Edge direction을 이용한 A Non-Reference Blur Metric

정 우 근
그래픽스 응용 연구실 부산대학교

Chung Woo Keun
Graphics Application Lab. Pusan National University

wkchung@pearl.cs.pusan.ac.kr

January, 2008

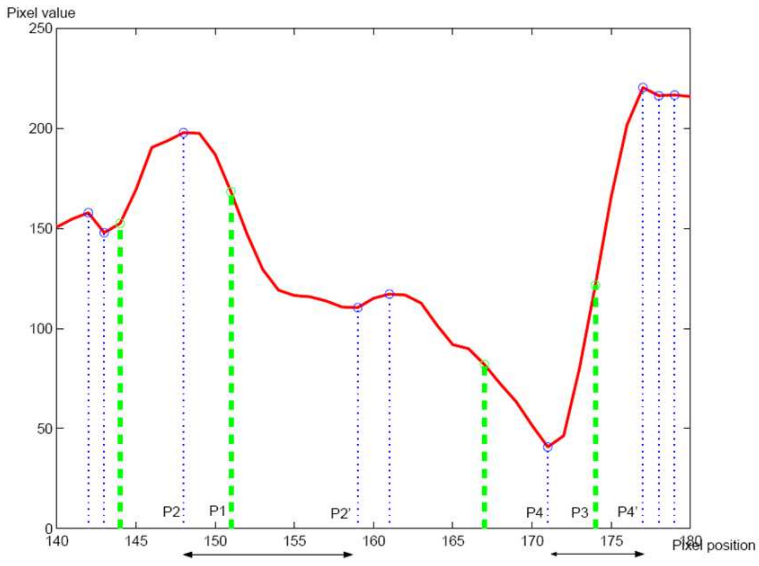
요 약

본 보고서는 이전 보고서와 논문에서 제시되었던 Blur Metric 방법을 개선하고 실험하기 위한 새로운 Blur Metric 측정 방법을 제시한다. 이전에 제시되었던 Blur Metric 측정 방법에서는 Vertical 탐색 방법을 통하여 에지(Edge)을 탐색하고 해당 에지의 두께(Slope)의 넓이를 측정하여 Blur를 측정 하였다. 본 보고서에서는 이전 Blur Metric 측정 방법의 문제점을 파악하고, 그 문제점을 해결할 새로운 방안을 제시한다. 본 보고서에서 제시될 방법은 Blur된 이미지가 가지는 에지의 방향을 고려하여 해당 하는 방향에서의 넓이를 계산하여 그 평균을 제시하는 것이다.

주제어: NO-REFERENCE, BLUR METRIC, Direciotn Edge

1 개요

본 보고서는 이전 보고서와 논문에서 제시되었던 Blur Metric 방법을 개선하고 실험하기 위한 새로운 Blur Metric 측정 방법을 제시한다. 이전 보고서와 논문에서 제시되었던 Blur Metric 측정 방법은 선명하지 못한 사진의 에지(Edge)의 탐색을 통하여 Blur를 측정 하였다. 측정 방법은 선명하지 못한 사진



(a)Blur Image의 한 row 값을 나타내는 그래프

그림 1: Blur Image의 한 row 값을 나타내는 그래프이다. 본 그래프를 보면, $P2$ - $P2$'와 $P4$ - $P4$'가 Blur된 Edge를 나타낸다.

의 에지를 추출하여 판별하는 방법이었다. 에지를 추출하는 방법은 소벨(Sobel) 에지 추출을 통하여 추출하였다. 추출된 이미지에서 Pixel 값들의 Horizontal 탐색을 통하여 Blur된 에지(Edge)를 찾아 Pixel 값들을 통하여 Blur를 측정하였다. 여기서 Blur된 에지는 그림1 에서 $P2$ - $P2$‘와 같은 감소하는 그래프를 그리는 Pixel 값들과 $P4$ - $P4$‘와 같은 증가하는 그래프를 그리는 Pixel 값을 가진다. 하지만 이전에 제시되었던 Blur Metric 측정 방법에는 몇가지 문제점이 있었다. 다음 단락에서 이전 보고서와 논문의 Blur Metric 측정 방법을 간략하게 다시 한번 알아보고 그 문제점을 제시해보겠다.

2 기존 Blur Metric의 문제점

이전 보고서와 논문에서 제시되었던 Blur Metric 측정 방법에 대해서 간략히 알아 보겠다. 이전 보고서와 논문에서는 Blur 측정의 구현의 편의를 위하여, OpenCV 함수를 사용하였다. OpenCV 함수는 이미지 프로세싱에 다양한 함수를 제공한다. Blur측정을 위해 이전 보고서와 논문에세 제시되었던 방법은 다음과 같다.

1. Blur 측정할 이미지를 로드하여 OpenCV 함수를 이용하여 Gray Image로 바꾸어 준다.
2. OpenCV 함수의 Sobel 마스크를 통하여 Gradient를 한다.
3. Sobel 마스크를 거친 Gray Image 파일의 픽셀값을 저장한다.
4. 그림3의 그래프값에 해당하는 픽셀값들을 찾는다.
5. 해당하는 픽셀값들을 에지라 판단하고 평균값을 산출한다.
6. 산출된 값들의 평균값을 계산하여 Blur를 측정한다.

기존에 제시되었던 Blur Metric 측정 방법에서 Edge 추출은 OpenCV 함수의 소벨(Sobel) 마스크를 통하여서 Edge를 추출하였다. 소벨 마스크는 모든 방향의 에지를 추출한다. 또한 돌출한 화소값을 비교적 평균화하므로 잡음에 대체로 강인한 편이다. 하지만 소벨 마스크는 수직, 수평 방향 에지 보다 대각선 방향 에지에 더 민감하게 반응하는 것으로 알려져 있다. 기존에 제시되었던 Blur Metric 측정 방법에서는 Edge 추출후 Horizontal 탐색을 통하여 Edge을 추출 하였다. 여기서 문제점이 발생한다. Edge의 탐색은 Horizontal이나 Edge를 추출하는 소벨 마스크는 정작 대각선에 강하다. 또한, 소벨 마스크를 통하여 추출된 Edge 값들의 탐색의 경우 이전 측정 방법에서는 Horizontal 탐색 방법을 제시하였다. 하지만 Horizontal 탐색 방법은 선명하지 못한 사진의 Blur 된 Pixel 값을 모두 탐색 할수가 없다. 선명하지 못한 사진의 경우 어떤 방향으로 흔들렸는지 알수가 없기 때문이다. 그림2에서 처럼 선명하지 않은 사진의 방향성을 미리 예측 할 수가 없다. 그러므로 Horizontal 탐색만으로는 선명하지 않은 사진의 방향성을 미리 예측할수 없는 만큼 Blur의 측정도 어렵다. 다음 단락에서는 새로운 Blur Metric 측정 방법을 제시해보겠다.

(a) 시계1 (b) 시계2

그림 2: 시계1의 경우 상하로 흔들린 것을 알수 있고, 시계2의 경우는 좌우로 흔들린 것을 알수 있다.

3 Blur Metric 측정 방법

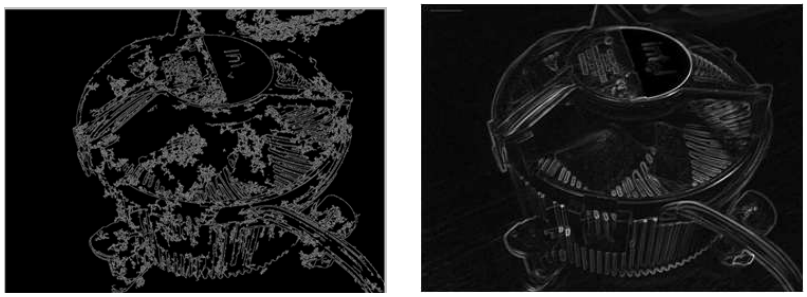
이번에 제시될 새로운 Blur Metric 측정 방법은 Gradient 의 방향성을 구하고, Blur를 측정하고자 하는 이미지를 1차 미분을 통하여 에지를 추출하고, Gradient의 방향성을 감지하여 그 방향성에 존재하는 에지를 탐색하여 에지의 넓이를 측정하여 Blur Metric을 측정한다.

$$\theta = \tan^{-1}\left(\frac{G_y}{G_x}\right) \qquad G_x = \frac{\partial f}{\partial x}, \quad G_y = \frac{\partial f}{\partial y}$$

(a) 식1 (b) 식2

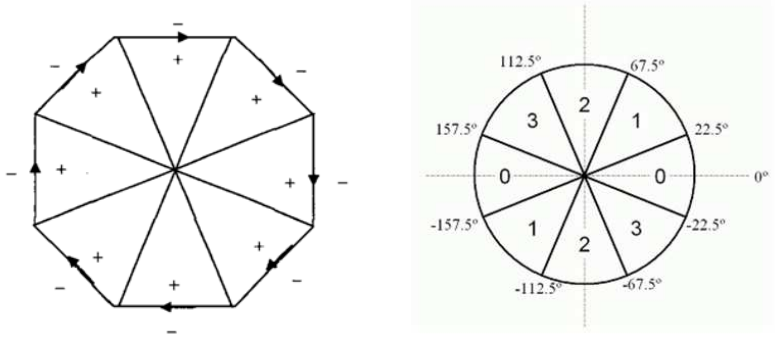
그림 3: f는 현재 Blur 를 측정할 Image이다. x, y는 Image의 x,y좌표이다.

첫번째로 제시된 Gradient의 방향성을 구하는 식은 그림3 과 같다. 그림3에서 식2을 통하여 매 픽셀 값에서 x, y 좌표를 통해 G_x, G_y 를 추출한다. G_x와 G_y의 tan 연산을 통하여 Θ 값을 구한다.



(a)사진 1 (b) 사진 2

그림 4: 캐니 마스크(사진 1)를 거친 후 이미지, 소벨 마스크(사진 2)를 이용한 결과보다 에지가 더욱 두터워 지고 잡음이 제거 되었음을 알 수가 있다.



(a) Gradient Direction 1 (b) Gradient Direction 2

그림 5: Gradient Direction 1에서 *Arrow*는 에지의 방향, +,−는 Gradient 의 방향을 나타낸다.

두번째는 Blur를 측정할 이미지에서 에지를 추출한다. 이전 보고서와 논문에서는 Soble Mask를 이용하여 에지를 추출하였다. 하지만 이번에 에지를 추출할 미분 방법은 캐니 마스크(Canny Mask) 이다. 대부분 에지 추출 마스크는 잡음에 대해 민감하므로,작은 잡음이라도 그것을 에지로 간주하여 추출하는

경우가 많다. 이러한 단점을 보완하는 캐니 마스크를 이용한 에지 추출 기법이 있는데, 실제로 잡음에 민감하지 않게 하며 강한 에지를 추출하는 것에 목적을 둔다. 4에서 사진 1이 캐미 마스크를 적용한 후 이미지이다. 전 보고서에서 볼 수 있었던 소벨 마스크를 이용한 결과(사진 2)보다 에지가 더욱 두터워지고 잡음이 제거된 것을 알수가 있다. 캐니 마스크를 이용한 에지 추출역시 OpenCV에서 제공하고 있으므로, OpenCV함수를 쓰도록 한다.

세번째는 그림5에서 제공하는 Gradient Direction 1을 통하여 직접적인 Blur를 측정한다. 그림5에는 총 8개의 방향을 가지고 있다. $\pm 0, \pm 45, \pm 90, \pm 135$를 알 수가 있다. 그림4에서 보여지는 $Arrow$는 에지의 방향을 나타내고 있다. 그리고 +,−는 Gradient의 방향성을 나타낸다. 첫번째에서 우리는 Gradient의 방향성을 구하였다. 하지만 Θ 값이 규칙적인 방향을 가질 수가 없다. 그리하여 그림5의 Gradient Direction 2를 보고 8개의 일반적인 값을 추출한다. 아래에 이에 해당하는 코드의 일부분이다.

```
if((thisAngle < 22.5) && (thisAngle > -22.5) )
    newAngle=0;
if((thisAngle > 67.5) && (thisAngle < 112.5) )
    newAngle=90;
```

thisAngle이 첫번째에 Gradient의 방향성을 구해 나온 Θ값이다. 이제 에지의 두께(Slope)를 측정하는 방법을 소개 하겠다. 에지의 두께를 측정 하는 방법은 5의 Gradient Direction 1 을 이용한다. 먼저 첫번째에서 주어진 식으로 구해진 Θ 을 5에서 Gradient Direction 1에 적용 시켜서 에지의 방향에 해당하는 $Arrow$를 찾는다. 예를 들어 첫번째에 주어진 식으로 구해진 Θ값이 +90도라 하면, 그림5보면 +90도는 ↑방향을 가리키고 있다. ↑ 의 좌측은 −, 우측은 +이다. 5에서 Gradient Direction 1 표가 말하는 바는 예를 들어 +90도 일때는 좌측으로 −이므로 감소하는 그래프를 그리는 Pixel 구간을 찾는 것이고, 우측으로는 +이므로 증가하는 그래프를 그리는 Pixel 구간을 찾는 것이다. 이때 +, − 구간을 만족하는 구간을 탐색하면서 각 Pixel값을 카운팅 한다. 이 카운팅 값이 한 에지의 Blur 측정한 값이다. 모든 에지를 탐색하여 평균을 내면 그 값이 Blur를 측정한 값이 되겠다. 다음 단락에서는 지금 까지 Blur Metric 측정 방법의 실질적인 구현 부분을 설명 하도록 하겠다.

4 구현

본 단락에서는 Blur Metric 측정 방법의 실질적인 구현 부분을 설명하겠다. Blur Metric 측정의 편의를 도모하기 위해 이미지 프로세싱에 다양한 함수를 제공하는 OpenCV 함수를 사용하였다. 첫번째로 전 단락에서 제시하였던 Gradient Direction을구해보겠다. Gradient Direction은 에지의 방향을 뜻한다. 영상 내에 3x3 Mask를 거쳐 미분을 추출한다고 하면 x 방향의 미분은 $G_x = f(x+1, y) - f(x, y)$, y방향의 미분은 $G_y = f(x, y+1) - f(x, y)$ 로 표현 한다. 이렇게 구해진 G_x,G_y을 통해서 tan값을 구한다. G_x, G_y를 구하는 소스코드는 아래와 같다. 아래의 소스 1,2에서 볼수 있는 $cvGetReal2D(Image, x, y)$는 Image에서 x, ,y 좌표에 Gray Level값을 가져오는 것이다. 3개의 인자가 들어 가는데, 첫번 째 인자는 Gray Level 값을 가져올 ImageFile 의 포인터값이며, 나머지 2개의 인자는 x,y 좌표이다. Gradient Direction을 구하기 위하여 1차 미분을 사용하였다. 하지만 1차 미분 효과와 유사한 관계로 소벨 마스크와 같은 1차 미분 에지 추출 마스크로 대치할 수 있다.

```
for(int i=0;i<height-mask_height+((mask_height-1)/2);i++){
     for(int j=0;j<width-mask_width+((mask_width-1)/2);j++){
1.       Gx=cvGetReal2D(image, i+1, j)-cvGetReal2D(image, i+1, j+2);
2.       Gy=cvGetReal2D(image, i+2, j+1)-cvGetReal2D(image, i, j+1);
     }
}
```

1차 미분을 통하여 구해진 G_x, G_y를 통하여 Θ를 구하는 코드는 다음과 같다.

```
double thisAngle=(atan2(Gx, Gy)/3.14159)*180.0f;
```

위와 같은 방식으로 구해진 thisAngle 값은 그림5의 Gradient Direction 2를 통하여 일반적인 데이터 값을 추출한다. 이렇게 추출된 일반적인 데이터 값 즉, Gradient Direction 값을 2차원 배열에 저장한다. 그다음에는 OpenCV에서 제공하는 캐니 마스크 함수를 이용하여 Blur를 측정할 Image를 필터링을 거친다. OpenCV에서 제공하는 캐니 마스크 함수는 $GS\_canny\_edge\_cvCanny(ImageFile, threshold1, threshold2)$ 이다. 3개의 인수가 들어가는데, ImageFile은 캐니 마스크 필터를 거칠 Image File이며, 두가지의 threshold값이 들어간다. 이 두 가지의 threshold 값은 1차 미분 마스크를 적용한 결과 중, 에지라고 추측되는 화소값이 나왔는데, 만약에 이 주위의 화소값 중에서 추측되는 화소값보다 더 큰 화소값이 존재할 경우가 생긴다. 이것은 가짜 에지라는 결론을 내린다. 또한 이 에지는 반드시 제거 또는 억제하여야 한다. 이것을 가리켜 비최대치 억제라고 한다. 이런 상황에서는 이중 threshold 값을 이용해야 한다. 왜냐하면 threshold 값을 낮게 설정하면 에지가 아닌 것이 남아 있을 수 있으며, threshold 값을 높게 잡으면 진짜 에지의 일부를 놓칠 수 있다. 그리하여 두 가지의 treshold 값을 설정하는 이유이다. 이렇게 OpenCV에서 제공하는 캐니 마스크 함수를 이용하여 Blur를 측정할 이미지를 필터링 시킨다. 필터링 시킨후 이미지 파일의 ImageData 값을 이용하여 2차원 배열에 Pixel 값을 저장 시킨다. 캐니 마스크 필터링을 거친 후 이미지의 Pixel값은 0 또는 255 값을 가진다. 255 값을 가지는 곳은 에지가 있다는 것을 나타내며, 0의 값은 에지가 없다는 것을 나타낸다. 캐니 마스크를 거친 후 이미지의 Pixel 값을 가지는 2차원 배열을 차례대로 탐색하면서 255값이 존재하는 곳의 x,y좌표를 이용하여, Gradient Direction의 정보를 저장하는 2차원 배열에 x,y값에 대응하는 Angle 값을 가지고, Blur를 측정할 이미지의 Pixel 정보를 탐색하여 Blur를 측정 한다. 그림5에서 Gradient Direction 1를 참조하여 Blur를 측정하는 부분에 대한 코드를 살펴 보겠다. 소스의 코드는 아래와 같다.

```
for(int i=0;i<height;i++){
        for(int j=0;j<width;j++){
1.        if(cannyArr[i][j]==255){
2.          if(direction[i][j]==0){
                    edgeCount++; edgeSum+=returnEdgeAvg(imageArr,
                    height, width, i, j, 0);
            }
            ....
            ....
            ....
          }
        }
}
```

위의 소스의 1.에 $cannyArr[][]$는 Blur를 측정할 Image에 Canny Mask 필터링을 거친후 Pixel 정보를 담고 있는 2차원 배열이다. 2.에 $direction[][]$는 Image 에서 Gradient Direction의 정보를 담고 있는 2차원 배열이다. $cannyArr[][]$ 탐새중 255을 가지고 있는 즉, 에지를 가지고 있는 x, y 좌표에서 $direction[][] == 0$ direction 값이 0도일때(다른 Gradient Direction의 구현은 위와 동일하므로 생략), Blur를 측정하는 $returnEdgeAvg(imageArr, height, width, i, j, 0)$ 를 호출한다.

```
if(nFlag==0){ int i=x;
  while(true){
   if(i==0){ break; }
      if(bFirst==false){
        prevData=data[i][y]; pixelCount++; pixelSum+=data[i][y];
        bFirst=true; i--;
      }else{
1.       curData=data[i][y];
2.       if(curData > prevData) {
            break;
         }
3.       pixelSum+=data[i][y]; pixelCount++; i--;
4.         prevData=data[i][y];
      }
  }
  prevData=data[x][y];
  i=x++;
  while(true){
    if(i==height-1){ break; }
    curData=data[i][y]; //현재 값 기억 시키고
5.  if(curData < prevData){ break; }
    prevData=data[i][y]; pixelSum+=data[i][y]; pixelCount++; i++;
  }
  return pixelCount;
```

위의 소스는 Gradient Direction 이 +0도 일때의 코드이다. 그림5에서 Gradient Direction 1를 보면 +0도 일경우 x, y 좌표 즉, $cannyArr[x][y] == 255$ 에지가 존재하는 좌표에서 y측을 기준으로 y측 좌표가 감소하는 구간에서는 감소하는 그래프를 그리는 Pixel값들을 찾고 증가하는 구간에서는 증가하는 값을 찾는 것이다. 위의 소스에서 첫번째 $While$문이 감소하는 구간에서 감소하는 그래프를 그리는 Pixel 값을 찾는 것이며, 두번째 $While$문이 증가하는 구간을 찾는 곳이다. 1.에서 탐색하는 현 좌표 값에 Pixel 값을 저장시키며 2.에서 더이상 감소하지 않는 구간을 발견시 루프문을 탈출한다. 만약 2.을 만족할 경우 Blur 측정을 위하여 pixelCount 값을 증가 시킨다. 3.번의 소스에 해당한다. 마지막으로 현 좌표의 Pixel 값을 기억시킨다. 4.번의 소스에 해당한다. 이렇게 하여 Blur를 측정하고자 하는 Image의 모든 위치를 탐색하며 위의 코드를 거치면 이 논문에서 제시하였던 Blur 측정이 끝난다. 다음 단락에서는 위의 코드를 바탕으로 여러가지 선명하지 못한 이미지로 실험 한 결과를 알아보겠다.

5 실험

본 단락에서 실험에 들어가기 앞서 이전 보고서에서 했던 실험 결과를 살펴 보도록 하겠다. 이전 보고서에 했던 실험 결과를 보면 선명하지 못한 사진이 공통적인 데이타가 선출 되지 못했으며, Blur를 측정하고자 하는 Image가 많은 에지의 유무에 따라서 실험결과가 만족적이지 못하였다. 그리하여 이번에 새롭게 제시하였던 Blur Metric 측정 방법의 실험 결과를 보도록 하겠다.

이번 실험의 결과의 경우 사진1, 9의 경우 선명한 사진 보다 높은 결과를 나타내었고 사진5, 6의 경우 같은 결과를 나탄냈으며 나머지 결과값은 전부 선명한 사진보다 낮은 결과를 나타내었다. 에지가 많은 사진과 에지가 적은 사진의 차이는 크게 나타나지 않았으며, 결과 값도 만족스러울 만하나 선명한 사진과 선명하지 못한 사진의 데이타 차이 값이 그리 눈에 띄게 크게 차이가 나지 않았다.

6 결론

본 보고서에서는 이전 보고서 했던 Blur Metric 측정 방법에서의 문제점을 개선 하고, 새로운 Blur Metric 측정 방법을 제시하였다. 새로운 Blur Metric 측정방법은 Gradient Direction을 통하여 Blur를

표 1: Blur 측정 데이타 값

	선명한 사진	**선명하지 못한 사진**
사진 1	4	3
사진 2	4	2
사진 3	4	3
사진 4	4	4
사진 5	4	4
사진 6	4	3
사진 7	1	1
사진 8	7	5
사진 9	4	5
사진 10	4	9
사진 11	5	12

표 2: Gradient Direction Blur 측정 데이타 값

	선명한 사진	**선명하지 못한 사진**
사진 1	9	10
사진 2	11	9
사진 3	9	7
사진 4	9	8
사진 5	9	9
사진 6	9	9
사진 7	9	7
사진 8	12	10
사진 9	12	18
사진 10	12	9

측정하였다. Gradient Direction를 통하여 Blur 측정한 결과를 따져 보면 그리 만족스러운 결과는 얻지 못했으나, 전 보고서 보다는 조금 일관성 있는 데이타 값을 이끌어 내었다. 전 보고서와는 달리 이미지에 따라 결과값이 틀린 경우는 보기 드물었다. 하지만 조금더 나아진 결과값을 얻기 위해서는 Blur Metric 측정시 맹목적인 탐색이 아닌 특정 부분을 측정 하는 것이 더 나은 결과값을 추출할 것이다.

References

[1] 이문호 정성환. *오픈소스 OpenCV를 이용한 컴퓨터 비전 실무 프로그래밍*. 홍릉과학출판사, 2006.

[2] 하종은 강동중. *Visual C++을 이용한 디지털 영상처리*. 사이텍미디어, 2003.

[3] S.Winkler Pina Marziliano, F.Dufaux and T.Ebrahimmi. A no reference perceptual blur metric. in proc. $InternationalConferenceonImageProcessing$2002, volume 3, pages III-57III-60 vol.3, 2002.

[4] R.R Bailey Sei-Wang Chen Yun-Chung Chung, Jung-Ming Wang and Shyang-Lih Chang. A non-parametic blur measure based on edge analysis for image processing applications. in proc. $IEEEConferenceonCyberneticsandIntelligentSystems$, volume 1, pages 356-360 vol.1, 2004.

[5] Z. Lu X. Yang S. Yao F. Pan L. Jiang E. Ong, W. Lin and F. Moschetti. A no-reference quality metric for measuring image blur. in proc.

SeventhInternationalSymposiumonSignalProcessingandItsApplications, volume 1, pages 469-472 vol.1, 2003.