Visualized Sanitation Evaluator in Microenvironment

Yu Wu 60185540
Shuo Sun 82350468

December 30, 2019

1 Introduction

Nowadays, people's requirements on the quality of life are increasingly improved, and they are more and more accustomed to using various machines to help clean the environment and objects around them. However, the corresponding experience of effective cleaning is insufficient (not intuitive, no knowledge). For example, after using germicidal instruments and washing machines, the users cannot see the cleanness effect of the product. At the same time, for purifiers and other products that need filter meshes, the users cannot easily know when the filter elements need to be replaced only by recognizing with their eyes.

According to those situations, we designed a module which can be installed in different devices, offering a clear view on the cleanness effect of them. We have applied computer vision technology on fluorescence reaction detection under ultraviolet ray (as figure) and identification of dust, mites and bacteria under a microscope to show the contrast before and after cleaning, which intuitively informs the customers of the cleanness effect.

Until now, we made a good performance on fluorescence reaction detection under ultraviolet ray in real cases with our detection algorithm based on gradient and color. Also we did some research and experiments on microscope photos segmentation and classification through different methods and achieved some results.

Note: The project is not purely on software-level. The first part contains some hardware support by teammates not in the class and we two are responsible for the CV engineering part. Thanks to other team members who are not in CS172 course offering the product requirement, market research, mechanical structure design, user interface and so on.

2 Methods

2.1 Workflow

Picture below is the workflow of our module. It mainly includes the client part and the server part. What we focus on is the server part, especially the detection and classification part.

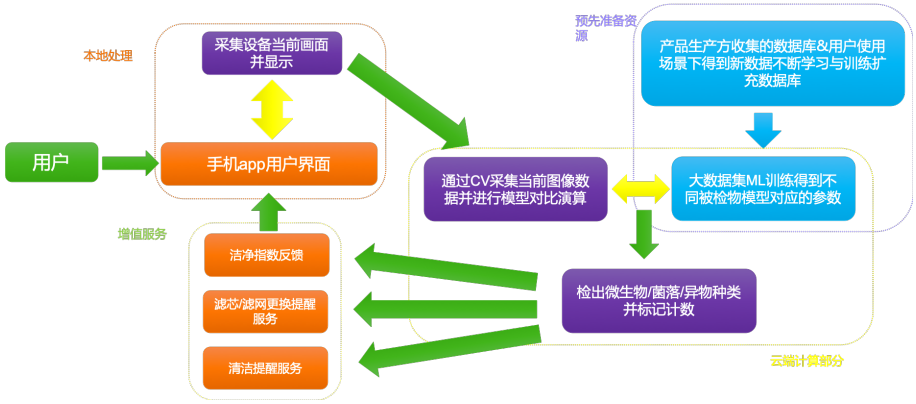

Figure 1: Workflow.

And those are the structure of the machine. We currently focus on the air quality evaluation in the air-conditioners. The system is controlled by the microcomputer, which is one subpart of the client part. The dirt is sampled using the transparent adhesive tapes and the rough surfaces of the snap fasteners. Therefore, we don't need to consider the complexity of different surfaces. And afterward, the pictures collected by the camera and microscope will be sent to the server and processed by our algorithms.

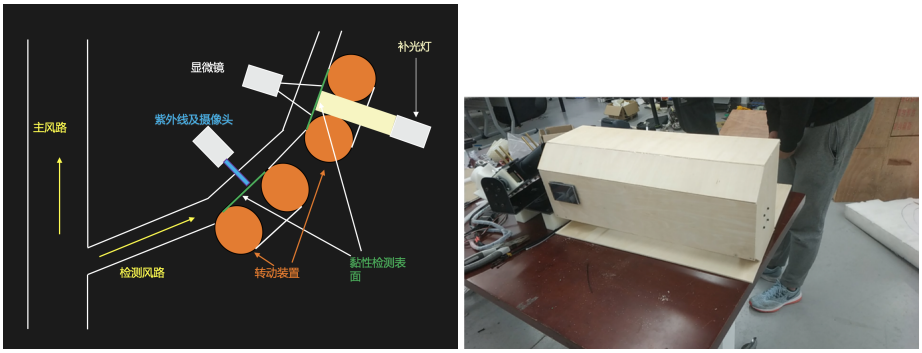

Figure 2: *Left:* Structure inside the air-conditioner demo. *Right:* The air-conditioner demo with our module installed.

2.2 Fluorescence Reaction Detection

Biology Theory

Picture below is the known material that will cause fluorescence reaction under ultraviolet ray. We can see them under ultraviolet rays with naked eyes. So it can simply be detected by a camera.

紫外	
生物污染物 如排泄物、分泌物等，是细菌繁殖及异味的来源	√
油脂类污渍 油渍/污渍等	√
油脂类、酚酞类物质	√
金属有机络合物	√
化学可燃物	√
荧光增白剂	√
含有荧光剂的漂浮纤维	√
苯并氧氮型物质	√

Figure 3: Known Fluorescence Reaction.

Detection Algorithm

The detection algorithm based on gradients and colors to find suspect pixels and then use a cluster to make near pixels into group. The algorithm is run in cloud and we use the module as camera and result receiver. The function details are in script *microbio.py*.

- Color Extraction
  - Each pixel has RGB values $\vec{p} = (r, g, b)$.
  - Use Gaussian filter to smooth the noises.
  - Several models to combine (can be trained):
    - $b > w_1 * r + w_2 * g$
    - $\|\vec{p} - \vec{p_0}\|_2 < \epsilon$
    - $b > b_0 \& r < r_0 \& g < g_0$
    - $grad(\vec{p}) > \delta$
    - ...
- Cluster
  - Optimize

$$L = \sum_{j=1}^{k} \sum_{i=1}^{n} \|x_i - \mu_j\|_2 \, |(t_i = j)$$

.

  - To simplify, choose distant centers but do not iterate.

- Non-maximum Suppression

2.3 Microscope Photos Segmentation and Classification

Datasets

At the beginning, we use tape to make some sample on the different dirty material surfaces aim to build up a data set and make a model to classify them. However we find those pictures hard to label due to we don't know which parts are the dirt and what types they belong to.

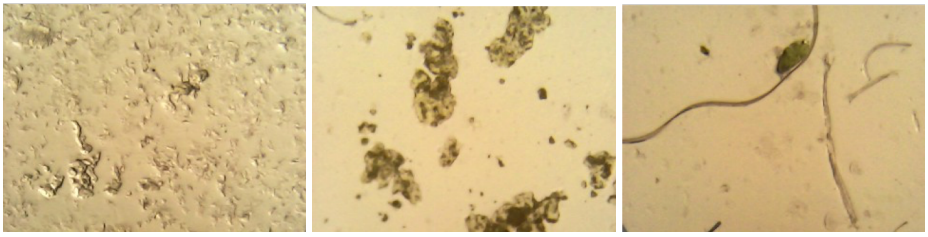

Figure 4: Different samples on some different surfaces under the microscope.

We temporarily put it aside and read some research papers on microscope pictures. We found that bacteria colony photos are quite similar to the sample we made except from they are colored and very big size pictures. We can use these pictures to train a model that may be suitable for our samples.

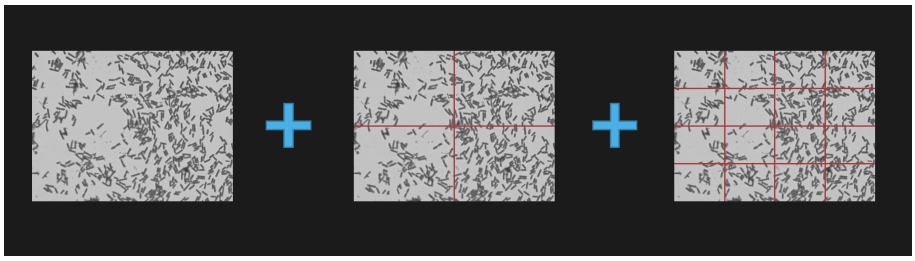

Figure 5: Grayed and divided photos as training set.

Therefore, we use the dataset DIBaS of bacteria colony photos [2] and cut them into small pieces of 256×191 to build our training set, in the different scales, as figure 5 shows. We make training sets in different sizes from categories of 7 running our script *split.py*, as table 1 shows. Those categories are the most familiar ones in our daily lives. And we made up some testing sets by our script *generator.py* to put bacteria with different species on a same picture, as figure 6 shows. It is worth mentioning that type 6 in the testing set comes from their different subspecies photos, considering they are the most common among all 7 species.

Index	Species
1	Bacteroides fragilis
2	Clostridium perfringens
3	Escherichia coli
4	Listeria monocytogenes
5	Proteus spp.
6	Staphylococcus spp.
7	Streptococcus agalactiae

Table 1: Included bacteria species.

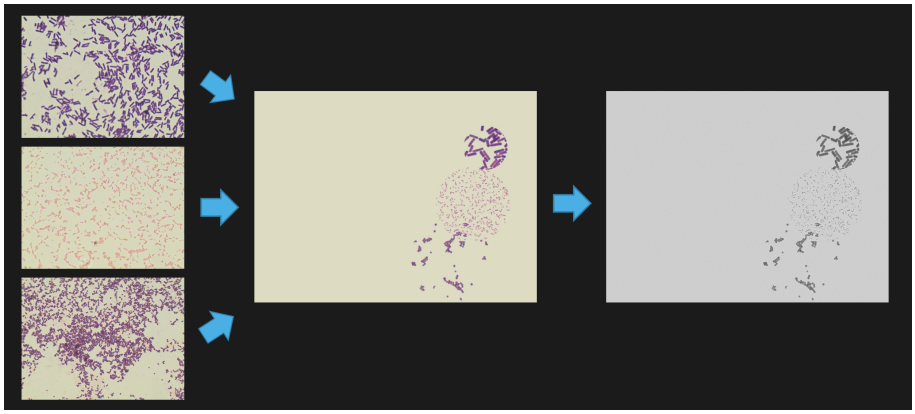

Figure 6: Test set photos.

Segmentation

We detect the suspected bacteria as before. To segment the different bacteria firstly using mean shift [5] algorithm on the test picture, as figure 7 shows.

Figure 7: Meanshift.

Classification

We carry out classification on the small areas that are in the marker rectangle, which will output the most alike result. Our main function is in *main.py*, to process the testing set.

We have used five different methods:

- **AlexNet:**
  We begin with a classical CNN, stated in [7]. Fortunately, PyTorch offers prepared model, and we train it by our training set.
- **SPPNet:**
  According to the feature of our training set that it includes multiple scales of the bacteria colonies, we choose to apply spatial pyramid pooling layer in our network [8].
- **SIFT + FC:**
  Inspired by the paper's great outcomes of the classification on the dataset DIBaS [2] but limited by the time to figure out its models, we choose an alternative to try SIFT feature and classify with fully connected layers. SIFT implement in PyTorch is from the work [10].
- **SIFT + FV + SVM:**
  Rather than using the complex source code provided by the paper [2], we implement one of the models in MATLAB, with the help of VLFeat library [11].

- **SURF + BoW + SVM:**
  Finally we try a simple bag-of-visual-word model [9] provided by MATLAB, because it is a robust model based on feature extraction, like what the paper [2] does.

3 Results & Discussions

3.1 Detection

It truly makes sense that we can detect different quantities of dirt in the photos of clean and dirty surfaces.

Figure 8: Result of detection.

3.2 Segmentation and Classification

Method	Accuracy	Average Loss	Size of Training Set
AlexNet	0.600	-4.4903	112000
SPPNet	0.775	0.5440	112000
SIFT + FC	0.325	-3.4583	112000
SURF + BoW + SVM	0.830	None	3500
SIFT + Fisher Vector + SVM	0.166	None	22400

Table 2: Performance of different methods.

We applied both SPPNet and BoW models to classify our segmented colonies.

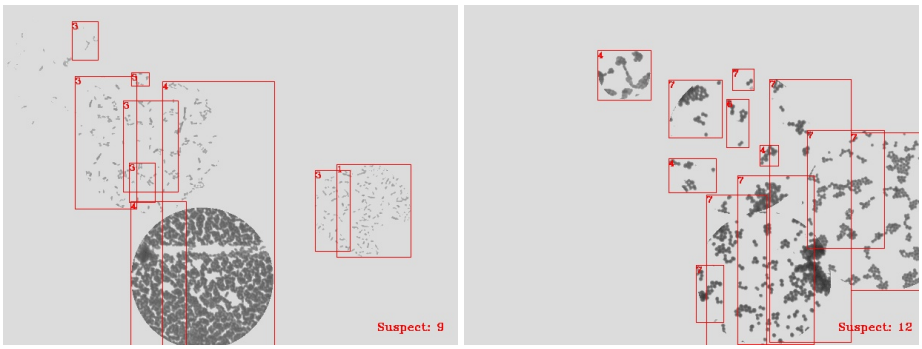

Figure 9: Example results with type numbers in different areas.

Filename	Accuracy	Detected Type 1	Detected Type 2	Detected Type 3	Detected Type 4	Detected Type 5	Detected Type 6	Detected Type 7
6_4_4.jpg	0.333333	None	None	None	0	None	1	None
6_6_5.jpg	0.666667	None	None	None	None	0	1	None
6_5_4_1_3.jpg	0.2	0	None	0	0	0	1	None
6_5.jpg	0.5	None	None	None	None	0	1	None
6_1_6.jpg	0.666667	0	None	None	None	None	1	None
3_7_7_2.jpg	0.75	None	1	0	None	None	None	1
7_3_7_6.jpg	0.25	None	None	0	None	None	1	0
4_6_2_6.jpg	0.25	None	0	None	1	None	0	None
1_2.jpg	1	1	1	None	None	None	None	None
6_3_2_5.jpg	0.5	None	1	0	None	1	0	None
6_3_6_1.jpg	0.5	0	None	0	None	None	1	None
3_6_6.jpg	0.666667	None	None	0	None	None	1	None
6_2_1_7_6.jpg	1	1	1	None	None	None	1	1
4_1_2_1.jpg	0.75	1	1	None	0	None	None	None
2_6.jpg	0.5	None	1	None	None	None	0	None
1_6_2.jpg	0.666667	1	0	None	None	None	1	None
4_2_5_6.jpg	0.25	None	1	None	0	0	0	None
6_7_2_6_3.jpg	0.6	None	1	0	None	None	1	0
6_3_2_7_6.jpg	0.8	None	1	0	None	None	1	1
4_2_4_1.jpg	0.75	0	1	None	1	None	None	None
2_2.jpg	1	None	1	None	None	None	None	None
1_2.jpg	1	1	1	None	None	None	None	None
7_4_6_5.jpg	0.75	None	None	None	1	1	1	0
4_5_5_5.jpg	0.25	None	None	None	1	0	None	None
5_2_3.jpg	0.666667	None	1	0	None	1	None	None
4_2_3_1_1.jpg	0.4	1	0	0	0	None	None	None
6_5_3_6_7.jpg	0.8	None	None	0	None	1	1	1
7_6.jpg	1	None	None	None	None	None	1	1
7_2_3_6_6.jpg	0.6	None	1	0	None	None	1	0
3_6.jpg	0.5	None	None	0	None	None	1	None

Figure 10: Test performance table for BoW.

Average Accuracy	Type 1	Type 2	Type 3	Type 4	Type 5	Type 6	Type 7
0.619	0.600	0.824	0.000	0.444	0.444	0.810	0.556

Table 3: The BoW model performance.

Filename	Accuracy	Detected Type 1	Detected Type 2	Detected Type 3	Detected Type 4	Detected Type 5	Detected Type 6	Detected Type 7
6_6_1_2_2.jpg	0.4	0	1	None	None	None	0	None
5_3_7.jpg	0.333333333	None	None	1	None	0	None	0
4_6_7_6.jpg	1	None	None	None	1	None	1	1
2_2_7_6.jpg	0.75	None	1	None	None	None	0	1
3_3_1_4.jpg	1	1	None	1	1	None	None	None
7_4_6_5.jpg	0.5	None	None	None	1	0	0	1
5_6_7.jpg	0.666666667	None	None	None	None	1	0	1
6_1_6.jpg	0.333333333	1	None	None	None	None	0	None
4_6_2_3_2.jpg	0.8	None	1	1	1	None	0	None
6_5_6_4_6.jpg	0.4	None	None	None	1	1	0	None

Figure 11: Test performance table for SPPNet.

Average Accuracy	Type 1	Type 2	Type 3	Type 4	Type 5	Type 6	Type 7
0.618	0.667	1.000	1.000	1.000	0.500	0.125	0.800

Table 4: The SPPNet model performance.

3.3 Discussions

- We found that the traditional descriptor methods perform well in these bacteria colony pictures. The background of these pictures are under control when the photos are taken so these bacteria colony are strongly distinct from the background. Thus we can get lots of SIFT or SURF descriptors which may help describe each kind of bacteria colony more detailedly and improve classification performance.
- CNNs are good, but they need more data to train, as we find a significant improvement from 0.14 to 0.60 of the accuracy of AlexNet when we grow our training set.
- Some types are easy to be classified, while others may be not. That is probably because some of them are too distinct from others. Methods need to be improve to distinguish those similar types such like type 1, type 3 and type 5.
- It is obvious that our mean shift segmentation algorithm needs to be improved. In simulation, we found that for those ”photoshoped” pictures, it sometimes put two close but different colonies together into one group. However, currently our classifier can only map one picture to one class, so it will make the second colony not classified.
- Our dataset is direct from cutting original photos, which brings lots of useless data such as photos with no bacteria. The dataset should be precise, meaning that we should process them again after cutting.
- Fisher vector should be an excellent approach. We doubt that our implement has something wrong, needed to be fixed in the future or learned from its source code.
- Theory is different from the reality. We have to found realistic methods to collect those bacteria in the daily life anyway.

References

[1] Carlos Xavier Hernandez, Mohammad M. Sultan. Using Deep Learning for Segmentation and Counting within Microscopy Data, 2017.

[2] B. Zieliński, A. Plichta, K. Misztal, P. Spurek, M. Brzychczy-Włoch, and D. Ochońska. Deep learning approach to bacterial colony classification, PLOS ONE, 12(9), 1-14, 2017.

[3] Lowe D G. Object recognition from local scale-invariant features[C]//iccv. 1999, 99(2): 1150-1157.

[4] F. Perronnin and C. Dance. Fisher kenrels on visual vocabularies for image categorizaton. In Proc. CVPR, 2006.

[5] Comaniciu D, Meer P. Mean shift: A robust approach toward feature space analysis[J]. IEEE Transactions on Pattern Analysis & Machine Intelligence, 2002 (5): 603-619.

[6] Allwein, E., R. Schapire, and Y. Singer. Reducing multiclass to binary: A unifying approach for margin classifiers. Journal of Machine Learning Research. Vol. 1, 2000, pp. 113–141.

[7] Krizhevsky A, Sutskever I, Hinton G E. Imagenet classification with deep convolutional neural networks[C]//Advances in neural information processing systems. 2012: 1097-1105.

[8] He K, Zhang X, Ren S, et al. Spatial pyramid pooling in deep convolutional networks for visual recognition[J]. IEEE transactions on pattern analysis and machine intelligence, 2015, 37(9): 1904-1916.

[9] Csurka, G., C. R. Dance, L. Fan, J. Willamowski, and C. Bray. Visual Categorization with Bags of Keypoints. Workshop on Statistical Learning in Computer Vision. ECCV 1 (1–22), 1–2.

[10] Dmytro Mishkin, Filip Radenovic, Jiri Matas. Repeatability Is Not Enough: Learning Affine Regions via Discriminability. Proceedings of ECCV, 2018.9.

[11] A. Vedaldi and B. Fulkerson. VLFeat: An Open and Portable Library of Computer Vision Algorithms. http://www.vlfeat.org/, 2008