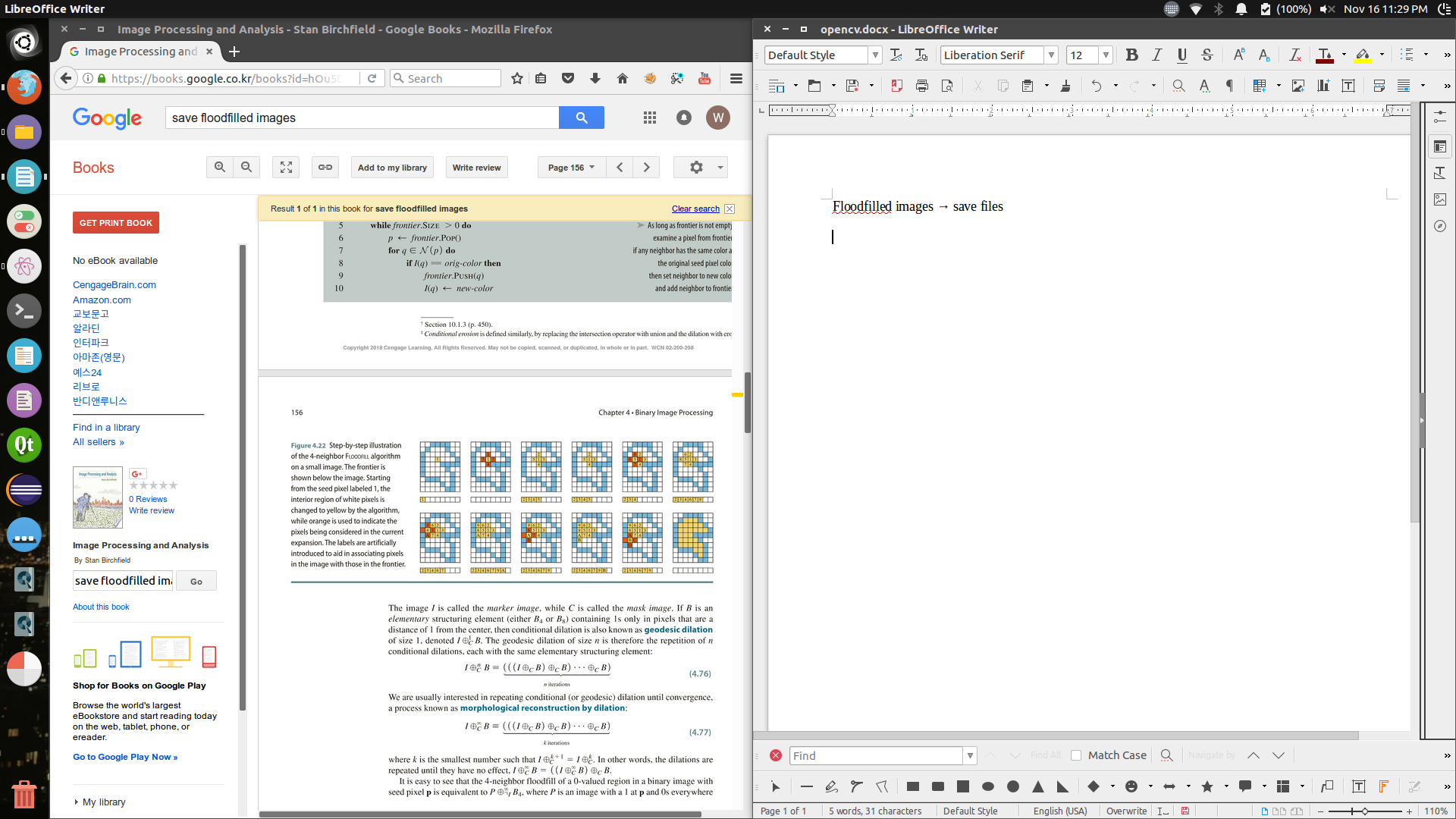
1. FLOODFILL as an image processing algorithm

1 - A. 이미지프로세싱에서 아예 floodfill이라는 개념이 존재.

원리 → 우리가 슬라이드에서 설명할 때 쓴 region growing이랑 같아 보임.

  
Illustration 1

Floodfill = repeated conditional dilations

Conditional dilation of an image I with respect to another image C : I⊕CB = (I⊕B) ⋂ C

1 - B. Algorithm

**floodfill(I, p, new-color)**

**Input**: image I, seed pixel p, and new color

**Output**: all pixels in I connected to p are colored new-color

*orig-color ← I(p)* // If seed pixel is already new color,

**if** *orig-color != new-color* **then** // then terminate.

*frontier.push(p)* // Otherwise, set frontier to seed pixel,

*I(p) ← new-color* // and set pixel to new color.

**while***(frontier.size > 0)* **do** // As long as frontier is not empty

*p ← frontier.pop()*  // examine a pixel from frontier;

**for** *q in Neighborhood(p)* **do** // If any neighbor has the same color as

**if** *I(q) == orig-color* **then** // the original seed pixel color,

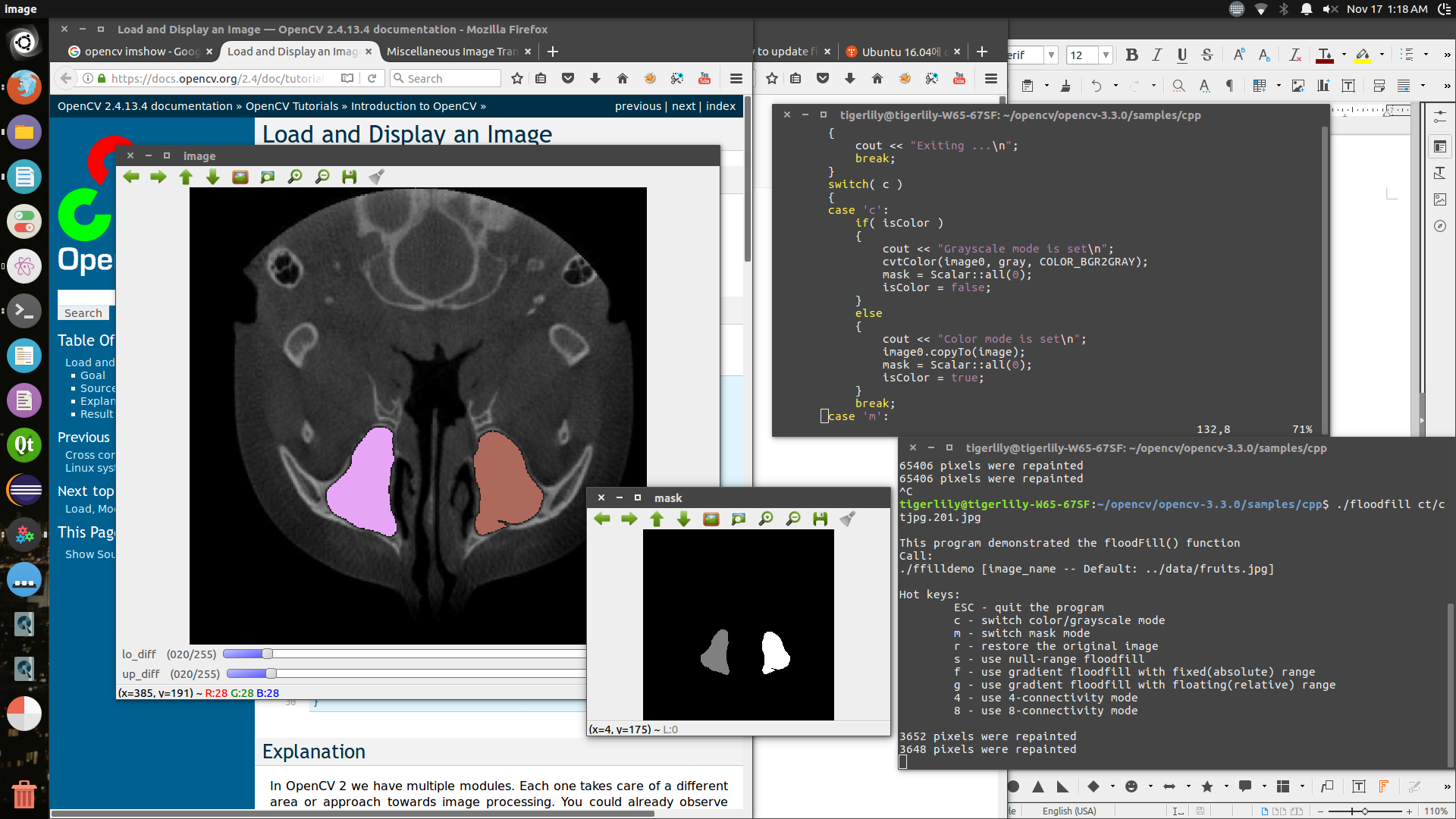
*frontier.push(q)* // then set neighbor to new color,

*I(q) ← new-color* // and add neighbor to frontier.

(Stan Birchfield, *Image Processing and Analysis, p.155 - 156*)

2. floodFill function in OpenCV

2 – A. 함수 개괄

  
Illustration 2

C++: int floodFill(InputOutputArray image, InputOutputArray mask, Point seedPoint, Scalar newVal, Rect\* rect=0, Scalar loDiff=Scalar(), Scalar upDiff=Scalar(), int flags=4 )

second argument “mask”

→ 가로 세로 2 픽셀씩 더 크다. image의 (x, y) 좌표는 mask의 (x+1, y+1) 좌표에 해당한다.

→ 클릭하여 얻은 부분만 따로 떼어 이미지화한 것. 검정 바탕에 영역 표시만 덩그러니 떠 있어서 마스크만 따로 떼는 것은 우리 이미지 프로세싱에는 도움이 안 될 듯함. (Illu 2 참고)

2 – B. 샘플 코드를 통한 이해

samples/cpp/ffilldemo.cpp와 상악동 단면 명확하게 나온 샘플 ct 사진 한 장 사용함.

- 실행 커맨드

>> g++ -o floodfill ffilldemo.cpp $(pkg-config --libs --cflags opencv)

>> ./floodfill ct/ctjpg.201.jpg

- ‘마스크’

코드 중 main함수 case ‘m’에서 imshow(“mask”, mask)를 imshow(“mask”, image) 로 바꿔 namedWindow에 mask 대신 image 띄우게 하면 원 이미지에 floodFilled된 부분까지 합쳐진 게 새창으로 뜨는 걸 확인할 수 있다.

- floodfill 결과는 Mat 타입에 저장

프로세스 하기 전 imported image file(예를 들어 내가 사용한 ct/ctjpg.201.jpg)을 imread 함수를 통해서 **Mat** type으로 로드한다. image, mask 모두 Mat 타입이므로 floodFill한 결과를 계속해서 연쇄적으로 변환하고 수정하기 용이할 것으로 보임. watershed 등 모든 프로세스에서 Mat 이용.

- 표시 영역 색깔

ffilldemo.cpp의 onMouse 함수에서 마우스 클릭 시 표시부분 색 결정하는 b, g, r은 theRNG() 때문에 클릭할 때마다 랜덤한 값이 assign된다. int b = 255; int g = 0; int r = 0; 으로 수정하면 시드 포인트 줄 때마다 영역이 파란색으로 표시된다. 향후 영역을 수정해나갈 때 일관성 있는 색깔이 필요하면 참고.

상악동 영역 내에서도 가장 가운데 부분은 R: 0, G: 0, B: 0 이지만 가장자리로 갈수록 R: 3, G: 3, B: 3 등 미약하지만 색깔 차이가 있기 때문에 up\_diff를 조정하면 검출되는 영역 넓이가 달라짐을 확인할 수 있다.