**프로젝트 완료 보고서**

프로젝트명 : Floyd Steinberg 디더링 알고리즘과 Direct Binary Search 알고리즘

작성자 : 노지홍

작성일자 : 2020-05-25

**프로젝트 개요**

주제 : Direct Binary Search 와 Floyd Steinberg 디더링 알고리즘

목적 : 기존의 Floyd Steinberg 디더링 알고리즘의 한계점을 해결하기 위해 Direct Binary Search 디더링을 적용시킴.

기간 : 2020-03-25 ~ 2020-05-22

개발도구 : Visual Studio 2017

**속도비교(초)**

이미지 사이즈 Floyd Steinberg DBS

512 X 512 0.0035 CPU : 8.351 GPU : 0.2138

2048 X 2560 0.056 CPU : 113.3 GPU : 3.158

4096 X 3072 0.497 CPU : 346.6 GPU : 7.248

**한계점 및 결론**

1. Floyd Steinberg 알고리즘은 에러값을 갱신해 나가면서 점의 픽셀값을 구하는 알고리즘이기 때문에 연산의 방향으로 인해 점이 방향성을 띄우며 찍히는 한계점을 가지고 있음. 또한 이미지의 픽셀 값이 흰색에서 검은색 또는 검은색에서 흰색으로 갑자기 바뀌는 구간에선 연산 방향으로 공백이 발생하는 현상이 나타남.

2. Floyd Steinberg는 알고리즘 특성상 다음의 연산이 전 단계의 연산결과에 의존성이 있으므로 GPGPU를 활용하여 병렬화시키는 것이 불가능하다.

3. DBS(Direct Binary Search)는 Floyd Steinberg의 한계점을 해결시킬 수 있는 알고리즘으로 우수한 품질을 생성한다. 단, 경계선 부분에서 Edge가 번지는 경향이 있다. (가우시안 필터의 영향)

4. DBS는 상당히 복잡하고 연산이 많은 알고리즘으로 Floyd Steinberg 알고리즘에 비해 속도가 매우 느리다. (실시간 온라인 프로세싱으로 쓰이지 못하는 결정적인 이유)

5. DBS는 점의 농도가 2%이하 98%이상인 구간에서 점이 유실되는 클리핑 현상이 일어난다. (DBS의 한계) 이를 해결하기 위한 Clipping Free DBS based halftoning에 대한 논문이 있다.

6. Clipping Free DBS based halftoning은 쿼드레벨의 블록구조를 이용하여 GPGPU로 병렬화를 시켜 구현할 수 있다. (CPU기준 : 346.6초 -> GPU기준 : 7.248초) 이 논문은 차후에 분석할 예정..

**코드 깃허브 주소**

1. <https://github.com/wlghd28/DirectBinarySearch>