EECS 504 Challenge Project

Team member: Tianhang Gao, Tao Jiang, Chi Zhang

Problem description

In this project, given datasets of “Images” and “Cards”, a spot-it automatically matching problem is solved by computer vision method.

Method and rationality

Brief introduction of method:

To achieve good results of matching, detector and descriptor of scale invariance and rotation invariance are required. We use DoG to find feature points within different scale space. And descriptor of pixel intensity with rotation invariance is implemented to windows of keypoints. Matching process is based on least error for keypoints descriptors in pairs of images. A color similarity and distance-based filter is used to remove weak and wrong matchings. And graphcut is used to remove background and label objects for display.

Implementation of our method is as the following:

Step1: Use graphcut method to remove background and label objects.

Graphcut based on max-flow gives a way to separate foreground and background by color similarity and spatial proximity of superpixels. In this project, a matching problem is conducted with the “objects” in images. By removing the part of background, foreground objects are clear to manipulate with, especially for the “Cards” dataset.

For the remaining foreground, compute distance and color similarity of superpixels to redetermine grouping of superpixels. Superpixels within the same pattern share much more similarity than others. And the distance is relevant smaller.

Step2: Find feature points in DoG scale space.

A scale space is built to determine the number of levels and corresponding sigma(scale). Local extremas in DoG scale space are considered as feature points. A filter is used to remove weak features and edges which are of low uniqueness in pictures.

Step3: Use pixel value descriptors for feature points.

Firstly rotate patches to the same orientation, and secondly resize patches to the same size using bilinear interpolation. Then transformed the processed patches to vectors of pixel values sorted in a descending order, which is the descriptor used to match.

Step4: Match corresponding points by similarity of descriptors.

Error of descriptors between each possible pair of corresponding points are computed. Smaller value of error indicates a higher possibility of correspondence. The smallest kCorr pairs of matchings are considered as relative correct matchings.

Step5: Remove bad matchings by color similarity and distance.

First, color similarity between objects with different labels are computed. Matchings between objects with color similarity less than threshold are rejected. Second, it’s known that matching on the correct object has a higher probability than matching on the wrong object. Therefore, the candidate matching points are denser on the correct object and sparser on the wrong object. This idea is used to find the optimal match.

Rationality:

In general, it is necessary to create a descriptor of the patch extracted from feature points, which is robust to scale and rotational variance.

In order to overcome rotation variance, it is understandable that the patch needs to be rotated to the same orientation for generating comparable descriptors. To rotate two patches to an unified orientation, finding domain orientation of each patch is paramount. Based on sift, the gradient of every pixel is firstly computed, and its magnitude is weighted by a Gaussian kernel with σequals to 1.5 times the scale parameter of the feature point. From the sampling principle of “3σ”, the half sample window size needs to be “3*1.5*sigma”, that explains the window size of patch extracted. For simplification, the dominant orientation is computed as the mean of the degree interval corresponds to the histogram peak.

For scale invariance, DoG scale space is required to detect the feature point at different scale, and the corresponding patch is obtained in a size based on the scale parameter of the feature point. Note that a right patch size is quite important as it may make a significant impact on the result. Specifically, a square patch is made with half window size of “ceil(3*1.5*sigma)”, which has been explained above. Thus, the process before generating the descriptor is made by firstly rotate patches to the same orientation, and secondly resize patches to the same size using bilinear interpolation.

Since the descriptor is based on the pixel value, it is not robust enough if the patch window is mislocalized. To overcome the weakness, the pixel value is firstly sorted in a descending order for each channel of the patch, and then computed the descriptor error as the sum of squared difference of two patches for each channel.

Additionally, since the error computation is based on the patch(a small region around the feature point), the error is only the representation of similarity between the local features rather than the general object/pattern. So it is possible to obtain a small error from two similar patches in different objects(e.g. the bomb and the taiji in the cards dataset, these two patterns have many similar local feature points and patches), only depending on the smallest errors may lead to a wrong match. To overcome the weakness, color similarity of superpixel and the distance of feature points are introduced to filter out the wrong matches. Color similarity of superpixel represents the similarity in a larger scale which can operate on the whole object rather than limited scale of local feature points, which can compensate for the limitation of matching only by feature points.

Good result and bad result

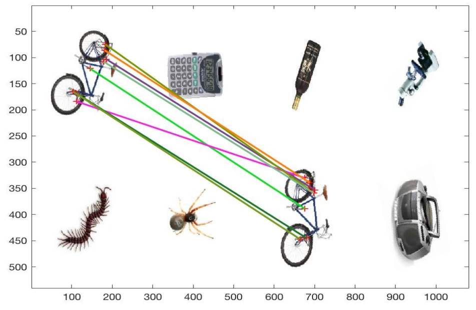
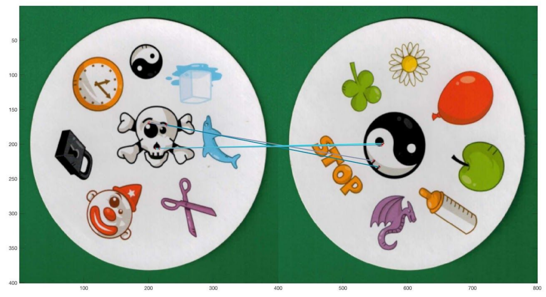

Figure 1 (left) good result (right) bad result

As is shown in the figure 1(left) above, this is a good result of 001.png and 003,png in dataset of images. Firstly, the number of local feature points is big enough to make a correct match. Secondly, the rotation degree of the matched object(bicycle) is not so big that the information(pixel value) which can be lost by rotation is negligible. Hence the descriptor of matched feature points are similar enough to determine a correct match.

Figure 1(right) is a wrong matching between 003.png and 004.png in dataset of cards. The feature points on the skull and the Taiji are much alike considering their pixel values. And color similarity between the two patterns is 0.9248, too high to be rejected. So the matchings are considered as reasonable in our system, even though it’s totally wrong. Another reason is that number of feature points in the two Taiji patterns is quite small (about 5 in each), making it difficult to match.

Deviation of performance between cards and images

First, due to the scale invariance in images and scale variance in cards, our method remains robust on image sets but less robust on card sets. Since each pattern in cards differs in size, scale space parameters have to be changed in order to match cards.

Second, it is easy to extract all patterns on images but hard to extract certain patterns on cards (ice block, for example). The probable reason is that the color is so delicate that it is hard to tell them apart from the background.

And composition of patterns in cards are more complex than those in images. Uniqueness of feature points are relative small. Since there are about 8 patterns in one card, one feature in an exact pattern may be of high similarity of that in another different pattern.