<보고서>

17011703 소프트웨어학과 홍예지

Summed area table은 이중 배열에서 각각 0에서부터 특정 자리까지 더한 값을 그 해당 자리(이중배열)에 값을 저장해서 평균을 계산할 때 더 쉽고 빠르게 할 수 있도록 도와준다. 평균필터를 구할 때는 4중 포문을 구해서 돌렸기 때문에 구현할 때의 속도가 훨씬 느렸다. 하지만 Summed area table은 미리 값을 누적시켜놓은 이중배열때문에 특정 부분만을 구해서 쓸 때 훨씬 빠르게 사용할 수 있다.

Summed area table을 구현할 시에 먼저 열끼리의 값을 구해서 앞의 값부터 누적해서 현재 자리 위치까지 더해 값을 저장하고 다음 누적된 값을 이중 배열에서 한 번 더 이중포문을 사용해서 행끼리 더해주었다. 열끼리 우선 더해준 뒤에 그 누적된 값에 행을 더해야 Summed area table이 완성된다. 기존에 사용했던 평균필터의 4중 포문보다 2중 포문을 2개 따로 사용하는 것이 속도도 빠르고 구현하는데 더 시간이 줄어들기 때문에 효율적이라고 할 수 있다.

사진의 크기는 제각각이기 때문에 원본으로부터 색을 받고 복사본에 색을 줄 때 동적할당을 사용해 주어야 하는데 y축과 x축이 있기 때문에 이차원 배열을 동적할당받아 사용해준다. 먼저 y축방향으로의 이미지의 y축 크기로 할당을 받고 그 할당 받은 열 속에서 행을 for문을 통해 할당받아준다. 이렇게 되면 변수 f에 이차원배열을 사용할 수 있게 된다. 색을 동적할당받는 것이기 때문에 CvScalar 로 변수를 선언해서 구현해준다. 처음에 타입을 정할 때는 어떤 타입으로 받아야하나 고민했었는데 색을 주고 받는 것이기 때문에 CvScalar로 변수를 선언해서 받아주는 것이 옳다.

Summed area table을 구현해주기 위해서 이중포문을 구현하는데 y축포문으로시작해 안에는 x축포문으로 구현한다. 우선 열방향으로 누적된 값을 더해줘서 전체 이미지를 한 번 훑어준다. 첫번째 이중포문에서는 값을 cvGet2D함수를 이용해서 받아주어야한다. 밝기값인 RGB를 모두 따로 더해주어야 한다. 처음에 따로 더해주는 것에 대해 혼동이 와서 그냥 더하려고 시도를 했으나 잘 구현되지않아 3부분으로 나누어서 따로 더해졌다. 밝기 값은 R,G,B 값이 모두 다르기 때문에 그냥 바로 더해줄 수가 없다. 세부분으로 나누어서 더해줘야 구현이 되기 때문에 각각의 이차원배열에 더해서 구현해준다.

다음에는 행을 누적해서 더해주어야 하는데 이는 위에서 열끼리 누적해줄 때와 마찬가지로 이중포문을 사용해서 구현해준다. 이 때는 x축으로 시작해서 y축으로 넘어와서 이중포문을 구현해준다. 색을 이미 첫번째에서 받아주었기 때문에 두번째 이중포문에서는 이미 들어있는 값들을 더해주는 것이 요점이다.

그리고는 이중포문을 한 번 더 사용해주는데, 이는 summed area table에서 보다시피 누적된 값을 이용해서 이차원 배열 처음부터 끝까지 값을 평균필터의 값으로 구현해주어야 한다. 그 색을 주고싶은 특정 위치의 이차원 배열의 값에서 열에서 (2*k+1)을 뺀 위치의 값을 빼주고

행에서 (2*k+1)을 뺀 위치의 값을 빼주고 두 번 빠지게 된, 열과 행에서(2*k+1)을 뺀 위치의 값을 다시 더해준 값에서 전체 면적을 나눠주면 완벽하게 평균필터의 값이 구현된다. 이렇게 누적된 값을 이용해서 구현해주면 2중포문으로 구현한 중간필터값이 빠르게 돌아간다. 처음부터 누적된 값에서 커널의 값이 달라진다 하더라도 이중포문만 돌려주면 되기에 속도가 일정해진다. 이 문제의 요점은 이중포문을 사용해서 속도가 빨라지게 하는 것이었는데 난 처음에 다 더해서 누적된 값을 구할 때도 4중포문을 이용해서 구현하려고 했다. 행끼리, 열끼리 더했으면 속도가 빨라지는데 이를 이해하지 못하고 4중포문을 돌렸다. 결과로는 정말 이미지가 느리게 떴고 이 방법은 틀렸다는 것을 깨달았다. 누적된 값을 들어오는 커널의 값에 따라서 바로 빠르게 값을 계산해서 필터를 구현할 수 있었고 색 또한 세개의 분야로 나누어서 따로 저장해주어야 했다. 마지막으로 동적할당을 해제해주어야하는데 포문을 이용해서 행을 동적할당 해제해주고 전체를 해제해주었다. 중간필터를 구현해주기 위해 동적할당을 두 이차원 배열에 받아서 색을 주고 받아 복사본에 넣어주었다.

커널의 크기에 상관없이 이중 포문으로 구현해주어 일정한 속도로 수행될 수 있게 해주었으며 평균필터에서 사용되었던 4중배열은 실행시간이 더 길다. 계속 중복되어서 계산하기 때문에 중복되는 계산을 한번의 계산으로 해결할 수 있는 것이 summed area table이다. 이미 한번의 누적된 계산을 실행하기 때문에 매번 더해서 평균을 내는 수고를 할 필요가 없다. 이미 더해진 수에서 필요한 면적에 따른 값만 계산해서 평균을 내주면 된다. 이로써 실행속도가 더 빠르고 간편한 summed area table을 이용한 평균필터를 구현하는 것이 끝났다.