NUS CS4243: Computer Vision and Pattern Recognition November 12, 2019

Here is Waldo

Lecturer: Angela Yao *Student(s): Liu Zhaoyu, Fong Wei Zheng, Yan Yicheng*

Abstract

The report describes the algorithm used to find Waldo, Wenda and Wizard, based on famous British puzzle book "Where's Waldo". We begin with a basic literature review on various existing methods currently used in the industry to solve this problem, followed by our proposed solution: a cascaded method involving haar detection, support vector machines, and feature matching using SIFT descriptors. In this report, we also describe and recorded down other methods we tried before arriving at our proposed solution. The report also presents a qualitative and quantitative view of the various detection results obtained, and analyzes them, along with the strengths and weaknesses of our proposed solution.

1 Introduction

1.1 Problem statements

"Where's Waldo?" Is a British puzzle book created by English illustrator Martin Handford. Readers are challenged to find Waldo and other characters in a series of images consisting of hundreds of and thousands of people. This project aims to implement an algorithm to find Waldo, Wizard and wizard in the given images with high speed and accuracy.

1.2 Definitions and challenges

The proposed solution will generate bounding box predictions on various images that might/might not contain Waldo, Wenda and Wizard. These bounding boxes will be stored in the VOC format, and evaluated based on their mean average precision (mAP). In this problem, we are also restricted to using non-deep computer vision techniques, and will try to perform better than the presented baseline mAP score on the validation set.

1.3 Existing methods and downsides

- Finding waldo with Wolframe Mathematica. The process is filtering out non-red pixels and non-striped pattern. The downside of the algorithm is that this method is not workable in images filled with red and white stripped pattern.
- Optimal path search. The problem is broken down using "traveling salesman problem" and combined with a "genetic algorithm" to improve the hunting speed. Even though this method finds waldo in less than 10 seconds for most images, the downside is when applying this method on an outlier image, it is time-consuming to follow the certain path, and also disoriented trying to get back to the path.

1.4 Proposed Approach

We propose implementing an algorithm combining three methods:

- We trained haar cascade classifiers for the three characters, and got first round of detection, which consisted of a few false positives.
- Then we proposed to use SVMs trained on HOG features, after passing through a certain threshold, most false positives will be eliminated.
- We compared the remaining detected images against templates using SIFT feature matching, after adjusting the threshold, we get the final detection.

In combining the three methods, we have managed to obtain a mAP score of 0.602 compared to the baseline mAP score of 0.575.

2 Proposed Solution

Our solution involves cascading the input images through 3 different layers of computer vision techniques, as shown in Fig. 1 below.

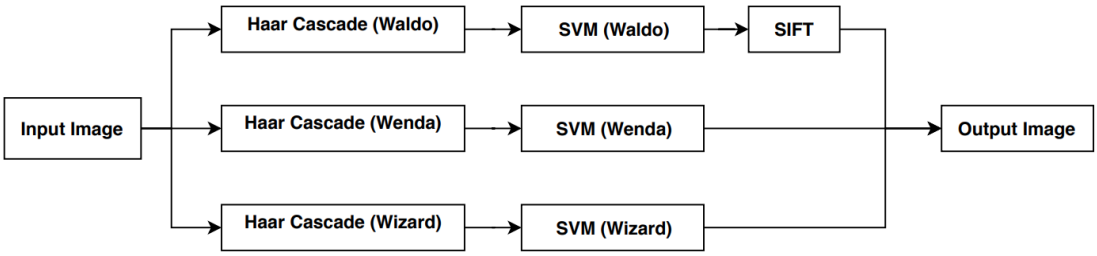


Figure 1: Our proposed solution.

The first layer consists of three Haar Cascade classifiers, which are used to detect Waldo, Wenda, and Wizard separately. Each classifier has two Haar cascade models, which are used to detect the character in different shapes (e.g. rectangle shape, square shape). This layer generates a list of bounding boxes across the entire image, which is sent to the next layer.

The second layer utilizes a linear Support Vector Machine (SVM) trained on histogram of oriented gradients (HOG) features. Small image patches based on the bounding boxes from the previous layer are first obtained, followed by computing their HOG features. These features are passed to the SVM returns a probability denoting the likelihood of the patch having the person of interest (Waldo, Wenda or Wizard). Using the probability values of image patches that pass a certain threshold, a final threshold is calculated. Image patches that pass the final threshold and are large enough are passed to the third layer.

The third layer uses SIFT-based feature matching. Only images that are considered large are passed into this layer, and are filtered once more based on a certain threshold.

3 Experiments

Before arriving at our final proposed framework, we tested various computer vision techniques on Waldo detection in order to gauge their viability. Once we found a suitable framework that did sufficiently well on Waldo detection, we applied similar frameworks towards Wenda and Wizard detection. In this section, individual experiments using haar cascade, SVM with sliding window detection, and SIFT matching are discussed.

3.1 Data Preparation and Configuration

3.1.1 Data collection

We firstly cropped all Waldo, Wenda, and Wizard images from the datasets provided and saved them into three folders according to their names. We also searched online and found more images to enlarge our data.

3.1.2 Data processing

After we collected images for Waldo, Wenda, and Wizard, we found the dataset for training is still very small. Therefore, we decided to use a method provided in Haar Cascade to create more training dataset, including both negative and positive.

1. Collection of negative dataset: We randomly cropped images of size 256*256 from the datasets and saved them as negative images (or called background images).

2. Generation of positive dataset: Since the original size of positive dataset was very limited, in order to have larger positive training data, we decided to generate positive data by randomly pasting the template images onto negative images (background images), and stored their locations in background images. This was achieved by a function in opencv called opencv_createsamples, which took in a list of background images and positive templates to generate the desired number of positive images. Examples for generated images can be seen in Fig. 2.

   Table 1 gives a summary of datasets after data processing.

Dataset	#positive	#negative	#Category
Waldo-rectangle	4,000	2,000	2
Waldo-square	4,000	2,000	2
Wenda-rectangle	2,000	1,000	2
Wenda-square	2,000	1,000	2
Wizard-rectangle	2,000	1,000	2
Wizard-square	2,000	1,000	2

Table 1: Summary of datasets after data processing.

3. Remarks: For each character, we divided the templates into two categories: square shape and rectangle shape, and generated positive images separately. The reason was that each

Haar Cascade classifier was trained to detect a fixed window shape. In order to support multi-shape detection, we decided to generate both rectangle and square positive images. Examples are given in Fig. 2, LHS is rectangle shape, while RHS is square shape.

Figure 2: Examples of the generated positive images.

3.2 Implementation

3.2.1 Haar Cascade model training

Theory and Rationale
Using information gathered from [2], we learned that a cascade classifier consists of a collection of stages, where each stage is an ensemble of weak learners. The weak learners are basically classifier called decision stump, which is a one-level decision tree. For each stage, the classifier will label the region at current location of the sliding window as positive or negative. If the location is labeled as negative, means no objects were found in this area, and the detector will slide the window to the next location. If the location is labeled positive, the classifier will pass this region to the next stage. The window area that is classified as positive in the final stage will be reported as an object founds by the detector.

The stages are designed to reject negative samples as fast as possible, and it is good at dealing with cases when the size of true positives is small. In order to work well, each stage needs to have a low false negative rate. During the training phase, if a stage labels an object as negative wrongly, the classification stops. However, each stage can have a high false positive rate, which means even if an object is labeled incorrectly, it can be corrected in the following stages. With more stages being trained, the overall false positive rate will reduce, however, the overall true positive rate will drop at the same time.

Cascade classifier training needs positive and negative images. The regions of interest used as positive samples are needed to be specified in the positive images. And in order to achieve acceptable detector accuracy, we need also tune the number of stages, feature type, and other function parameters.

Training process
After generating the training data, we trained Haar Cascade models for three characters separately. For each character, we trained two Haar Cascade models with two different window

shapes, one was 1:1 (square window shape), the other was 1:2 (rectangle window shape). The reason why we used two models for each character is that each Haar Cascade model can only detect bounding boxes with a fixed shape, in order to detect with multi-shape bounding boxes, we trained two models for each character.

We used the opencv method "opencv_traincascade" to train around 25 stages for each model, then chose the stage with the best performance. The maximal desired false alarm rate for each stage of the classifier we chose is 0.5.

3.2.2 SVM model training

The SVM models were trained on HOG features. Using the templates generated during data preparation, each template was first resized into a small image patch, then the skimage function "skimage.feature.hog" was used to generate the HOG features. The HOG function computes the row and column gradients of the input grayscale image patch, and stores them in gradient bins, thus forming a final gradient histogram output. The histograms formed from this function will then be flattened into a 1D array to represent the feature vector for this image patch. The SVM model was tuned by varying the number of orientation bins, pixels/cell, and cells/block during generation of the HOG features. Since a linear SVM is being trained, each image patch was labelled either as the target or vice versa.

The SVM has been configured to return a confidence score based on the labels: 'target' and 'not target'. During testing, we found that it was difficult to stick to one threshold for all validation images. We postulated that this was due to a lack of training data of target characters in different backgrounds. As such, we recalculated the threshold based on the initial results. We identified all the confidence levels that pass a pre-defined base threshold, then considered the top x% of these values as accurate detection. When implementing the recalculation methods, we also found that the SVMs for Waldo, Wenda and Wizard worked well with different threshold levels.

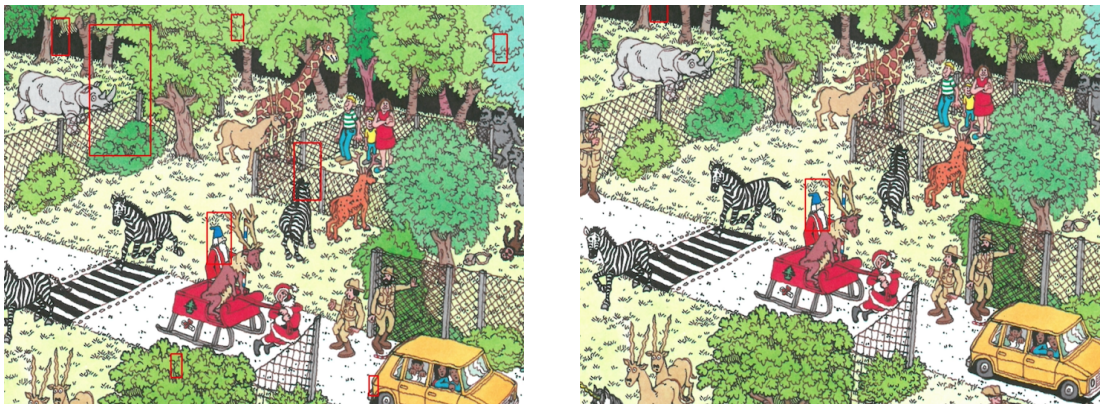
3.2.3 SIFT matching

The SIFT matching was used during the last phase after all other filters, in order to reduce as many false positives as possible. For each existing bounding box, we would match it with all templates of the corresponding character and calculated a score for each match (the score is calculated by: # good matches/max(# keypoints for img1, # keypoints for img2)). The highest score will be compared with a threshold, if it is higher than the threshold, we would accept this bounding box as a true positive, otherwise, a false positive. The value for threshold is tuned to maximize the mAP score.

3.2.4 Parameters tuning

Haar Cascade
After we trained the Haar Cascade models, we also tuned parameters during the detection phase. We used a function in cv2 named "detectMultiScale" to detect objects of different sizes in the input image, and the main parameters we tuned were "scaleFactor" - parameter

specifying how much the image size is reduced at each image scale; "minNeighbors" - parameter specifying how many neighbors each candidate rectangle should have to retain it; and "minSize" - minimum possible object size. Objects smaller than that are ignored. The parameters were tuned based on a principle which is find as many true positives as possible with the minimum number of bounding boxes. Since we would further filter the objects chosen from this phase, we need to make sure that there were as many as true positives. Moreover, by trying to reduce the number of bounding boxes, the time taken would be shortened. We mainly used qualitative (visualization) method to evaluate the performance for current parameters because this method can show the number of true positives and bounding boxes intuitively. Take Wizard as an example, Fig. 3 shows the performance of Haar Cascade model before and after parameter tuning.



(a) Before Tuning (b) After Tuning

Figure 3: Performance improvement of Haar Cascade model after parameter tuning.

SVM

As mentioned in section 3.3.2, we tuned the SVM model by varying the parameters used to generate the features it is trained on. Since we utilize a threshold recalculation method to improve our detection results, variables used in the recalculation process are also tuned according to our needs.

We first examined the effect of varying HOG generation parameters. These parameters are mainly the number of orientation bins, pixels per cell, and cells per block. Specifying the number of orientation bins allow us to vary the resulting histogram bin width. Having a large bin size often results in lack of differentiation between each bin, and a small bin size can cause the descriptor to be unable to capture local gradient differences. In his paper, Dalal identified that increasing bin size increases performance significantly, up to about 9 bins [3]. Therefore, we began tuning our models with 9 orientations as the baseline. Dalal also mentions that 6x6 pixel cells function best for human detection [3]. In order to check the effects of tuning these parameters, we compared the SVM confidence scores of image patches generated after haar cascade detection. Fig. 4 below shows an example of a Wizard target detected via haar cascade, and its corresponding SVM confidence scores in Table 2.

Based on the Table 2, we can tell that having a bin size of 10 or 11 with a pixel per cell value of

Figure 4: Haar detection image patch fed into the SVM.

#Orientation Bins	Pixels per Cell	Confidence
9	6x6	0.9910
	8x8	0.9974
	10x10	0.9937
10	6x6	0.9894
	8x8	0.9999
	10x10	0.9974
11	6x6	0.9975
	8x8	0.9999
	10x10	0.9756
12	6x6	0.9795
	8x8	0.9963
	10x10	0.9947

Table 2: Quantitative results of adjusting HOG parameters on SVM accuracy.

8x8 yields the best results. However, this is targeted towards a wizard detection. As we trained different SVMs to detect waldo and wenda, they had different optimal HOG parameter values. For example, Waldo yielded the best results at 10 orientation bins with 8 pixels per cell. Additionally, we did not tune the cells per block parameter. According to skimage documentation, this parameter affects the block normalization, which is intended to introduce better invariance to illumination, shadowing, and edge contrast [1]. In his paper, Dalal also mentioned that a cells per block value of 2x2 or 3x3 works well for human detection. Since the images for waldo detection do not differ much in illumination and shadowing, we decided to leave it constant at 2x2.

The last parameter required when tuning the SVM detection model was the threshold recalculation factors. As mentioned in Section 3.3, the recalculation method involves two variables, the base threshold and the percentile level of 'good results'. In order to obtain the best combination of threshold parameters, we monitored the qualitative detection results. The resulting parameters varied for Waldo, Wenda and Wizard detection, due to the difference in quantity and quality of training data for the SVM. For example, we used a base threshold of 0.65 for waldo compared to 0.5 for wenda. However, when tuning the percentile level, we used the top 50% of 'good' Waldo results compared to using the top 10% for Wenda and Wizard.

3.3 Results

In this section, visual and quantitative results of various methods are displayed. Qualitative (visual) results were used when we wanted to get a rough representation of the model's performance, and quantitative results were measured for further fine-tuning. Fig. 5 shows a sample of Waldo detection using purely the Haar Cascade model. Fig. 6 shows a sample of Waldo detection using the SVM only, and Fig. 7 shows a sample of Waldo detection using our proposed solution framework.

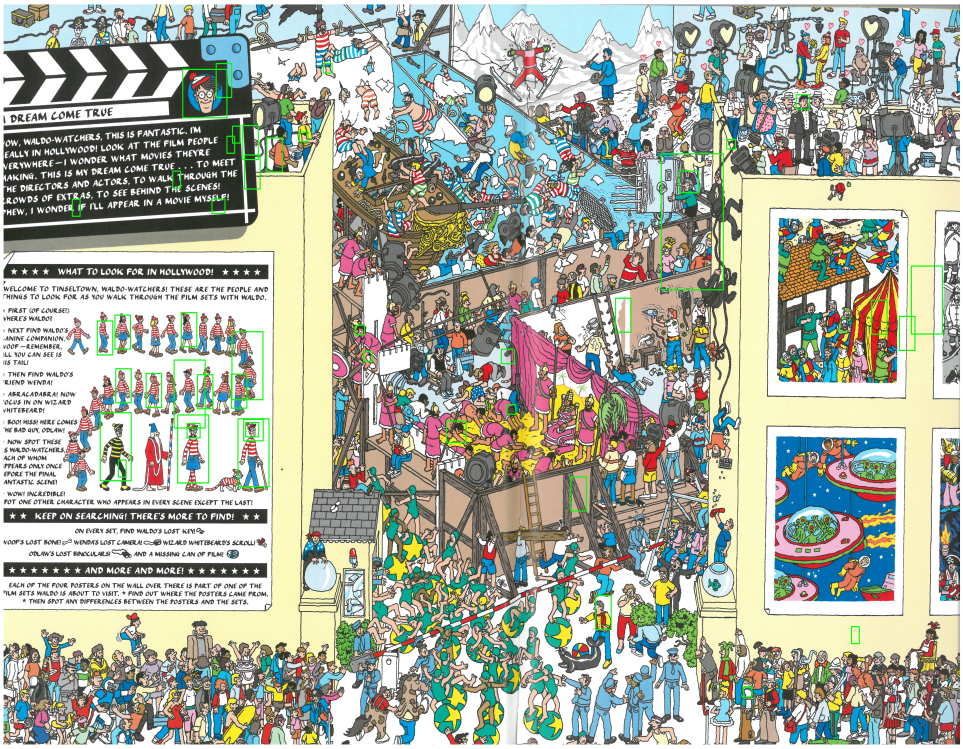


Figure 5: Waldo detection using Haar Cascade model.

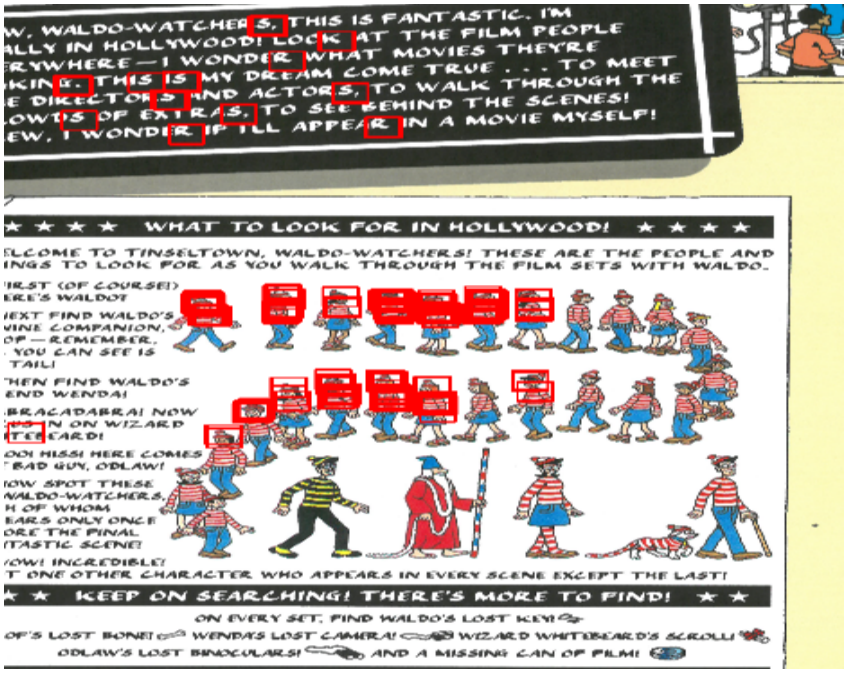


Figure 6: Waldo detection using SVM model.

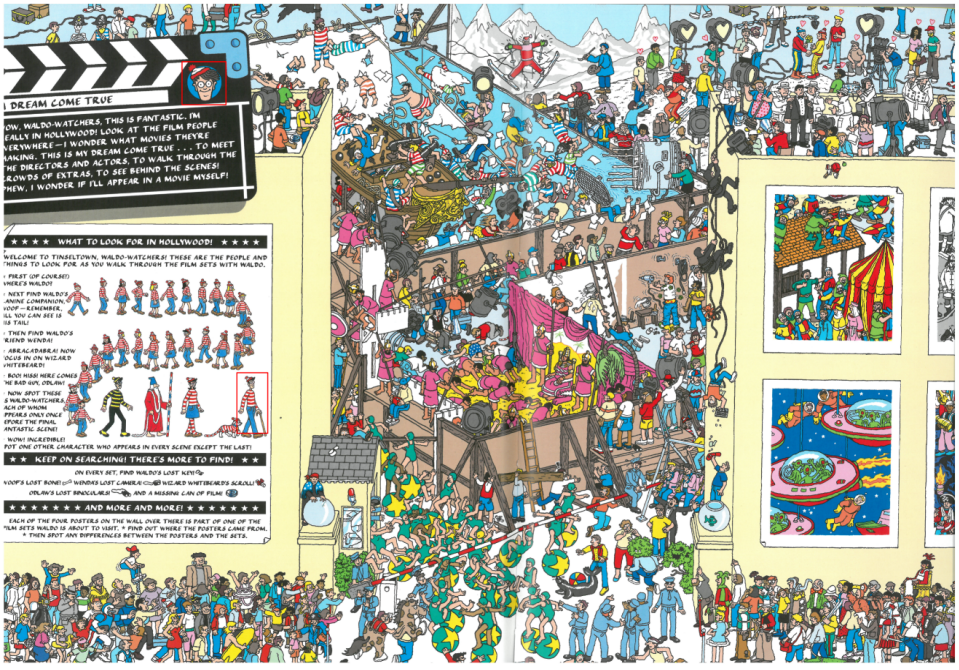


Figure 7: Waldo detection using Haar+SVM+SIFT.

Based on the visual results of our experimented methods, it is clear that cascading a combination of methods yields the best results. When looking at the individual methods, both the

haar cascade and SVM methods yield similar results, with many false positives. However, when obtaining an initial detection across the entire image, a sliding window detection scheme takes a long time compared to haar cascade, especially for large images. Hence, we decided to use haar cascade as our first layer in the proposed solution framework.

After deciding on the solution framework and tuning the model parameters, we also examined the quantitave results yielded by the solution. Table 3 below displays the average precision and recall for Waldo, Wenda and Wizard.

Threshold	Character	Recall	AP	mAP
0.05	Waldo	0.7692	0.7692	0.6024
	Wenda	0.5714	0.5714	
	Wizard	0.6666	0.4666	
0.25	Waldo	0.5385	0.4884	0.5088
	Wenda	0.5714	0.5714	
	Wizard	0.6666	0.4666	
0.50	Waldo	0.5385	0.4884	0.3811
	Wenda	0.4286	0.3214	
	Wizard	0.3333	0.3333	

Table 3: Quantitative results of proposed solution.

In order to generate recall and precision values, we utilize the given VOC evaluation function. The function uses an intersection over union (IoU) function to determine whether the detection was considered a true or false positive. At a threshold level of 0.5, it meant that the bounding box detection had to overlap at least half of the recorded ground truth. Based on table 3, there is a clear trend where lower thresholds resulted in higher mAP and recall. However, the recall results at high threshold levels differed from our qualitative results. Upon further examination, we discovered that this was due to the disparity in bounding box information. Fig. 8 below displays this disparity.

Figure 8: Bounding box differences.

Since majority of our recorded detections are faces (as shown in Fig. 6 below), it resulted in low IoU values as the ground truth bounding box values encompassed the whole body. By lowering the threshold and thus, allowing face detections to count as true positive detections, the recall and average precision values increased, as shown in Table 3.

3.4 Discussion

When testing out individual haar cascade detection, SVM or SIFT matching, Waldo detection always required us to lower the threshold to a point where many false positives are generated. This might be due to a lack of training data. However, when we cascade these various methods together, the initial results go through many filters which greatly reduces the rate of false positives.

However, since our proposed solution of cascading methods involves haar cascade detection as the first layer, if the haar cascade detection fails, the error will also carry forward into the other layers. Additionally, many of our detections are facial detections instead of the whole body, which gives bad quantitative results because the overlap against ground truth will be small.

4 Conclusion

In this project, we have talked about the approach for finding Waldo, Wenda and wizard and manage to reach mAP of 0.6024 in a short time. The first step is training Haar Classifier classifers, then using linear SVMs to train HOG descriptors of detections. The final step is feature matching to filter out non-Waldos. This paper not only describes our approach to find three characters and the way to implement it, but also presents a set of detailed experiments. For quantitative results, the mAP of wenda and wizard is lower than of waldo, the possible reason is that the dataset is limited to train. If we are able to acquire more various images of these two characters, the results can be improved.

5 Group Information

Member	Student ID	Email	Contribution
Liu Zhaoyu	A0177318X	e0253678@u.nus.edu	5.1-1
Wei Zheng Fong	A0156708X	e0035221@u.nus.edu	5.1-2
Yan Yicheng	A0170030A	e0191578@u.nus.edu	5.1-3

Table 4: Group member information.

5.1 Contribution

1. Liu Zhaoyu: Train Haar Cascade model and tune parameters for Waldo, Wenda, and Wizard; implement SIFT feature matching; Generate dataset; Code documentation; Write report.

2. Fong Wei Zheng: Train SVM model and tune parameters for Waldo, Wenda and Wizard; Put separate models together to form solution framework; Code documentation; Write report.

3. Yan Yicheng: Generate and prepare datasets; Write report

References

[1] Histogram of oriented gradients. [Online]. Available from: `https://scikit-image.org/docs/dev/auto_examples/features_detection/plot_hog.html`. Accessed: 2019-11-12.

[2] W. Berger. Deep learning haar cascade explained. [Online]. Available from: `http://www.willberger.org/cascade-haar-explained/`, Aug 2018. Accessed: 2019-11-12.

[3] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR05).*