**Vectorized Floor Plan generation from image of Emergency Evacuation Plan**

Input source: images of Emergency Evacuation Plan

Expected Result: geospatial format (shp, geojson) of vectorized floor plan

GitHub: <https://github.com/chungkang/L5IN_task2>

**Steps of logics**

1. **Take photo of Emergency Evacuation Plan**

Better to have well projected, high resolution and low glare/reflected images

|  |  |
| --- | --- |
| 텍스트, 평면도, 그림, 지도이(가) 표시된 사진  자동 생성된 설명 |  |
| well projected | With reflection and projected from side view |

1. **Reproject (Rectify) image**

If is not well projected, reproject(rectify) image

Compute Vanishing points using RANSAC and rectify the image

Reference: Image-Rectification of chsasank from github repository

<https://github.com/chsasank/Image-Rectification>

based on the paper: Chaudhury, Krishnendu, Stephen DiVerdi, and Sergey Ioffe. "Auto-rectification of user photos." 2014 IEEE International Conference on Image Processing (ICIP). IEEE, 2014.

<https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/42532.pdf>

|  |  |
| --- | --- |
| 텍스트, 화이트보드, 직사각형, 평행이(가) 표시된 사진  자동 생성된 설명 | 텍스트, 직사각형, 지도, 문구용품이(가) 표시된 사진  자동 생성된 설명 |
| Projected from side view | Reprojected/rectified image |

crop the part which is needed

1. **Filter the image to reduce noise**

Reduce noise from image with **bilateral filter** of OpenCV

|  |  |
| --- | --- |
|  |  |
| Before filtering noise | After filtering noise |

Adjust following parameters for preserving the edges of an image while reducing noise

Diameter: The diameter of each pixel neighborhood used during filtering. It controls the size of the neighborhood. Larger values o result in stronger smoothing.

sigmaColor: The standard deviation of the color space. It controls how different colors are considered to be neighboring pixels. Larger values of sigmaColor result in more colors being included in the filtering process.

sigmaSpace: The standard deviation of the coordinate space. It controls the spatial extent of the filter. Larger values of sigmaSpace result in pixels farther apart in the spatial domain being considered as neighbors.

1. **Remove symbols from image with Feature matching**

Reduce symbols on emergency evacuation plan

Feature matching method of OpenCV

Feature matching steps:

Clustering: MEANSHIFT => divide image

Feature Extraction: SIFT => extract feature points from divided image

Feature Matching: FLANN => match feature points between image and symbol

Homography calculation: RANSAC => calculate offset, rotation, scale of symbol on image

Symbols should be cropped from each image

|  |  |
| --- | --- |
|  |  |
| Before removing symbols | After removing symbols |

그래픽, 상징, 카민, 그래픽 디자인이(가) 표시된 사진

자동 생성된 설명

Cropped symbols from image

Adjust following parameters for reducing symbols from the image

MIN\_MATCH\_COUNT: This parameter determines the minimum number of matches required for a template image to be considered a valid match. For perspective Transform, more than 4 points are needed.

MATCH\_DISTANCE: This parameter is used in the Lowe's ratio test during feature matching. It controls the threshold for accepting matches based on their distance ratio.

NUMBER\_OF\_TEMPLATES: This parameter specifies the number of template images to be processed.

PAD: This parameter defines the padding value used when generating the destination points for perspective transformation.

Reference: <https://stackoverflow.com/questions/42938149/opencv-feature-matching-multiple-objects?noredirect=1&lq=1>

1. **Stitch images**

If it is not covering whole floor, stitch images to make single image to make single floor

Should be stitched one by one

|  |  |
| --- | --- |
| 텍스트, 평면도, 도표, 개략도이(가) 표시된 사진  자동 생성된 설명 | 스케치, 지도, 그림, 평면도이(가) 표시된 사진  자동 생성된 설명 |
| 3OG part1 | 3OG part2 |
| 도표, 평면도, 스크린샷, 라인이(가) 표시된 사진  자동 생성된 설명 | 텍스트, 스케치, 평면도, 도표이(가) 표시된 사진  자동 생성된 설명 |
| 3OG part 3 | 3OG part1 + part2 |

평면도, 도표, 라인, 개략도이(가) 표시된 사진

자동 생성된 설명

3OG full floor

Reference: <https://gist.github.com/tigercosmos/90a5664a3b698dc9a4c72bc0fcbd21f4>

1. **Extract contours of wall**

MIN\_AREA: This parameter defines the minimum contour area required for an object to be considered valid. Contours with an area smaller than this threshold will be filtered out.

BINARY\_THRESHOLD: This parameter sets the threshold value for creating a binary image from the grayscale image. Pixels with intensity values below this threshold will be set to 0, and pixels above or equal to the threshold will be set to 255.

APPROX\_CONTOUR: This parameter controls the level of approximation when converting contours using the cv2.approxPolyDP() function. It specifies the percentage of the contour perimeter that should be approximated.

Logic steps: Binarize -> extract contour -> straighten contour -> save as geojson

|  |  |
| --- | --- |
| 스케치, 그림, 도표, 평면도이(가) 표시된 사진  자동 생성된 설명 | 평면도, 도표, 지도, 텍스트이(가) 표시된 사진  자동 생성된 설명 |
| Binarize | Extract contour |
| 라인, 도표, 평행, 그래프이(가) 표시된 사진  자동 생성된 설명 | 지도, 그림, 평면도, 도표이(가) 표시된 사진  자동 생성된 설명 |
| Straighten contour | Geojson format of contour |

1. **Georeference with QGIS**

With QGIS, vector file can be georeferenced with [Layer-Georeferencer] function.

Since QGIS version 3.26, QGIS provides vector georeferencer function also for vector format.

지도, 평면도, 텍스트, 도표이(가) 표시된 사진

자동 생성된 설명

Georeferenced vector data on OpenStreetMap

Among transformation algorithms of QGIS, Thin Plate Spline (TPS) algorithm has been chosen, which provided better results.

<https://docs.qgis.org/3.28/en/docs/user_manual/working_with_raster/georeferencer.html#id7>

1. **Detect doors**

Possible solution 1: define door with size of area

<https://stackoverflow.com/questions/55356251/how-to-detect-doors-and-windows-from-a-floor-plan-image>

Possible solution 2: Detect the center of arc part

[**https://stackoverflow.com/questions/62804419/detecting-the-center-of-an-arc-by-using-open-cv**](https://stackoverflow.com/questions/62804419/detecting-the-center-of-an-arc-by-using-open-cv)

[**https://stackoverflow.com/questions/59099931/how-to-find-different-centers-of-various-arcs-in-a-almost-circular-hole-using-op**](https://stackoverflow.com/questions/59099931/how-to-find-different-centers-of-various-arcs-in-a-almost-circular-hole-using-op)

Possible solution 3: detect ovelapping area of hallway and room

1. Unsolved problems

Symbol removal is not perfectly performed

Glare/Reflection correction didn’t applied

Adjacent object along wall affects contour detection

Door detection solution didn’t perorm