San José State University Department of Mathematics and Statistics

A Study on the Detection of Static Objects Under Various Illumination Settings

Dhatri Chennavajula, Sombo Koo, Bryan Humphries, Rishi Bommu, Aerin Kim
September 7, 2015

1 Introduction

The goal of this project is to study methods in which we identify static objects, mainly stop signs and stop lights, in images under various illumination settings. Throughout this text, a *feature* will be defined abstractly as an "interesting" part of an image. There interesting features are dependent upon the geometry of the object. Some of these features include but are not limited to: corners, lines, edges, and blobs. We will state which features will be of interest. A *feature descriptor*, henceforth *descriptor*, is a vector that describes a feature, in other words, a descriptor is a vector that describes the "interesting" part of an image.

There is a common theme associated with solving problems in computer vision: detection, description, and matching (henceforth DDM) [1] [2]. Selection of the feature plays a large component in the success of the DDM process. There are desired qualities that we would want our feature to have: locality, pose invariance, distinctiveness, and repeatability (reference needed). In order to detect objects under various illumination setting, we want to be able to detect features that are invariant to changes in pixel intensity. This poses a difficult challenge for information may be lost when images are exposed to a high or low level of light.

After we have detected our desired feature, we then undergo a description method. Common descriptors that are used are Scale - Invariant Feature Transform (SIFT) and Speeded - Up Robust Features (SURF) which will be discussed in more detail later in the paper. These description methods are useful for describing local features of an image [3]. Contrary to the locality methods of SIFT and SURF, we looked into a global descriptor called *Histogram of Oriented Gradients* (HOG). We will use various combinations of features and descriptors throughout this project.

2 Open Source Computer Vision (OpenCV)

OpenCV is a library of programming functions mainly aimed at real-time computer vision developed by Intel Russia Center and now supported by Willow Garrage and Itseez [4]. It is free for use under the open-source BSD license. The library is cross-platform and it is mainly focused on real-time image processing. There are various benefits of using OpenCV. It is a free software where users have the ability to run programs. Software may be developed and code may be modified. More importantly, there are over 500 ready to use computer vision application program interfaces (APIs).

The primary interface is in C++ [5]. OpenCV has now included interfaces in Python, Java, and MATLAB/OCTAVE (as of version 2.5) [4]. Wrappers in other languages such as C#, Perl, Ch and Ruby have been developed to encourage adoption by a wider audience. However, many of the new developments and algorithms in OpenCV are continued to be developed in the C++ interface [5][4].

A CUDA-based GPU interface has been in progress since September 2010. In addition, an OpenCL-based GPU interface has been in progress since October 2012. Furthermore, OpenCV runs on Windows, Mac, Linux, Android, FreeBSD, OpenBSD, iOS, Blackberry. OpenCV uses CMake, CMake is cross-platform free and open-source software for managing the build process using a compiler-independent method [4].

3 Features

We will discuss the various features that will be of most interest in this project. Based on the geometry of stop - signs and stop - lights, which is an octagon and a rectangle respectively, we thought that corners and lines would be the best features to detect.

3.1 Harris Corner Detector

The Harris Corner Detector uses a sliding window approach to detect corners [20]. It was an improved method of detecting corners in images developed by Chris Harris and Mike Stephens [6][20]. Given an 2-D image whose pixel intensity values $I(i, j)$ is given, the method of the Harris Corner Detector compares two patches of the image I. The original patch with particular coordinate $I(u, v)$ and the shifted patch $I(u+x, v+y)$. The method is outlined in [20]. For our purposes, the Harris Corner Detector along with SIFT is a very common combination of feature detection and feature description in the computer vision community [4].

3.2 Hough Transform

Currently, the Generalized Hough transform (GHT) is a feature extraction method used to detect analytically defined shapes such as lines, circles, ellipses, and etc. Furthermore, the

GHT is not restricted to objects that are curved. It may be used to detect polygons or defined objects [7].

The Hough transform takes assigns a pixel value from an image (x, y) to parameter space (r,θ). Historically, the Hough transform was a method used to describe lines, but using a variation of this parameterization allows for detection of both analytically defined objects and non-analytic objects. For analytical shapes, the GHT uses a voting method in parameter space implemented by directional information . Shapes that have no particular geometry are defined by their boundary points. Each point on the boundary is then parameterized using the Hough transform, with a difference that we subtract boundary points with respect to a point of reference [7] [8].

3.3 Haar-Like Features

There are methods to which we are able to detect objects of interest that are considered better than random guessing. These methods are called *weak classification.* A *cascade classifier* or a *Haar-like feature* is a detection method that are considered to be weak classifiers [9].

There were many available algorithms that used Haar-like features. The first method of interest were to detect vehicles, in particular, recognizing vehicles from behind. The Haar-cascade cars3.xml that we used for this project was trained using 526 images of cars from the rear (360 x 240 pixels, no scale). The images were extracted from the car dataset proposed by Brad Philip and Paul Updike taken of the freeways of Southern California.

A classifier is trained with a few hundred sample views of a particular object (i.e., a face or a car), called positive examples, that are scaled to the same size (20x20), and negative examples - arbitrary images of the same size. After a classifier is trained, it can be applied to a region of interest (of the same size as used during the training) in an input image. The classifier outputs are binary: 1 if the region is considered a match, and 0 if it not [10]. The algorithm works by partitioning the original image and performing a match for the object of interest via Euclidean distance. The classifier is designed so that it can be easily resized in order to be able to find the objects of interest at different sizes, which is more efficient than resizing the image itself. This process is done multiple times to find the object [10].

4 Description

In this section we will discuss the description methods that are of common use in computer vision. This is the second stage of the DDM process. One important factor in considering the description method is computational speed and accuracy. However, based on the application, one would trade computational speed for increased accuracy and vice-versa. In this section, we discuss the different descriptors used in the analysis of the data.

4.1 Scale Invariant Feature Transform (SIFT)

In 1999, David Lowe introduced one of the most prominent feature detection algorithms, SIFT [11]. Features are invariant to scale and rotation, furthermore, there is a partial invariance to change in illumination and 3-D viewpoint [2]. Lowe explains how scale and rotation invariance of his descriptors is obtained [11]. In summary, Lowe uses Laplacian based methods to approximate local features. The scale-invariance is a consequence of the approximation of the Laplacian operator via difference of Gaussian (DoG) method in detecting interest points. The descriptor that SIFT produces is a 128 dimensional vector. This 128 dimensional vector is one of the reasons why SIFT is more accurate than 64-SURF (next section). However, the downside is that it is computationally more expensive in the production of the descriptor. It should be noted that SIFT is patented by the University of British Columbia; any commercial use will require payment to UBC.

4.2 Speeded-Up Robust Features (SURF)

In 2006, Herbert Bay, Tinne Tuytelaars, and Luc Van Gool proposed an algorithm that was inspired by SIFT called Speeded-Up Robust Features or SURF. Contrary to SIFT that uses Laplacian based approximation methods to construct the detectors, SURF uses a Hessian matrix [12]. Furthermore, contrary to the 128 dimensional descriptor that SIFT produces, SURF descriptors are 64 dimensional vectors. Hence, SURF methods are faster in the description phase. There is also an analog to SIFT, SURF-128, which produces a 128 dimensional descriptor, which [12] claims to be slightly more accurate in their paper. However, SURF offers no illumination invariance in the detection of features.

4.3 Bag of Keywords

The bag-of-words (BoW) methodology was first proposed in the text retrieval domain problem for text document analysis, and it was further adapted for computer vision applications [13]. For image analysis, a visual analogue of a word is used in the BoW model, which is based on the vector quantization process by clustering low-level visual features of local regions or points, such as color, texture, and so forth.

To extract the BoW feature from images involves the following steps:

1. automatically detect regions/points of interest
2. compute local descriptors over those regions/points [14][11]
3. quantize the descriptors into words to form the visual vocabulary
4. find the occurrences in the image of each specific word in the vocabulary for constructing the BoW feature (or a histogram of word frequencies) [15].

This model can be used in conjunction with Naïve-Bayes classifier or with a support vector machine for object classification [13].

4.4 Histogram of Oriented Gradients (HOG)

In 2005, Navneet Dalal and Bill Triggs introduced their description algorithm, Histogram of Oriented Gradients or HOG. Unlike SIFT and SURF, HoG is a global descriptor. Dalal and Triggs trained a Support Vector Machine, or SVM (discussed in more detail later in the paper), to recognize HOG descriptors of people. Unlike SIFT or SURF, HoG uses a global feature to describe a person rather than a collection of local features [14]. In other words, the entire person is represented by a single feature vector, as opposed to many feature vectors representing smaller parts of the person. The HOG person detector uses a sliding detection window which is moved around the image. At each position of the detector window, a HOG descriptor is computed for the detection window. This descriptor is then shown to the trained SVM which classifies it as either person or not a person. To recognize persons at different scales, the image is subsampled to multiple sizes. Each of these subsampled images is searched [14]. Hence HoG is scale-invariant and HoG descriptors are elements of $\mathbb{R}^{128\times 64}$.

5 Statistical Methods, Classification, and Machine Learning

In this section we discuss machine learning and classification. The category of machine learning algorithms deduces a solution from trained data (citation needed). Unlike the previous descriptors which have a set method into detecting static objects, statistical methods may detect objects slightly better than guessing. These are called *weak classifiers.* Contrary to weak classification, *strong classifiers* are algorithms that will detect objects with a high level of certainty. One drawback in using strong classifiers is computational speed. Weak classifiers tend to be much faster.

The algorithm and application for these algorithms vary significantly. Data is passed to the input as training data and given a certain given output. SIFT and SURF description is not invariant under various illumination factors.

5.1 RANSAC

The *Random Sample Consensus*, more commonly known as *RANSAC*, is algorithm developed by Martin A. Fischler and Robert C. Bolles that utilizes the robust estimation methods for homographies [16]. It is a general parameter estimation approach designed to cope with a large proportion of outliers in the input data. Unlike many of the common robust estimation techniques such as M-estimators and least-median squares that have been adopted by the computer vision community from the statistics literature, RANSAC was developed from within the computer vision community [4].

The input to the RANSAC algorithm is a set of observed data values, a way of fitting some kind of model to the observations, and some confidence parameters. RANSAC achieves its goal by repeating the following steps:

1. Select a random subset of the original data. Call this subset the hypothetical inliers.
2. A model is fitted to the set of hypothetical inliers.
3. All other data are then tested against the fitted model.
4. The points that fit the estimated model well, according to some model-specific loss function, are considered part of the consensus set.
5. The estimated model is reasonably good if sufficiently many points have been classified as part of the consensus set.
6. The model may be improved by reestimating it using all members of the consensus set.

This procedure is repeated a fixed number of times, each time producing either a model which is rejected because too few points are part of the consensus set, or a refined model together with a corresponding consensus set size. In the latter case, we keep the refined model if its consensus set is larger than the previously saved model [16][4].

5.2 K-means Clustering

K-means is a clustering algorithm. The goal is to partition n data points into k clusters. Each of the n data points will be assigned to a cluster with the nearest mean. The mean of each cluster is called the *centriod* or *center* [17]. Overall, applying k-means yields k separate clusters of the original n data points. Data points inside a particular cluster are considered to be more similar to each other than data points that belong to other clusters [17]. In our case, we will be clustering the pixel intensities of a red, green and blue image. Given a $M \times N$ size image, we thus have $M \times N$ pixels, each consisting of three components: red, green and blue respectively [17]. We will treat these $M \times N$ pixels as our data points and cluster them using $k-$means. Pixels that belong to a given cluster will be more similar in color than pixels belonging to a separate cluster. One caveat of $k-$means is that we need to specify the number of clusters we want to generate ahead of time [17]. There are algorithms that automatically select the optimal value of k, but these algorithms are outside the scope of this report.

5.3 Support Vector Machines

Support Vector Machines are a class of non-probabilistic binary linear classifiers. The algorithms in a binary classification will attempt to categorize test data according to the trained model. This allows the algorithm to return very accurate true or false propositions. This is the foundation for Latent SVM [18].

Latent SVM uses SVMs to classify unknown variables of recognitions, called *latent variables*. These unknown variables represent different parts of a certain descriptor. For example, a car is embodied with wheels and headlights which we will consider latent variables. Latent SVM works by classifying all these latent variables into their own SVM trained model data set. Once the entire data has been trained, Latent SVM will enable a score threshold that will determine the descriptor [18]. Thus, given a low score, Latent SVM will say that every aspect of a picture represents a certain descriptor. Given a high score, the Latent SVM will be very particular on what it considers a certain model. A score in the middle will give us an accurate output on the current data [18].

REFERENCES

[1] Y. Yu, K. Huang, W. Chen, and T. Tan, "A Novel Algorithm For View and Illumination Invariant Image Matching," *Image Processing, IEEE Transactions on*, vol. 21, no. 1, pp. 229–240, 2012.

[2] D. G. Lowe, "Distinctive Image Features From Scale-Invariant Keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[3] Wikipedia, "Scale Ivariant Feature Transform," 2005, [Online; accessed 14-January-2015]. [Online]. Available: https://en.wikipedia.org/wiki/Scale-invariant_feature_transform

[4] R. Laganière, *OpenCV 2 Computer Vision Application Programming Cookbook: Over 50 recipes to master this library of programming functions for real-time computer vision.* Packt Publishing Ltd, 2011.

[5] Wikipedia, "Scale Ivariant Feature Transform," 2015, [Online; accessed 14-January-2015]. [Online]. Available: https://en.wikipedia.org/wiki/OpenCV

[6] C. Harris and M. Stephens, "A combined corner and edge detector." in *Alvey vision conference*, vol. 15. Citeseer, 1988, p. 50.

[7] Wikipedia, "Hough Transform," 2015. [Online]. Available: https://en.wikipedia.org/wiki/Generalised_Hough_transform

[8] unknown, "Hough Transform," 2004. [Online]. Available: http://www.cs.jhu.edu/~misha/Fall04/GHT1.pdf

[9] Wikipedia, "Statistical Classification," 2015, [Online; accessed 15-March-2015]. [Online]. Available: https://en.wikipedia.org/wiki/Statistical_classification

[10] ——, "Statistical Classification," 2015, [Online; accessed 15-March-2015]. [Online]. Available: https://en.wikipedia.org/wiki/Haar-like_features

[11] D. G. Lowe, "Object Recognition From Local Scale-Invariant Features," in *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, vol. 2. Ieee, 1999, pp. 1150–1157.

[12] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded Up Robust Features," in *Computer vision–ECCV 2006.* Springer, 2006, pp. 404–417.

[13] V. N. Vapnik and V. Vapnik, *Statistical learning theory.* Wiley New York, 1998, vol. 1.

[14] N. Dalal and B. Triggs, “Histograms of Oriented Gradients for Human Detection,” in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 1. IEEE, 2005, pp. 886–893.

[15] G. Csurka, C. Dance, L. Fan, J. Willamowski, and C. Bray, “Visual categorization with bags of keypoints,” in *Workshop on statistical learning in computer vision, ECCV*, vol. 1, no. 1-22. Prague, 2004, pp. 1–2.

[16] M. A. Fischler and R. C. Bolles, “Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography,” *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[17] Wikipedia, “K-Means Clustering,” 2015, [Online; accessed 15-March-2015]. [Online]. Available: http://en.wikipedia.org/wiki/K-means_clustering

[18] R. B. Girshick, “Discriminatively Trained Deformable Part Models,” 2012, [Online; accessed 8-March-2014]. [Online]. Available: http://www.cs.berkeley.edu/~rbg/latent/

[19] A. Bosch, X. Muñoz, and R. Martí, “Which is the best way to organize/classify images by content?” *Image and vision computing*, vol. 25, no. 6, pp. 778–791, 2007.

[20] Wikipedia, “Statistical Classification,” 2015, [Online; accessed 15-March-2015]. [Online]. Available: https://en.wikipedia.org/wiki/Corner_detection

[21] X.-C. Yin, X. Yin, K. Huang, and H.-W. Hao, “Robust Text Detection in Natural Scene Images,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 36, no. 5, pp. 970–983, 2014.

[22] M. Mostajabi and I. Gholampour, “A Robust Multilevel Segment Description for Multi-Class Object Recognition,” *Machine Vision and Applications*, vol. 26, no. 1, pp. 15–30, 2015.

[23] E. Oyallon and J. Rabin, “An Analysis and Implementation of the SURF Method, and its Comparison to SIFT,” *Image Processing On Line*, pp. 1–31, 2013.

[24] J. R. Parker, *Algorithms for Image Processing and Computer Vision.* John Wiley & Sons, 2010.

[25] Z. Kim, “Robust Lane Detection and Tracking in Challenging Scenarios,” *Intelligent Transportation Systems, IEEE Transactions on*, vol. 9, no. 1, pp. 16–26, 2008.