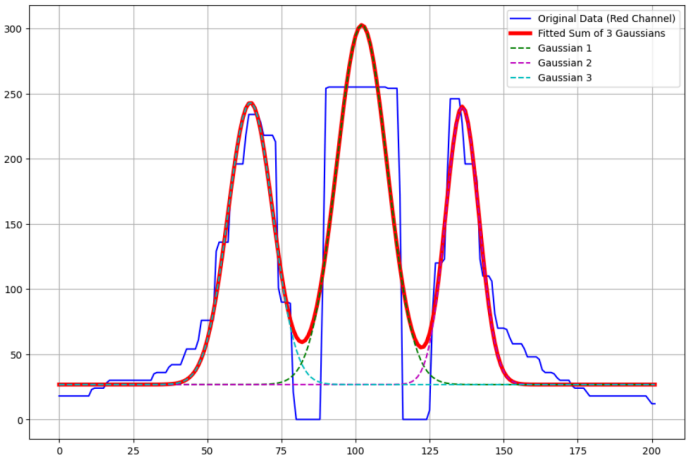
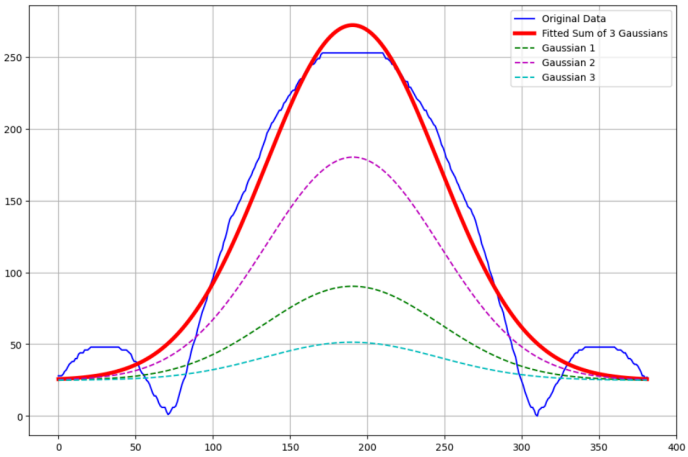
진행사항

approximation by superposition of 1D gaussians

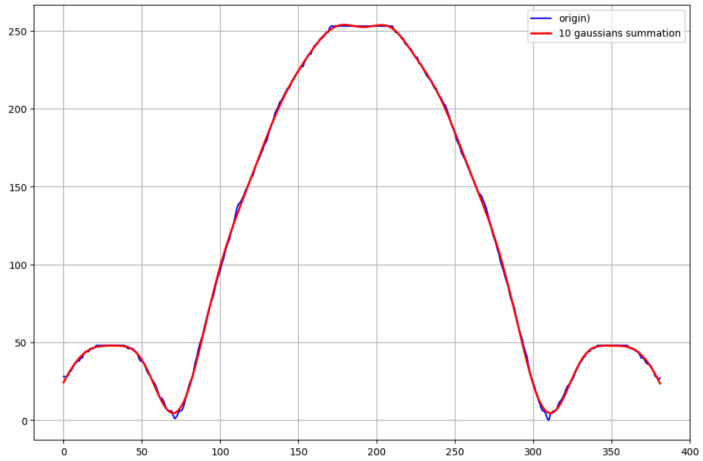
참고

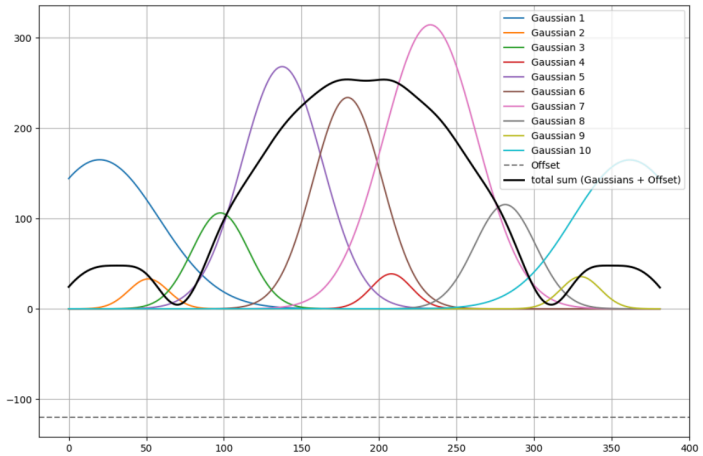
최소제곱법을 이용해서 colab 상에서 1D 수준에서 pixel들의 rgb 값을 읽고 3개의 gaussian을 이용해서 approximation 진행. 우측 그래프의 경우 양 끝 값이 작아서 근사가 안되는 것으로 보임. 파이썬 scipy 라이브러리에서 curve_fit 모듈과 cpp 상에서는 ceres 라이브러리를 사용하여 구현했다.





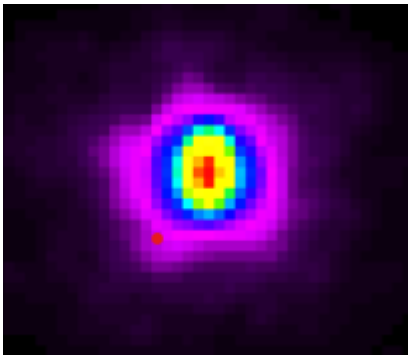
10 개의 gaussian으로 fitting 했을 때 결과와 각 gaussian samples

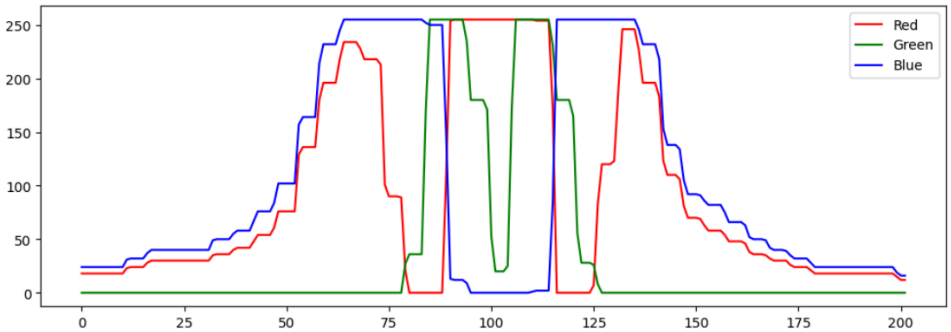




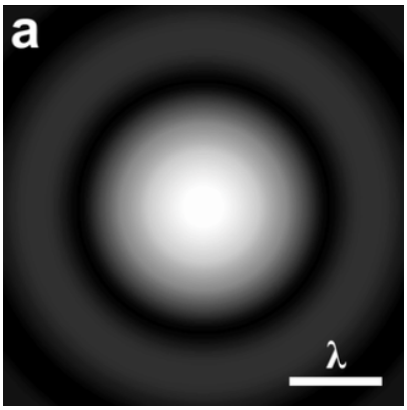
vglare sample data

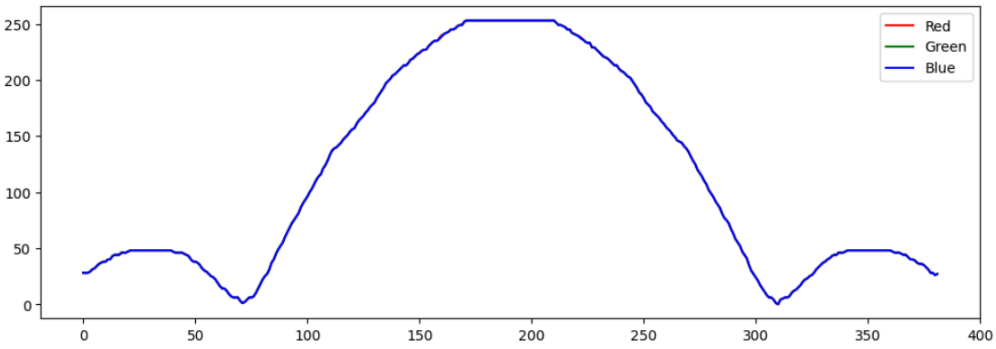
sample1





sample2





Gaussian function 1D fourier series aggregation

cpu상에서 initial update시에 각각의 gaussian function에 대한 fouries series를 실행하고, 각각의 coefficient의 합을 sum_a, sum_b vector에 저장했다. FourierSeries1D를 구성하는 클래스는 기존에 rex에 존재하던 FourierSeries1D plugin에서 사용했다.

```
    fs_1d = new FourierSeries1D(gaussian_function, 4.0f, 10);
    fs_1d_2 = new FourierSeries1D(gaussian_function2, 4.0f, 10);

    sum_a = vector<float>(100, 0);
    sum_b = vector<float>(100, 0);

    for(int i = 0; i <= 100; i++)
    {
        sum_a[i] = fs_1d->a[i] + fs_1d_2->a[i];
        sum_b[i] = fs_1d->b[i] + fs_1d_2->b[i];
    }
```

위는 initial_update()에서 coefficient를 합하는 부분이다.

```
std::vector<vertex> vertices = export_result(4.0f, -1.0f,1.0f, 0.01f, 10,
(float*)sum_a.data(), (float*)sum_b.data());
```

```
vao = gxCreateVertexArray("LINES", vertices.data(), vertices.size());
if(!vao) return;
FBO->set_state(false, false);
FBO->bind(FOURIER1D);
FBO->clear();
effect->bind("draw_fs_1D");
vao->draw_arrays(0, 0, GL_POINTS);
```

export_result 함수는 sum_a, sum_b coefficient vector를 참조하여 cos, sin에 대한 10번의 합으로 값을 저장한다. 위는 render()부분에서 실행된다.

그 결과로, 두 개의 gaussian function이 fourie series의 aggregation으로 그려진다.