프로그램 순서

1. 기울어진 각도 계산 & 돌린 image 반환

A. 기울어진 각도 계산

a. pair<int,int> LineSegmentation::ret_middle_point_rough()

b. void LineSegmentation::get_skewed_angle(pair<int,int> mid)

B. 돌린 Image 반환

Void LineSegmentation::correct_skewed(pair<int,int> mid)

2. Line segmentation

A. find contours

Void LineSegmentation::find_contours()

B. 1차로 Line 나누기

void LineSegmentation::divide_contours_into_lines()

C. Line에 포함 안된 contour들 나누기

void LineSegmentation::include_not_matched()

D. Line 안에 bar가 있어서, 위/아래의 Line과 병합해야 하는 Line들 처리

void LineSegmentation::include_dividend()

E. 각 Line들의 구간 최종적으로 구하기

Void LineSegmentation::update_range()

F. Image print

---

1. 기울어진 각도 계산 & 돌린 image 반환

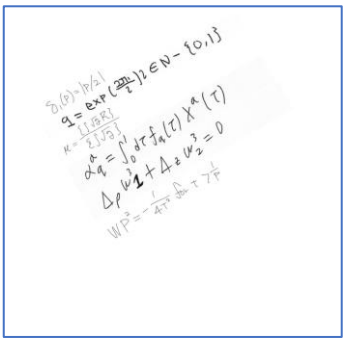
pair<cv::Mat,cv::Mat> ret_rotated_grey(int argc, char *argv[]);

반환값의 첫번째 element는 color image이고, 두번째 element는 grey image입니다.

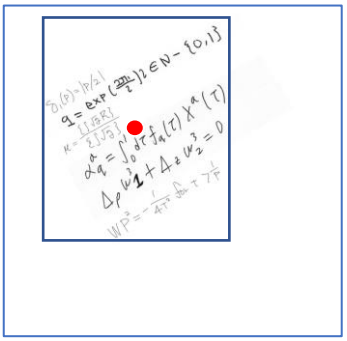
A. **기울어진 각도 계산**

a. pair<int,int> LineSegmentation::ret_middle_point_rough()

기울어진 수식 단락의 중심점의 좌표를 반환합니다.



이미지가 이렇게 주어졌을 때,



빨간 점의 좌표를 반환합니다

b. void LineSegmentation::get_skewed_angle(pair<int,int> mid)

Hough transformation & voting을 적용해서 수식 단락의

기울어진 각도를 계산합니다.

https://stackoverflow.com/questions/24046089/calculating-skew-of-text-opencv

코드 구현은 위 링크를 참고했던 것 같습니다.

B. **돌린 image 반환**

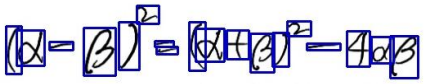
Void LineSegmentation::correct_skewed(pair<int,int> mid) 함수에서

Cv::getRotationMatrix2D함수를 사용해서 주어진 image를 회전시킵니다.

2. line segmentation

A. find contours

Void LineSegmentation::find_contours() 에서



이런 식으로 글자들의 bounding box를 구합니다

이 과정에서 글자들 Bounding box들의 평균 Height, Width를 구합니다.

또한 이 과정에서 각 Bounding box들이 분수 혹은 복소수 켤레를 표현할 때 사용되는 Bar인지 확인합니다. 그 기준은

1) 해당 Bounding box의 width가 height의 5배 이상 길고 & Bounding box들의 평균 width보다 1.3배 이상 길다

2) 해당 Bounding box의 width가 height의 3배 이상 길고 & Bounding box들의 평균 height보다 0.5배 이하로 짧다.

두 조건 중 하나의 조건 이상을 만족해야 합니다.

(bool TH_Rect::get_is_bar() 참고)

B. 1차로 Line 나누기

void LineSegmentation::divide_contours_into_lines()

Bounding box의 Height가 큰 Contour 먼저 처리를 합니다.

가장 큰 Height의 Bounding box로 최초의 Line을 하나 만듭니다.

그 이후에 Height가 큰 순서대로 Bounding box들을 각 Line으로 나눕니다.

기존에 존재하는 Line 중 좌표 구간 상 겹치는 구간이

가장 큰 Line의 index(max_line_idx 변수)와 해당 Bounding box의 Height에서 겹치는 구간의 비율(max_joint 변수)을 저장합니다.

Max_joint가 0.2 이상일 경우, 해당 Line에 Bounding box를 포함시킵니다.

0.2 이상으로 겹치는 Line이 없을 경우

1) 해당 Bounding box의 Height가 Bounding box들의 평균 Height의 85% 이상일 때에만 새로운 Line으로 나눕니다.

2) 85% 이하일 때에는, 새로운 Line으로 나누지 않고, LineSegmentatoin::not_matched vector에 저장합니다.

C. Line에 포함 안된 contour들 나누기

void LineSegmentation::include_not_matched()

LineSegmentatoin::not_matched vector에 저장된 Bounding box들을 기존에 있는 Line들에 포함시킵니다.

a)해당 bounding box 아래에 아무 Line도 없을 경우, 가장 아래에 있는 Line에 포함시킵니다.

b) 해당 bounding box 위에 아무 Line도 없을 경우, 가장 위에 있는 Line에 포함시킵니다.

c) 해당 bounding box가 bar인 경우

-> 분수의 bar인지 확인

해당 bounding box 위에 있는 Line까지의 거리 (above_dist)와 아래에 있는 Line까지의 거리(below_dist)의 비율과 두 거리 중 큰 거리(max_dist)와 bounding box의 평균 Height를 비교

(코드구현)

above_dst/below_dst)<1.5

&&

Max_dist<CONTOUR_AVG_HEIGHT_GLO(bounding box의 평균 Height)

1) 분수의 bar이면, 위의 Line과 아래 Line을 Merge.

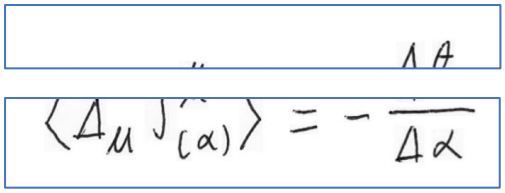
2) 아니면, 가까운 Line에 포함시킴

D. Line 안에 bar가 있어서, 위/아래의 Line과 병합해야 하는 Line들 처리

void LineSegmentation::include_dividend()

$$\langle \Delta_\mu J^\mu_{(\alpha)} \rangle = -\frac{\Delta\theta}{\Delta\alpha}$$

이렇게 segmentation해야 하는데,



이렇게 두 라인으로 나눴을 때 한 라인으로 merge해야 하므로 거치는 과정입니다.

Bar애서 다른 Line에 있는 글자까지 거리(other_dist)와 bar가 포함된 Line에서 y축 방향으로 가장 가까운 글자(mine_dist)까지의 거리를 기준으로 merge 여부를 판단합니다.

Mine_dist, other_dist를 구하고 나서,

bool ret_include_divide_or_not(int mine_dist,int other_dist,int mean_line_h) 함수를 이용해서 merge 여부를 판단합니다. 아래 1), 2) 중 하나의 조건을 만족하면 merge합니다.

1) Other_dist와 mine_dist가 bounding box의 평균 Height의 60%보다 작다

&

other_dist/mine_dist의 비율이 0.2보다 작다

2) Other_dist와 mine_dist가 bounding box의 평균 Height의 80%보다 작다

&

아래 a,b,c 중 하나의 조건을 만족할 때.

a)other_dist/mine_dist의 비율이 0.7~1.4 구간에 포함된다

b) mine_dist이 0이고, other_dist가 bounding box의 평균 Height의 60%보다 작다

c) other_dist이 0이고, mine_dist가 bounding box의 평균 Height의 60%보다 작다

E. 각 Line들의 구간 최종적으로 구하기

Void LineSegmentation::update_range()

각 Line들이 포함하고 있는 Bounding box들의 최소/최대 y값, 최소/최대 x값을 이용해서 Line의 구간을 update합니다.

F. Image print

Line Segmentation의 결과를 print합니다.