다. 그 이유는 커널의 크기를 사이즈의 곱으로 구성되는데 중앙을 기준으로 대칭 구조여야 중앙에 하나의 픽셀을 둘 수 있으므로 커널을 홀수로 둔다. 미디언 필터 역시 적절한 값을 가지고 적용해야 한다. 과하게 적용한다면 이미지의 세부 정보가 함께 손실되므로 적절한 값을 찾아야 한다.

평균 필터는 각 필셀의 값을 주변 픽셀의 평균값으로 계속 바꾸는 필터이다. 다시 말하면 커널 내 모든 픽셀 값을 더하고 평균을 구해 커널의 중앙 픽셀 값을 대체하는것이다.

```
cv::blur(input_image, output_image, cv::Size(kernel_width, ke
```

kernel_width와 kernel_height는 필터의 커널 크기이다. 커널 크기가 클수록 더 흐려진다는 특징이 있다. 사용하는 이유로는 이미지에 가우시안 필터와 유사하게 뭉개지는 효과를 위해 사용한다. 단순히 평균을 사용하므로 cost가 낮다. 하지만 가우시안 블러에 비해서도 엣지를 더 많이 뭉개기 때문에 더 흐려진다는 특징이 있다. 따라서 평균 필터는 커널 크기에 따라 노이즈를 더 많이 제거할 수 있다는 장점이 있지만 이미지의 자세함 또는 엣지가 크게 달라질 수 있다.

마지막으로 샤프닝 필터이다. 샤프닝 필터는 이미지의 고주파 성분을 강조해 경계선과 디테일을 더 뚜렷하게 만드는 필터이다. 샤프닝 필터는 커널을 사용해 중앙 픽셀의 값을 더 크게 하거나 강조하는 방식으로 지금까지 쓰인 필터와는 반대의 효과를 얻기 위해서 사용한다. 샤프닝 필터는 일반적으로 3*3 크기의 커널을 사용한다. 커널을 작은 부분에 적용해 특정 부분을 강조하기 위해서 사용하는 것이 일반적이다.

```
cv::Mat kernel = (cv::Mat_<float>(3, 3) <<
0, -1, 0,
-1, 5, -1,
0, -1, 0);
cv::filter2D(input_image, output_image, -1, kernel);
```

이런 식으로 중앙값을 대조적으로 강조하는 역할을 한다. 중앙에 양수값을 주고 주변은 음수값을 준 것을 볼 수 있는데 이를 통해 경계선을 강조한다.

마지막으로 각 필터를 사용했을 때의 기대값을 설명하겠다.

1. 미디언 필터의 기대값은 노이즈 감소, 엣지 보존 등이 있다. 극단적인 이미지를 픽셀 값의 중간값으로 중앙값을 대체함으로서 밝거나 어두운 노이즈가 있어도 실제 이미지에는 영향이 없다. 경계를 부드럽게 하지만 가우시안 블러처럼 흐리게 하지 않아서 자세한 경계가 유지된 상태에서 노이즈를 줄어들게 할 수 있다. 즉, 예상되는 기대값으로는 선명한 경계가 중요하면서 노이즈를 줄일 수 있다는 점이 있다.
2. 평균 필터의 기대값으로는 이미지를 부드럽게 할 수 있다는 점과 고주파 성분이 감소한다는 점이다. 평균 필터는 단순히 평균을 계산해 중앙값을 대신하기 때문에 이미지가 뭉개지는 효과가 있다. 노이즈가 만약 여러 군데에 고르게 있다면 위의 필터를 적용해 노이즈를 없앨 수 있다. 평균 필터는 경계값을 뭉개기때문에 고주파 성분을 감소시키고 색상 차이를 줄어들게 할 수 있다. 즉 예상되는 기대값은 넓게 분포되어 있는 노이즈를 제거할 수 있다.
3. 샤프닝 필터의 기대값으로는 디테일 및 엣지 강조가 있다. 고주파 성분을 평균 필터와는 달리 강조하므로 경계가 더 뚜렷해진다. 즉 흐릿한 화면을 더 명확하게 보이도록 할 수 있다. 또한 경계선을 강조해 배경과 객체의 경계를 강하게 강조해서 더욱 선명한 이미지를 얻을 수 있다. 즉, 흐릿한 이미지를 선명하게 할 수 있다.