Sean Augenstein
SU ID 0533 0698

CS 229 Project Report
Learning Techniques to Aid Pose Estimation via SIFT

Overview

I am working on machine learning techniques to intelligently track and match features in a sequence of visual images. Specifically, the feature I am tracking is known as the Scale Invariant Feature Transform (SIFT). My project involves using a camera to capture images of the motion of robotic objects in my lab. I used PCA to find a small subset of the SIFT vectors that best match the object between images. I also investigated using SVMs to reduce the size of the data we are working with, and aid the speed/quality of the overall matching process. This should result in a recognition of which features are "most distinct" for tracking a target, as well as a speedy and computationally inexpensive way to identify the object (and ultimately its pose) in future images.

Background / Hardware / Software

I work in the Aerospace Robotics Lab in the Aero/Astro Dept. We are researching techniques to autonomously track, rendezvous, and dock with a satellite that has lost attitude control and is tumbling. One can consider the case of an important asset, such as the Hubble Telescope, losing a gyro and thus its ability to point or stabilize. Towards this end, we are interested in ways to do efficient and robust pose estimation, which would be used by a spacecraft coming in to autonomously dock with the damaged satellite.

The SIFT technique was conceived by Dr. David Lowe at the University of British Columbia (Lowe, 2004). SIFT vectors are invariant to image scale and rotation, partially invariant to illumination change, and highly distinctive (because they are vectors in high dimensional space). This would make them ideal for application to the space environment, where illumination and perspective change quite a lot.

SIFT vectors are found by taking a difference of Gaussians pyramid. We then find scale-space extrema, and keep all those that have high enough contrast. At each extrema, a "keypoint descriptor" is constructed, consisting of histograms of the gradient of the image, sorted by orientation. These histograms form the elements of the SIFT vector.

As a test vehicle, I used a 'free-flying' robot that floats on an air-bearing on a frictionless granite table. Overhead cameras image the vehicle, as it moves on the table. I used code written by Dr. Lowe's lab to take an image and output its SIFT vectors, as well as their location in pixel-space. I wrote a parsing script to bring these vectors into MATLAB, and then wrote functions to perform PCA, label/match SIFT vectors, and interact with SVM-Light, a software package that trains SVMs (Joachims, 1999)

Machine Learning Aspects

While SIFT vectors are highly useful and distinctive features, the price we pay is their large size, and thus computational complexity. Typically, SIFT vectors contain 128 elements. In the images I took, there were usually 2000-3000 SIFT vectors. This means

mathematical manipulations to register common vectors between images can take many operations, because of their size and large population. It would be useful to reduce the number of SIFT vectors we're searching for matches with.

In this project, I investigated 2 different ways to speed and simplify the SIFT vector matching process. I used PCA to choose an easily distinguishable sub-set of the SIFT vectors associated with the target. I also investigated the uses of SVMs for pre-filtering images, to check if the target is present in an image, or to classify lighting as bright/dim. Figure 1 shows how these techniques fit into the overall Pose Estimation process.

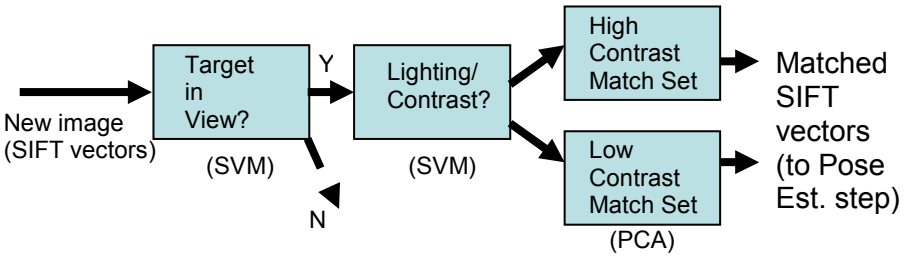

Figure 1 – Applications of Machine Learning to Pose Estimation

PCA for Smart ID Vector Selection

What is a good sub-set of the target SIFT vectors in image *i* to use, in order to find the target among the SIFT vectors in image *j*? We wish to select ID vectors that are easy to distinguish, and won't be mismatched with extraneous SIFT vectors (i.e., not residing on the target). See Figure 2. The yellow dots in the left image are the SIFT vectors we know correspond to the target; we want to match these vectors to vectors in the right image, in order to find the target in that image. So our 'ID set' is chosen from the yellow set.

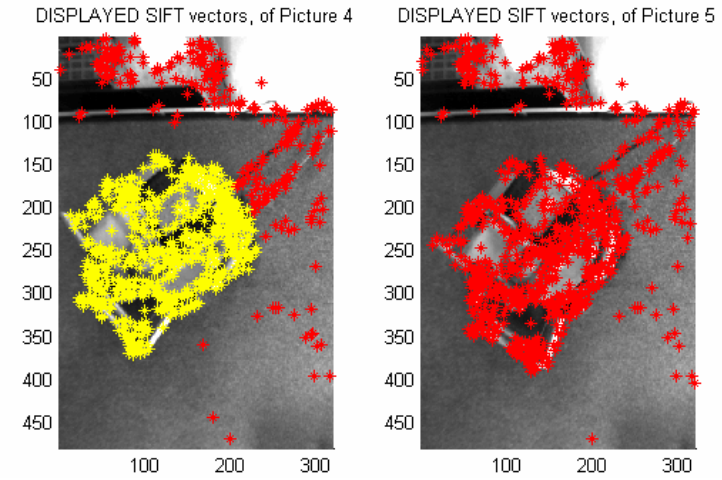

Figure 2 – The SIFT vector Matching Problem

We can do Principal Components Analysis on the full set of SIFT vectors in image *i*, which will give us the eigenvectors. The principal eigenvector (u_1) represents the direction (in 128-space) of highest SIFT vector density, and the minimal eigenvector (u_{128}) represents the direction of lowest SIFT vector density. So an ID vector that is closely aligned with u_{128} should be easy to distinguish; it is less likely we will form a false match to some other (extraneous) SIFT vector in the new image. (Note I am implicitly assuming that the overall SIFT population of images *i* and *j* are similar.)

This motivates our choice of ID vectors. We select a group of N vectors from the target set ($S_T^{(i)}$), that are the most distinct from the general SIFT vector population. We do this by selecting the SIFT vectors that maximize inner products with the smallest eigenvectors, beginning with the minimal eigenvector (u_{128}). We then find the second target vector by repeating with $-u_{128}$, and then go to u_{127}, $-u_{127}$, and so forth.

$$s_1 = \arg\max_{s \in S_T^{(i)}}(s^T u_{128}); s_2 = \arg\max_{s \in S_T^{(i)}}(-s^T u_{128})$$
$$\vdots$$
$$s_{N-1} = \arg\max_{s \in S_T^{(i)}}(s^T u_{128-(N/2)+1}); s_N = \arg\max_{s \in S_T^{(i)}}(-s^T u_{128-(N/2)+1})$$

In this way, we have chosen an ID set of SIFT vectors that should be easily distinguishable from each other, as well as from the general population of SIFT vectors present in the picture. Figure 3 shows the matches in a new image, for an ID set of 180 SIFT vectors. Approximately 17% of the ID set was matched in the new image (30 matches, out of 180 ID vectors). The number displayed is the index of the vector in the ID set.

I applied the same matching algorithm that Lowe uses in the SIFT paper; basically, it tests Euclidean distance, finds the vector that minimizes, and accepts a strong match with this vector as long as it is less than .6 times the Euclidean distance to the next smallest vector.

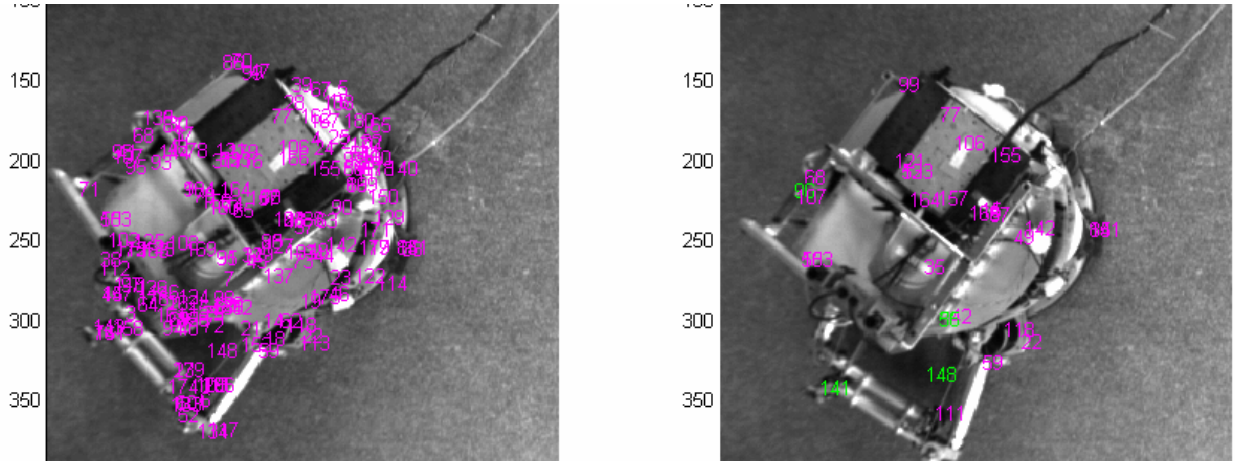

Figure 3 – forming matches with ID vector set

I also wanted to observe slightly weaker matches, so I tested for minima that were less than .7 times the next smallest vector. In Figure 3, magenta represents strong matches, and green represents weaker matches.

Forming the ID set is a Supervised Learning process, since it requires a labeled set from which to select. If this algorithm is to be used autonomously or semi-autonomously, we would like to minimize the number of times we need to build a new ID set. I tried to maintain a single ID set, and match it to successive images in time. This resulted in fewer and fewer matches as time progressed, with only about 3 matches occurring after the robot had moved through 15 images. To improve match performance, I reassigned the ID vector to its match when one occurred; in this way, the vectors of the ID set were 'propagated' through time to keep up with the changing robot. This still resulted in eventually no matches, so in the future, the ID set selection algorithm will need to be modified to autonomously re-select the optimal set of ID vectors, and to do this periodically so that match performance is maintained.

SVM for Pre-Filtering / Logic Blocks

Our matching problem suggests the use of a SVM (or other classifier) to separate SIFT vectors belonging to the object from extraneous other SIFT vectors. Once the vectors have been located on the image, I might classify them as being on the object ($y^{(i)} = 1$) or off the object ($y^{(i)} = 0$). I attempted this, using the SVM-Light software package to train an SVM with the features being the 128 elements of the SIFT vectors. I failed utterly. In retrospect, I realized that this is a data set that cannot be separated by a hyperplane in 128-space. Put another way, there is nothing intrinsically different about a SIFT vector associated with the target vs. an extraneous SIFT vector.

The next idea was to use SVM classifiers as Logic Blocks, to help make decision on how the algorithm should proceed, based on the overall scene. In this case, the training example is not an individual SIFT vector, but rather a complete image. In order to construct an SVM, I had to find a common feature vector framework, that would represent images containing different numbers of SIFT vectors. Based on the previous section, I decided to try PCA, and use the concatenation of the eigenvectors as a feature vector. So the feature vector contains 128^2 elements.

The first logic block was a basic "Target Present" identifier; could I train an SVM that would decide if the robot was visible in the scene or not? I used a training set of 10 different images, and then tested. The results were not promising. I found that at different illumination levels, the classifier performed poorly. I also changed the image size, and found the classifier was rejecting images, even when the robot was visible in the frame. In the future, I might try to retrain this classifier, with a larger training set, covering a broader range of illumination levels and image dimensions.

A second use of an SVM as a logic block was to judge the relative illumination level of an image. I observed in the previous section that matching of SIFT vectors tended to be sensitive to how bright the scene was. A set of SIFT vectors chosen to use as an ID set under darker conditions would perform worse when the lights in the room were turned brighter, and vice versa. So, we can improve registration of the target, if we choose an appropriate ID set based on the illumination of the scene.

Again using SVM-Light, I used a training set with 35 images, showing the robot and granite table set-up under both darker and brighter conditions. This time, an SVM worked very well, properly classifying 127/140 test images. My motivation for using a smaller training set was to minimize the amount of 'help' the system gets; ultimately, I would like the SIFT vector selection/match process to be almost completely autonomous.

Observations and Ideas for Future Work

When using PCA, I observed there is sensitivity to 'noise', which manifests itself in the lowest eigenvectors. If I take different images from the test set, and use my algorithm to find the first six ID vectors, different vectors are selected (see Figure 4). I hypothesize this is due to variations in SIFT vectors from image to image, which affects the directions of the lowest eigenvectors (because these are the directions of lowest vector density).

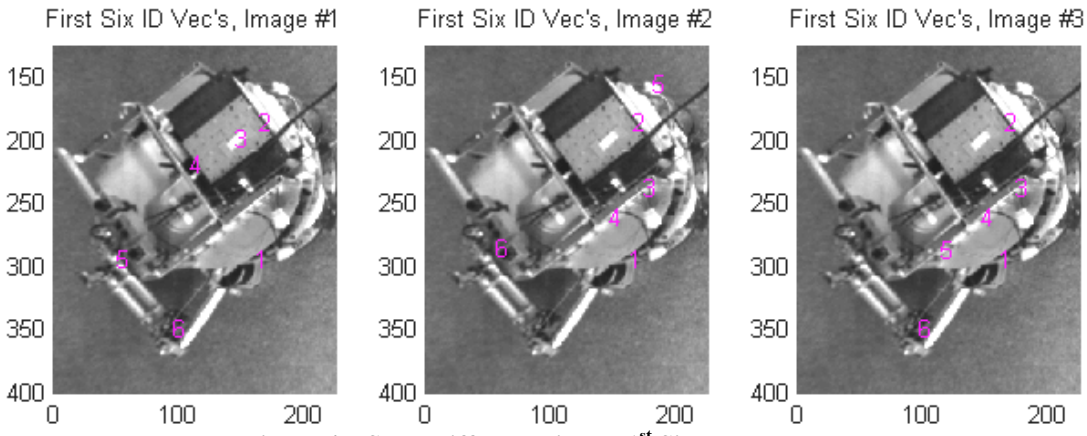

Figure 4 – Some difference in the 1st Six ID vectors chosen

This could be corrected for by doing k-means (or some kind of discriminant analysis) clustering over images of an identical scene (like the 3 images above), and then doing PCA on the clusters. This would give us a set of eigenvectors that are more stable from image to image. Also, this process could be applied to the set of target vectors from which we are choosing our ID set, so that we are sure the SIFTs we select are strongly recognizable, and not just fluctuating at the edge of scale-space detection.

The idea that the smallest eigenvectors are the most sensitive to changes in the overall vector set motivates an idea for changing the SVM we build for pre-filtering. I used a feature vector of 128^2 elements (the concatenation of ALL the eigenvectors). In a brief attempt to find a smaller feature vector, I tried using the principal eigenvector for training SVMs, and was not successful. I tried again, using the concatenation of the 5 largest eigenvectors, and again failed. But, as I observed above, it is the *smallest* eigenvectors that most reflect gradual changes to a vector set, so therefore, it might be a good idea to train an SVM with the smallest eigenvectors. My future plans for building an SVM would be to use feature selection in choosing some set (10?) of eigenvectors, to confirm my hypothesis that it is the smaller eigenvectors that are more indicative of changes to the overall set.

Acknowledgements

The author would like the thank Professor Andrew Ng, Catie Chang, and Kiran Murthy for many useful discussions concerning intelligent target SIFT vector selection. Stephen Russell and Sean Kamkar assisted in data collection. Finally, Deborah Meduna and Michael Vitus helped with understanding of general Machine Learning concepts.

References

Lowe, D.G. Distinctive Image Features from Scale-Invariant Keypoints. In *International Journal of Computer Vision,* 2004.

Joachims, T. Making large-Scale SVM Learning Practical. Advances in Kernel Methods - Support Vector Learning, B. Schölkopf and C. Burges and A. Smola (ed.), MIT-Press, 1999.