Real Time, Scalable Object Recognition with Linemod and Winner Take All

Abi Raja
Stanford University
abii@stanford.edu

Ivan Zhang
Stanford University
zhifanz@stanford.edu

1. Introduction

We will be investigating general, real-time object recognition. This problem is interesting because of its various applications in robotics. A good real-time object recognition algorithm would enable robots to perform complex tasks such as identifying mugs in the close vicinity in heavily occluded scenarios and fetching it for the human user. In particular, our goal with this project is to make an existing template matching algorithm, LINE-MOD, faster and more scalable.

2. Problem Statement

The algorithm that we are working on involves the combination of two existing techniques: (1) Linemod, a template matching technique that allows for extremely fast recognition based on extracting features from multiple modalities and generating response maps to a test image, and (2) Winner take all, a hashing algorithm that has proven quite useful in generating better results for a wide variety of similarity searches in higher dimensions including matching local feature descriptors.

Because the task at hand is primarily concerned with combining two existing techniques, the papers are extremely relevant to our problem:

Multimodal Templates for Real-Time Detection of Texture-less Objects in Heavily Cluttered Scenes. IEEE International Conference on Computer Vision (ICCV), Barcelona, Spain, November 2011.

The Power of Comparative Reasoning. Jay Yagnik, Dennis Strelow, David Ross, Ruei-Sung Lin. International Conference on Computer Vision (ICCV), 2011.

The exact method of combination is described in greater detail in the next section. Briefly, the present Linemod matching algorithm is linear in the number of different object views, but we use the ”winner take all” hashing mechanism to reduce the dimensionality of the object views for each object.

Since our algorithm produces visual results, qualitatively we can evaluate our algorithm by observing how well it performs the recognition task on various image data. We will also use standard quantitative methods to evaluate our results, such as constructing the precision and recall curves when the system identifies various objects. To resolve the ambiguity involved with a correct result, we dene a recognition to be correct if the bounding box overlaps at least 50

3. Technical Approach

We are currently following the approach proposed by Dr. Bradski which is to extract constant-size feature vectors from all the available templates for a particular object.

The LINE-MOD method primarily relies on discriminative gradient features. More specifically, a gradient image is computed for each color channel, and at a specific location on the image, the algorithm selects the gradient orientation with the largest magnitude among 3 color channels. Intuitively this improves robustness compared to using gradients computed from gray scale intensities. The algorithm then quantizes these computed gradients into bins similar to the technique used in SIFT. Similarly, a depth image is also used to compute surface normal features which is also quantized into discrete features. In Dr. Bradskis implementation, a LINE-MOD feature at a given location is 8-bits.

After computing these discriminative gradient features in training images containing objects we are trying to identify, the LINE-MOD algorithm employs a similarity measure such that, given a gradient feature in our training image and a fixed offset in the input image coordinates, search a rectangular region around the corresponding gradient location to find the most similar gradient orientation in the input image. The problem with this approach is of course that it grows linearly with the number of training images. And because we do mostly online training (for robotics), there might be a very large number of views/training images for a particular object. The method proposed by Dr. Bradski generates a number of random LINE-MOD features at each point and then, computes the responses from the training set for each of the random set of features. This is where the ”Winner Take All” algorithm becomes useful. The ”Winner Take All” algorithm is a hashing mechanism that has properties that are very useful in matching applications because it is stable to perturbations and outperforms the best machine learning methods. In our case, we take K of the maximum responses and repeatedly perform the WTA hashing until we attain a vector of desired size (the size is something that we have to figure out experimentally). Hence, we manage to reduce a large number of different object views/templates into a smaller set of constant-size feature vectors for each object.

Once feature vectors are computed for each object, we can continue to use the existing similarity measure defined in the LINE-MOD paper in order to determine matches with a test image. However, it's also possible to use a different learning algorithm that discriminates between various objects based on the feature vector of each object. FLANN is one library that can be used to accomplish the similarly search and it uses a approximate nearest neighbors technique. Doing this might make the results better but at the expense of running time depending on the learning algorithm we decide to implement (we haven't made this decision yet; FLANN, for example, will likely be faster because it's approximate and designed to be faster than a linear search).

4. Intermediate/Preliminary Results

In this section, we outline how we have utilized our time and efforts thus far. Our initial goal was to thoroughly understand the two papers. This proved quite time-consuming due to our inexperience in this field. However, once we had understood them, it became clearer how the desired speedup and scalability ought to be achieved.

Another component of this project is the existing implementation of the LINE-MOD algorithm with OpenCV written by Dr. Bradski. We spent some time understanding the codebase, getting it to compile locally and testing on various example sets of images.

Once we had a reasonable amount of familiarity with the existing code, we began implementing the enhancements proposed in the previous section. The implementation of the WTA hash itself is only a few lines of C++ and turned out to be quite straightforward. However, combining this step with the existing codebase proved to be much harder. We started implementing the random generation of LINE-MOD features. However, it's still too early to test because we haven't completely integrated the generation of the feature vectors with the rest of the codebase (we have to extend the existing Objects class to support feature vector comparison and create new match functions for our modified version of the algorithm). This is mostly a matter of coding further and takes a while due to the complexity of the project (5000+ lines of C++ code).

Going forward, the first and most important goal is to get an initial set of test results and compare the recognition results and speed of our combined algorithm with that of LINE-MOD. Once we are able to test that, we will explore other potentially interesting changes that we might be able to make (e.g. combining modalities vs. treating them separate when generating feature vectors with WTA hashing). We have yet to decide what the best approach to do matching would be (similarly measure vs. FLANN or something else). We plan experimenting with a variety of different algorithms here. We also anticipate having to vary the parameters (the length of the feature vectors and the k in the hashing algorithm) in order to get decent results.