Finding Correspondence Candidates for Indoor Object Recognition using Logistic Regression

Gunhee Kim
The Robotics Institute
gunhee@cs.cmu.edu

1 Introduction - Motivation

The topic of this project is defined as one of sub-problems in my ongoing research. The goal of my current research is to develop a recognition system for indoor objects such as a refrigerator and a microwave oven. The main difficulty of recognition for those kinds of objects lies in that they give us little visual information on it. As shown in Fig. 1, the most objects that are conventionally used in computer vision society (ex. butterfly) show large variation of color and texture, which give important clues for object recognition. However, general indoor objects (ex. refrigerators) are quite "boring," so it's not easy to capture meaningful information on their surfaces. For example, the refrigerator in Fig.1.(a) just looks like two big white rectangular boxes. This makes it difficult to apply conventional local feature descriptors such as SIFT [1] and informative image fragments [2].

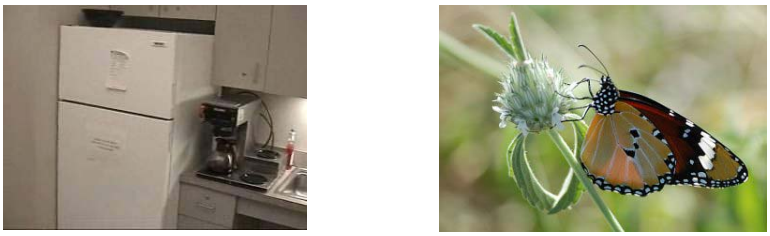
Although it's not easy to obtain meaningful information from images, one of optimistic underlying observations is that most indoor objects possess relatively consistent geometric relations within the same object class. For example, apparently, almost all refrigerators consist of two rectangles, in which one rectangle is located above another longer rectangle. And, a microwave oven looks like "a big rectangle, inside another rectangle, and some knobs and buttons in the right inside of the big rectangle."

For this reason, our approach is composed of 1) a simple line-typed feature detector to get most information from object boundaries and 2) a spectral technique based matching algorithm using pairwise geometric relations between features [3]. The details of our matching algorithm are out of the scope.

From several experiments, it turned out that the performance of our recognition algorithm is highly dependent on how well correspondence candidates are found before applying the spectral technique. Fig.2 shows an example of failures caused by this issue. The right picture is a test image, and the left one is the best matched training image. Small line segments in two images represent matched features between them. The test scene shows a plant in the hallway, but the recognizer reported a refrigerator as a best detection result. This is one of common false positive cases; if the scene is a hallway in perspective, we can see many rectangles made by doors, floors, and walls. This situation often confused our recognizer with other regular shaped objects like refrigerators.

Currently, for each feature in the test image, we find ten fixed number of correspondence candidates in each training image. The decision criterion is how much the two feature are similar in terms of each feature's length, angle, or regional

information around it. To some extent, our spectral matching can reject bad corresponding candidates by geometric filtering or score thresholds. However, experimentally, it turned out that some bad candidates are not rejected and severely distract the system. And, the complexity of our matching algorithm is quadratic with respect to the number of corresponding candidates (n) (i.e., $O(n^2)$). Therefore, the consideration of bad corresponding candidates severely increases the computation time as well as degrades the recognition performance.



(a) A "boring" indoor object – a fridge (b) An conventional object class in vision – a butterfly

Figure 1: Comparison between the target objects and other conventional objects.

Figure 2: A problem of the current approach.

2 Problem definition

2.1 Project idea

The main idea for this project is that when finding correspondence candidates, we not only the information of a single feature itself but also its neighbor such as local feature arrangements and their color/texture information . This strategy is also derived from the observation of our target object – consistent geometric relations within the same object class. That is, our underlying assumption is that if a feature in an image corresponds to a feature in different image of the same object class, they should have similar neighbor information as shown in Fig.3. Therefore, the correspondence candidates finding problem can be formulated like this; given a pair of feature, we need to decide whether these two have a similar local structure or not. That is, it is a binary decision based on several continuous variables derived from the local structure of features. Therefore, this setup can be thought of typical problem to be solved by logistic regression (i.e., binary decision based on several continuous random variables.)

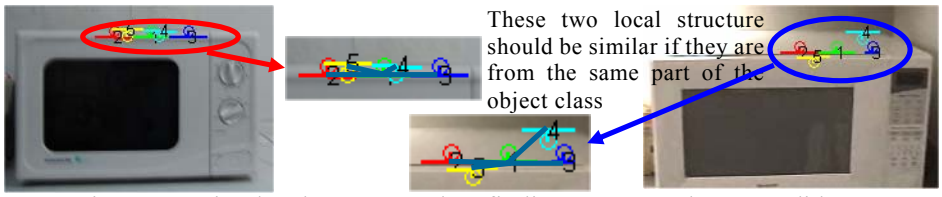


Figure 3: Using local structure when finding correspondance candidates

3 Proposed method

3.1 Intuition

Our expectation of the proposed correspondence candidate finder are two; 1) improvement of the recognition performance by finding more good candidates and rejecting more bad candidates, and 2) significant decrease of the computation time by reducing the search space of our main matching algorithm – the spectral method [3] . As a result, the new system will have two steps of recognition. First, the correspondence candidate finder will very quickly identify which features needs to be considered in next matching algorithm. As a second step, the more deliberate spectral matching algorithm correctly recognizes object within reasonable time.

3.2 Description of the algorithms

We explain the proposed algorithm by dividing into training and testing steps. Fig.4 is given for better understanding of these steps.

(1) Training Step

The procedure of training is as follows.

a. Feature extraction: For each training image, line features are extracted by using the simple Canny edge detector [4] (Fig.4.(a)).

b. Constructing KD- tree: For each training image, a KD-tree is generated in order to easily find out *n*-nearest ones for each feature (Fig.4.(b)). Since we make a local structure of each feature by integrating its local *n*-nearest features, it's much efficient to make a KD-tree beforehand.

c. Generating positive and negative samples: Within each object class, two training images are picked. Then, we manually assigned a pair of corresponding features which represent the same part of the object. These pairs become positive samples, and randomly picked pairs of features are negative samples (Fig.4.(c)). That is, it is highly likely the negative sample represent different part of the scene.

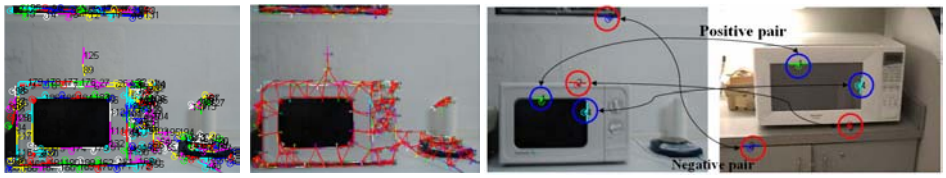
d. Constructing the training vector X and learning: Each positive and negative sample is merged into a local structure with its *n*-nearest neighbors. The input vector X is calculated from geometric and regional information within the structure, which are introduced in the Fig.4.(d). Each vector of X is defined as follows.

$$
\begin{aligned}
X &= [X_1,\ldots,X_{11}] \\
&= [1 \;\; ratio(l_{F,i}, l_{S,i}) \;\; |\alpha_{F,i} - \alpha_{S,i}| \;\; \sum_{k=1}^{n} ratio(l_{F,i,k}, l_{S,i,k}) \,|/n \\
&\quad \sum_{k=1}^{n} |\beta_{F,i} - \beta_{S,i}| \,|/n \;\; \sum_{k=1}^{n} ratio(d_{F,i,k}, d_{S,i,k}) \,|/n \;\; \sum_{k=1}^{n} |\theta_{F,i} - \theta_{S,i}| \,|/n \\
&\quad |Hl_{F,i} - Hl_{S,i}| \;\; \sum_{k=1}^{n} |Hl_{F,i,k} - Hl_{S,i,k}| \,/n \quad |Ht_{F,i} - Ht_{S,i}| \;\; \sum_{k=1}^{n} |Ht_{F,i,k} - Ht_{S,i,k}| \,/n]
\end{aligned}
\tag{1}
$$

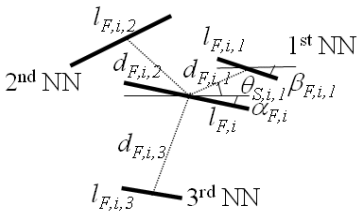
where subscript F indicates the structure from the first image and S means the one from the second image. Note that we consider a pair of local structure from a pair of image. *ratio(A,B)* means A/B if A>B, otherwise B/A. (That is, it is always larger than 1.) H_l in eighth and ninth terms represents the luminance descriptor of each feature. We take small region around the line feature and make histogram using luminance value of each pixel in that region. Similarly, H_t indicates texture descriptor, which is 32 dimensional texture histogram based on Leung-Malik filter banks [5].

$$
P(Y=1|X) = \frac{1}{1+\exp(w_0 + \sum_{i=1}^{n} w_i \cdot X_i)}, \quad P(Y=0|X) = \frac{\exp(w_0 + \sum_{i=1}^{n} w_i \cdot X_i)}{1+\exp(w_0 + \sum_{i=1}^{n} w_i \cdot X_i)} \tag{2}
$$

Using positive and negative samples, we can compute the parameter w_i in (2), which constitutes a decision surface to classify them. We iterate this step by changing n (i.e., the number of nearest neighbors to be consider when making a local structure), and find the n which show best classification performance.



(a) Feature extraction (b) Making K-D trees (c) Generating positive and negative samples



- $l_{F,i}$: the length of i-th line feature
- $l_{F,i,1}$: the length of 1st NN of the i-th feature
- $d_{F,i,1}$: the distance between i-th feature and its 1st NN
- $\alpha_{F,i}$: the absolute angle of the i-th feature
- $\beta_{F,i,1}$: the absolute angle of the 1st NN of the i-th feature
- $\theta_{F,i,1}$: the relative angle between i-th feature and its 1st NN

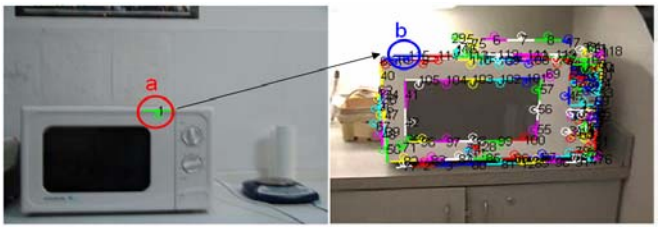
(d) Making X vectors from geometric relations

Figure 4: Overview of the algorithm in training step.

(2) Testing step

The basic task in the testing step is to find correspondence candidates in the training image for each feature in a novel test image. Thus, we scan all features in training image one by one to see if it can be a candidate or not. We compute the Eq.(2) for each

pair of features, if $Y(y_i=1|X) > 0.5$, we can decide it a correspondence candidates. Fig. 5 shows an overview of testing step.



Is a feature b (in a training image) can be a corresponding candidate for the feature *a*? (i.e., $Y(y_i=1|X) > 0.5$?). Iterate this step for all training image.

Figure 5: Overview of the Testing step.

4 Experiments

4.1 Dataset

The object classes in which we are interested are {refrigerators, microwave ovens, monitors, phones, printers, sofa, wall clocks, wiry chairs}. The database is composed of 1) 40 training images for 8 object classes (i.e., five images each object class), and 2) 146 test images. These images are taken in kitchens and hallways at CMU.

4.2 Training results

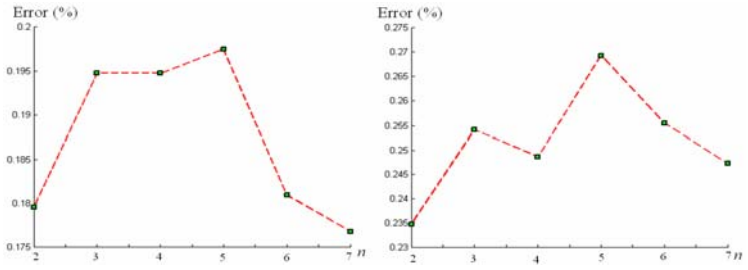
We iterate learning by changing n (i.e., the number of nearest neighbors to be considered when making a local structure) from 2 to 7. Table 1 summarizes the variation of parameters of logistic regression (w_i). Fig. 6 shows the training errors of positive and negative samples. It depicts how many training samples are misclassified. The overall tendency is that the error increases as n rise, but decrease again at high n.

Table 1: The variation of parameters of logistic regression (w_i)

n	w_i
2	[8.671, -1.538, 0.151, -0.469, -0.651, -0.271, -0.230, -0.291, 0.004, -1.181, -0.985]
3	[9.576, -1.591, 0.159, -0.452, -0.753, -0.342, -0.401, -0.206, -0.166, -1.161, -1.12]
4	[10.502, -1.647, 0.14, -0.604, -0.821, -0.363, -0.457, -0.179, -0.203, -1.114, 1.408]
5	[10.995, -1.696, 0.107, -0.610, -0.844, -0.416, -0.541, -0.272, -0.063, -1.104, 1.569]
6	[11.764, -1.727, 0.119, -0.607, -0.835, -0.477, -0.675, -0.263, -0.081, -1.082, 1.922]
7	[12.345, -1.714, 0.126, -0.682, -0.856, -0.518, -0.783, -0.369, 0.120, -1.065, 2.206]

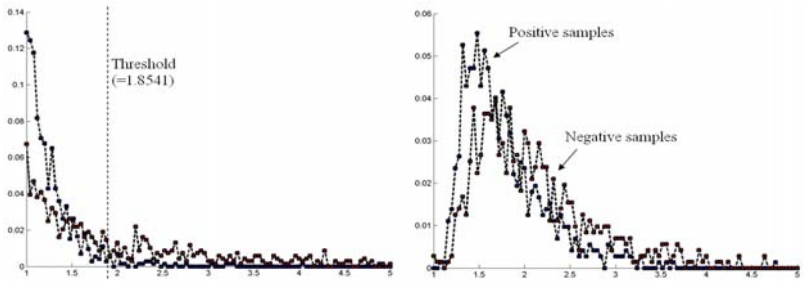
Fig. 7 shows examples of the distributions of X_i (i=1,…,11) for both positive and negative samples. The overlapped region of two distributions for a single X_i is not small, but we have 11 dimensions to distinguish them. The tendency of X_i is that the values of positive samples are a little smaller than those of negative samples. These graphs can give us the information on which X_i has more discriminative power in our training vector X. It's related to value of w_i in that more discriminative X_i has higher absolute value of w_i. From these graph, we also derive thresholds for each X_i for fast computation during the testing step. The threshold is set so that 95% of positive samples have the value of X_i below the threshold. In the example of Fig.7, for X_2

(n=6), the threshold is assigned to 1.8541. During the testing step, if X_2 of a certain pair of features is larger than this threshold, it is rejected no matter what other X_i has.



(a) Positive samples (b) Negative samples

Figure 6: Training errors.



(a) X_2 (i.e., $ratio(l_{F,i}, l_{S,i})$) (b) X_6 (i.e., $\sum_{k=1}^{n} ratio(d_{F,i,k}, d_{S,i,k}) | / n$)

Figure 7: Distributions of X_i of positive and negative samples (n=6).

Fig.8 shows a result of correspondence candidate finding during the testing step. The left image is a novel test image, and the right one is one of training image. For one of features in a test image, we find all features in a training image whose $Y(y_i=1|X)$ is larger than 0.5. In this example, we found six correspondence candidates. The values of $Y(y_i=1|X)$ are [0.9689, 0.9128, 0.9426, 0.9772, 0.9804, 0.9636]. As you see in Fig. 8, the forth best correspondence feature (i.e., the cyan colored line in the right image) is the actual correspondence feature to the query in the left image.) We can say this case is successful since the found candidates include the true correspondence feature. Even though the other five features are false positives, the next spectral matching can easily reject them. In conclusion, we can find not only the correct correspondence candidate but also reduce more searching space than the previous scheme (i.e., just using ten correspondence candidates without any learning).

Figure 8: An example of correspondence candidate finding.

Fig. 9 shows the final recognition results for all 146 test images. The black graph indicates the case in which the proposed candidate finder is not used. Rather, The top 10 features which have most similar luminance and texture descriptors are used as correspondence candidate. Generally, the recognition ratio with the proposed candidate finder is better than the ratio of that without it, although the old one is better in some object classes such as monitor and printer. Our observation is this result does not mean the logistic regression lowers the recognition performance of these two objects. But, the main reason of that degradation is that considering only the feature itself is better than considering the feature and its local neighbors in these objects. Since there's not much variation in illumination and viewpoint for these two classes of our current database, it seems to be enough to consider only the feature itself.

Another important improvement from the new candidate finder is that the overall computation time is dramatically reduced. It depends on the image, but it takes only 1/6 computation time of the previous scheme when matching a pair of images. It is due to the fact that the candidate finder can largely reduce the search space of the next deliberate spectral matching algorithm.

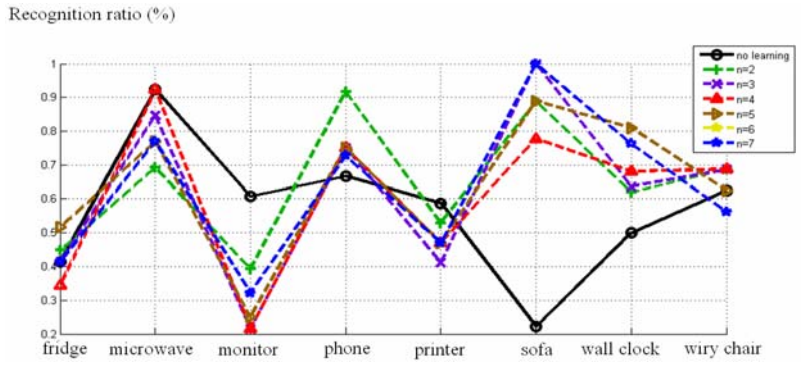
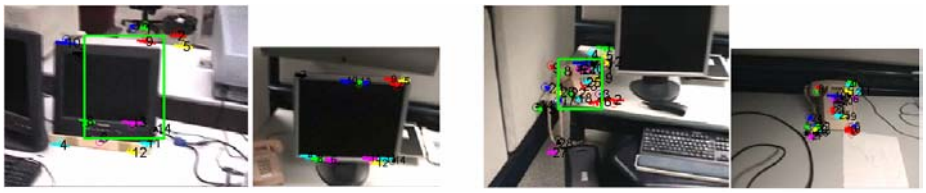
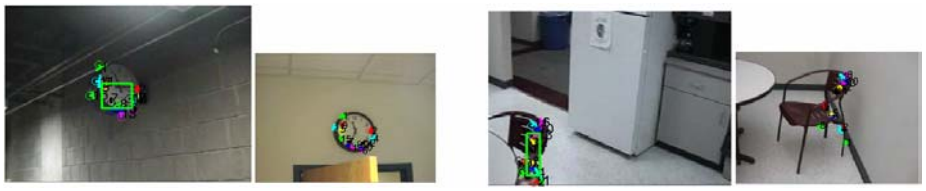


Figure 9: An example of correspondence candidate finding.



(a) Intra-class variation (b) In the cluttered scene



(c) Illumination variation (d) With only partial view

Figure 10: Examples of successful recognition.

Figure 11: Examples of failures.

Fig.10 shows some examples of successful recognition. Generally, our recognition system is robust in some variation of object instances (Fig.10.(a)) and illumination (Fig.10.(b)). Also, it's successful in the cluttered scene (Fig.10.(b)) and with only partial view of the target object (Fig.10.(d)). Fig.11 shows some typical examples of the failures. It caused by the confusion between similarly looking objects. Many of our target objects have rectangular shapes such as fridges, monitors, and microwave ovens. Therefore, it is not easy to discriminate between monitor/microwave ovens, sofas/wiry chairs, and fridge/printers.

5 Conclusions

This project deal with finding corresponding candidates for object recognition by considering not only the information of a single feature itself but also local feature structure. This strategy is based on our observation that our target system is indoor objects such as a refrigerator and a microwave oven and they have little information on it, but have strong geometrical consistency within the same object class.

Our objective is to quickly and accurately finding correspondence candidates. We formulated this problem as a binary decision based on several continuous variables from geometric and regional information of the local structure of features. As a main learning algorithm, we apply logistic regression.

By integrating the newly developed candidates finder with our main matching algorithm – the deliberate spectral technique, we can have two advantages; 1) better recognition performance using more systematic approach, and 2) significant reduction of computation time (only 1/6 computation time of the previous algorithm on average). We evaluate the developed correspondence finder with our own database, which is composed of eight indoor object classes with 40 training images and 146 test images.

References

[1] Lowe, D. G. (2006) Distinctive image features from scale-invariant keypoints, *Int. Journal of Computer Vision*, 60(2): 91-110.

[2] Vidal-Naquet, M. & Ullman, S. (2003) Random subwindows for robust image classification, *IEEE Int. Conf. Computer Vision*. pp. 281-288.

[3] Leordeanu, M. & Hebert, M. (2005) A Spectral Technique for Correspondence Problems using Pairwise Constraints. *Int. Conf. Computer Vision.*

[4] Canny, J.A. (1986) Computational Approach To Edge Detection, IEEE *Trans. Pattern Analysis and Machine Intelligence*, 8: 679-714.

[5] Leung, T. and Malik, J. (2001) Representing and recognizing the visual appearance of materials using three-dimensional textons, *Int. Journ. Computer Vision.*