Vision

For a simpler version first, we just take a picture of the whiteboard as the input image.

A screenshot of a cell phone

Description automatically generatedA close up of text on a whiteboard

Description automatically generatedA close up of a map

Description automatically generated

1.“Clean” the image

After we get the raw image data from camera, a cleaner version of the whiteboard picture binary image is needed. On the one hand, we should filter the noise in the raw image. This is really useful because light variation and possible shades in the image will affect the image processing behavior in the following process. We deploy parts of the ImageMagick to “clean” this raw image tremendously. On the other hand, we want clear contrast between the whiteboard white background and the writing’s on the whiteboard.

A close up of a logo

Description automatically generatedA close up of text on a white background

Description automatically generated

2.Image binarization

As for the image binarization, we have tried different method on different pictures and found different suitability. Mainly we’ll use global thresholding and adaptive gaussian thresholding in turning image into binary.

we tried to use a global value as threshold value. But it may not be good in all the conditions where image has different lighting conditions in different areas. In that case, we go for adaptive thresholding. In this, the algorithm calculate the threshold for a small regions of the image. So we get different thresholds for different regions of the same image and it gives us better results for images with varying illumination. There are two main parameters in adaptive thresholding.

Adaptive Method - It decides how thresholding value is calculated.

Block Size - It decides the size of neighborhood area.

From the test figures above, we see the second image has a better performance in adaptive gaussian thresholding than global thresholding.

3. Determine the clean area based on sparsity of data

In terms of cleaning, we have this largest-area first strategy. Because our aim of using vision in cleaning is to detect where to clean and save time and energy. In fact, we will frequently encounter cases when whiteboard has lot of white space, in another word, sparse data. Thus our top priority is to neglect this largest clean area in terms of efficiency. Also in practical, the areas needed to be cleaned is mostly the whole whiteboard space subtracted by this largest clean area.

A screenshot of a cell phone

Description automatically generated

The blue box defines the largest area that is excluded from cleaning, whereas the rest while be cleaned in an easy model. (Largest-area Excluded Model)

However, when we scale down the window size and want more accuracy than time-efficiency, we get more exact boundaries of the clean area. As before, the area needed to be cleaned is the subtraction.

A screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generated

Left one is when window size drop to ¼ and right one is 1/8. We can see the clear boundary is shown to us, we can choose different method based on our motor accuracy and the overall cost-efficiency. (Smaller window size filtering)

We gradually narrow down the whiteboard space to the most part in need of cleaning.

4.bounding box segmentation

This step we will segment the cleaning area and design bounding boxes that cover the entire cleaning area. We set the default number of boxes that binds the areas to be 2-5. The boxes are computed out of the center of mass of different areas. When we finish segmenting the area into different bounding boxes (clusters/blocks), the cleaning task is operated in these bounding boxes.

A screenshot of a cell phone

Description automatically generated

5.Return data to robot control center

We could return either return the bounding boxes we get from step 4 or the boundaries of the clean data from step 3 for the cleaning task. Thus, we can send the coordinates of these to our robot control center and the hardware will do the following process afterwards.

**Extension**

As is illustrated, the raw data we start with is the whiteboard image we capture from camera. Not only is vision needed in reducing noise and detecting cleaning area to start with, but we also want to monitor the cleaning behavior of the cleaning robot. Thus we’ll implement the video-recording and tracking system further. Since the cleaning robot is cleaning one box at a time, we can set the rule for clean next box to be after fully completing current box. In this case, the cleaning robot will move on to next box only if current one is cleaned. We believe this will improve the cleaning efficiency and avoid moving backwards to areas that have not been entirely cleaned.

**Reference**

1.Image Processing in OpenCV¶. (n.d.). Retrieved from https://opencv-python-tutroals.readthedocs.io/en/latest/py\_tutorials/py\_imgproc/py\_table\_of\_contents\_imgproc/py\_table\_of\_contents\_imgproc.html#py-table-of-content-imgproc