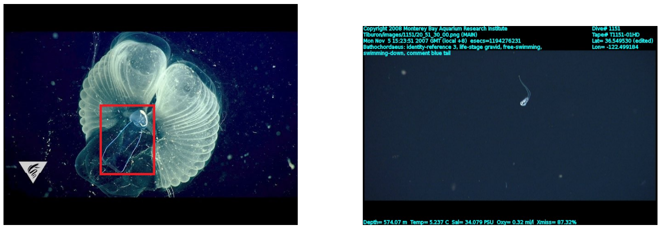
Object Detection Using Machine Learning for Autonomous Larvaceans Tracking

Miao Zhang (SUNetID: miaoz2)

Abstract— **This paper discusses an object detection algorithm that outputs the bounding box containing the target object, larvacea, given a raw submarine image. The location of the bounding box is determined by performing true/false classification on sliding windows of varied sizes on the image. Cropped images of larvaceans and non-larvaceans are labeled as positive and negative samples, mapped into SIFT and HOG feature space, as the dataset. Two class SVM and Exemplar SVM classifier, trained on SIFT features and HOG features respectively, are evaluated by performance. Two class SVM model trained on 200 clustered SIFT descriptor feature vectors has demonstrated to be the best performing model (up to 91.5% test accuracy) and is therefore used as the classifier for the object detection algorithm. Hard negative mining was also performed to improve object detection performance. Although slow, this algorithm is generally successful in outputting the correct location of the target.**

I. Introduction

The scientist at Monterey Bay Aquarium Research Institute (MBARI) are currently studying a type of planktonic marine animal named "larvacea", which is a transparent tadpole-like creature that builds mucus mesh house around its body and use the mesh to trap food particles. The state-of-art data collection is using human-piloted Remotely Operated Vehicles (ROVs) underwater with camera to film larvaceans in their natural habitat; however, the observing time is limited by human fatigue due to repetitive precise thrust corrections with high precision [1]. A full cycle of larvaceans building and abandoning the mucus mesh house takes about 1 to 2 days, in which case would require an autonomous tracking system mounted on Autonomous Underwater Vehicles (AUVs) to lock and follow the target for extended amount of time of observation. Vision algorithm, such as segmentation methods introduced in [2] has demonstrated promising result when implemented on pilot aid system [3] of ROV *Ventana* at MBARI; The ROV was able to track jellyfish for up to 89 min. Nevertheless, unlike jellyfish, which can be separated from the background by blob detection and filtering out marine snows and other objects in the frame, larvacea sometimes is contained inside the mucus house it builds (Figure 1a), which would require an vision algorithm to identify its presence and find its location. Moreover, a free-swimming larvacea (Figure 1b) also need to be distinguished from other same sized objects in the environment. Therefore, machine learning techniques are introduced to develop an algorithm that takes an image as input, and can output the location of larvacea within the input image, by applying classifier on sliding windows with varied size across the image. There are 2 proposed classification models, each trained on features extracted by a different algorithm. One classifier is trained on histogram of Scale-invariant feature transform (SIFT) features using a two class SVM model; the other classifier adapts Exemplar-SVM (E-SVM) model [4] using the Histogram of Gradient (HOG) features as exemplars. Unlike traditional object detection target, such as pedestrians, human faces etc, which usually appear in the same orientation and shape in the image, swimming larvacea body has 3 degrees of freedom of rotation and it's also deformable, which is the main challenge for this tracking task.


Copyright 2008 Monterey Bay Aquarium Research Institute
Tiburon/images/1151/20_51_30_00.png (MAIN)
Mon Nov 5 15:23:51 2007 GMT (local +8) esecs=1194276231
Bathochordaeus: identity-reference 3, life-stage gravid, free-swimming, swimming-down, comment blue tail
Dive# 1151
Tape# T1151-01HD
Lat= 36.549530 (edited)
Lon= -122.499184
Depth= 574.07 m Temp= 5.237 C Sal= 34.079 PSU Oxy= 0.32 ml/l Xmiss= 87.32%

(a) a

(b) b

Fig. 1. Images of Larvaceans. (a) Larvacea in its inner house (marked by the red box). (b) Free swimming larvacea

II. Approach

A. Dataset

Raw images containing larvaceans are acquired from video frames shoot by MBARI ROVs. Positive samples and negative samples are manually cropped out from raw images to train the classification models. Representative negative samples, such as ROV equipments that are captured in the image, other marine animals, and most importantly, the mucus house built by larvaceans, are carefully selected. Rotated and flipped copies of each sample are also added to the sample pool for data augmentation. After data augmentation, there are 92 positive samples (cropped out larvacea images) and 500 negative samples (cropped out non-larvacea object images).

B. Feature Derivation

1) SIFT Features: It is a common problem for swimming deformable bodies like larvaceans to come in various scale and orientation in images. SIFT algorithm is therefore chosen to extract features from images, since SIFT is scale and rotation invariant, and it captures local features that are present regardless of orientation changes (Figure 2).

The author is a student in the Department of Aeronautics and Astronautics at Stanford University, `miaoz2@stanford.edu`.

Comparing to other scale and rotation invariant feature extraction algorithm such as speeded up robust features (SURF), SIFT is more robust regarding to scale, orientation, blur and affine changes [5].

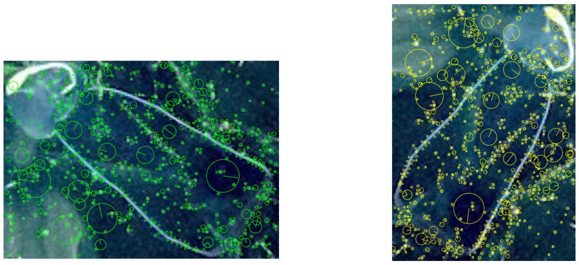


Fig. 2. SIFT features extracted from images of larvaceans in different orientations

Open source SIFT algorithm from `http://www.vlfeat.org` are used to extract SIFT features from image samples. Each sample image is first expanded in size in order to generate more layers of Gaussian pyramid, which can increase the number of stable keypoints detected [6]. Increasing the number of local features can be helpful when generating feature clusters for "bag of words". Each keypoint represents a highly distinctive local region in the image, with a descriptor in $\mathbb{R}^{128}$ composed of a 4×4 arrays of 8-bin histogram of local gradient around the keypoint [6]. When visualizing the keypoints on the images, hyperparamenters such as peak threshold and edge threshold are chosen by inspection so that most of the keypoints detected are on the larvacea animal itself.

Although the descriptors for each keypoint are in the same dimension, for different image samples, the number of keypoint detected is not uniform. To train an SVM model, each sample need to be mapped into a feature space of the same dimension; therefore, k-means clustering was used to find the centroid of clusters of SIFT descriptors in $\mathbb{R}^{128}$, i.e., to generate the "bag of visual words"; the final feature vector of an image is then a normalized histogram of the occurrence of the visual words. An example of the feature vector with 200 clusters is shown in Figure 3.

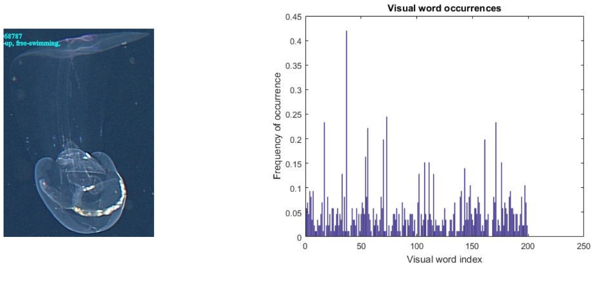


Fig. 3. SIFT feature vector of a sample image

2) HOG Features: When putting SIFT descriptors into bag of visual words, spatial information in the images are lost; hence HOG features, which preserves the spatial information, are also considered. Each image is resized into 50×50 pixels with HOG cell size 5×5 pixels, in order to get HOG feature of the same length. It is acceptable to alter the aspect ratio of the sample image, since the resized image still pertains the general shape of a larvacea. However, HOG is also rotation dependent, and the HOG feature vectors of a rotated image is not comparable to the original one, as shown in Figure 4. To resolve the issue, each orientation in the positive data pool is used as an exemplar against all negative samples to train an independent SVM.

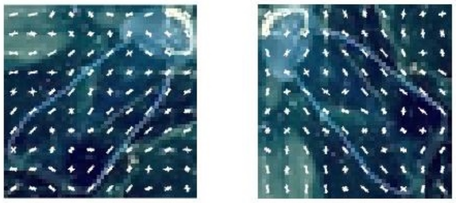


Fig. 4. Visualization of HOG features extracted from images of larvaceans in different orientation

C. Classifier Model Training

1) Two class SVM on SIFT features: Image samples split into 75% training data and 25% validation data. Each sample is labeled either "positive (1)" or "nagative (0)" and mapped into SIFT feature vector space to train a 2 class linear SVM. For k clusters feature space, the i-th sample has feature vector $x^{(i)} \in \mathbb{R}^k$, and label $y^{(i)} \in \{0, 1\}$. The model prediction $\hat{y}^{(i)}$ is computed using model parameter $\bar{w}, b$ and feature vector $x^{(i)}$: $\hat{y}^{(i)} = \bar{w} \cdot x - b$. The model is trained to find parameters $\bar{w} \in \mathbb{R}^k, b \in \mathbb{R}$ that minimizes the regularized cost given by:

$$J(\bar{w}, b) = \sum_{i=1}^{m} max(0, 1 - y^{(i)}(\bar{w} \cdot x^{(i)} - b)) + \frac{\lambda}{2}||\bar{w}||^2 \tag{1}$$

The model is trained using MATLAB built-in function *fitcecoc.m* (Error correcting output codes), while the hyperparameters such as regularization coeffiecient λ remains the default value.

2) Exemplar SVM on HOG features [4]: An independent linear SVM is trained on each positive sample against all other negative samples. Then, positive samples are grouped based on similar orientations and shapes. To calibrate the the model, each positive sample (exemplar) is held out from the exemplars of the same group, and hold out cross validation is performed using fitted sigmoid function (Eq. 2) as posterior probability transformation function (MATLAB built-in *fitSVMPosterior.m*).

$$p(y|x; \theta) = \frac{1}{1 + e^{-\theta^T x}} \tag{2}$$

Exemplars scored higher across other exemplars of the same type will have the calibrated SVM decision boundary shifted away from it, representing higher confidence around the exemplar; exemplar scored low across other exemplars of the same type will have decision boundary shifted towards it, representing low confidence around the exemplar (Figure 5).

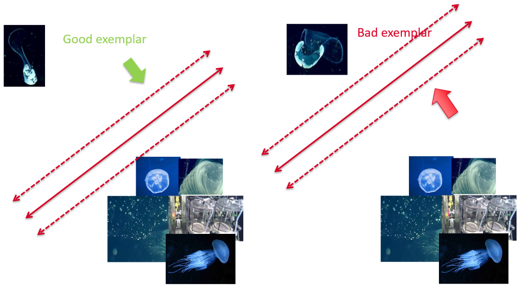
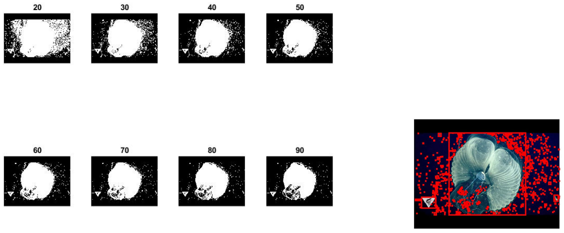


Fig. 5. E-SVM calibration process

D. Image Preprocessing

For each raw image to run the object detection algorithm on, preprocessing is first performed to get rid of the ocean background, reducing the computation time by decreasing the area of image to scan through. A simple grayscale percentage thresholding algorithm is applied to the raw image to identify and separate blobs from the empty ocean background. The grayscale percentage threshold is dependent on the illumination condition, generally, 70% as threshold does a good job at separating the blobs, as shown in Figure 6a.



(a) a

(b) b

Fig. 6. Image threshold for blob detection. (a) Binary images resulted from different percentage threshold. (b) Detected blobs marked by red bounding boxes

Only blobs that exceeds $\frac{1}{20}$ of the image area are considered as regions of interest to run the sliding window algorithm for object detection. For example, in Figure 6b, only the blob containing the house and larvacea is used for larvacea detection, which increased the efficiency.

E. Object Detection

For each cropped out region of interest, windows of varied size are slided through the entire image. Classification is performed in each window, and the bounding box will be saved if the classifier classifies the image in the window as positive. Since the region of interest blob is either only the larvacea or larvacea inside the house, the size of window is chosen according the possible size proportion of larvacea relative to the house. The classifier model is improved by hard negative mining, i.e. iteratively adding false positive samples into the negative sample pool and retrain the model. Finally, the bounding box output is given by applying non-maximum suppression to all the bounding boxes saved by the classifiers. The whole process is shown in Figure 7.

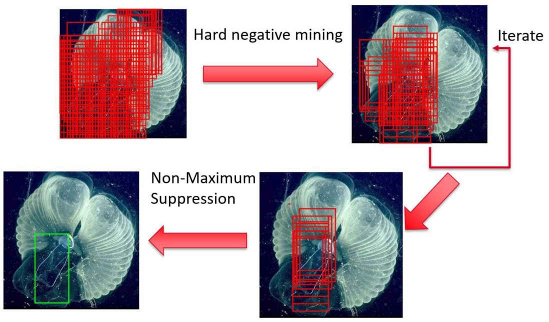


Fig. 7. Object detection process

III. Model Performance

The performance of both two class SVM and E-SVM is evaluated on the training and test set. The accuracy of each model can be seen in Table I.

TABLE I
Model Performance

Model	Training Accuracy	Test Accuracy
Two Class SVM		
100 clusters	88.6%	82.9%
200 clusters	92.7%	91.5%
500 clusters	94.3%	90.2%
Exemplar SVM		
	84.4%	76.9%

It can be observed that the training accuracy and test accuracy are more consistent when using 200 clusters for mapping the SIFT descriptors. The drop in test accuracy from training accuracy when using 500 clusters can attribute to the high variance of the overfitted model. The 100-cluster model clearly has underfitting issue, result in low accuracy in both training and test set. The 200-cluster model, proven to be the most successful on various test sets, is therefore used as the classifier for the object detection algorithm. Figure 8 shows the confusion matrix of the 200-cluster model for one of the test.

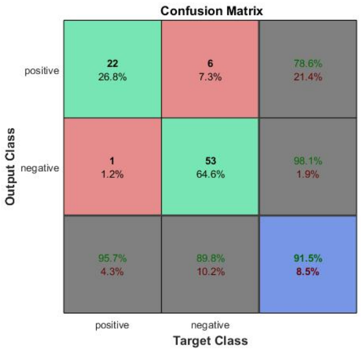


Fig. 8. Test confusion matrix for 200-cluster two class SVM model

The result for E-SVM is less satisfactory, which contradicts the high accuracy performance in [4]. However, this is by no means surprising, as the accuracy of E-SVM model heavily relies on the number of negative samples. In [4], millions of negative samples are used to train, cross validate and calibrate the model, which suggests that 500 negative samples are far from sufficient to train a good model. An interesting aspect of the model performance result is that the model predict zero false positive, and all the error sources are coming from predicting false negative, as shown in the test confusion matrix (Figure 9).

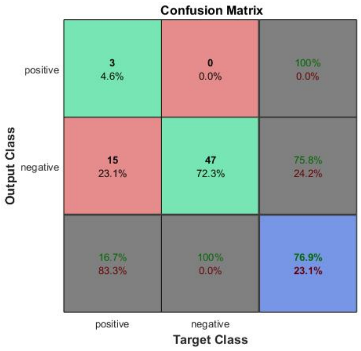


Fig. 9. Test confusion matrix for E-SVM model

Based on the calibration process demonstrated in Figure 5, this phenomenon can be explained as: all exemplars scored really low when cross validating with other exemplars of the same type, resulting in really low confidence around all positive samples, and the model tends to have a tendency to predict negative. One way to resolve this is to regroup the exemplars more carefully so that exemplars ended up in the same group has more similarities than the grouping scheme before. This zero false positive characteristic of the E-SVM model provides no advantage for the autonomous tracking application, though, since in order to be able successfully capture the target, this application actually prefer zero false negative over zero false positive; thus, the E-SVM model is not incorporated in the final object detection algorithm (only the 200-cluster two class SVM is used).

IV. Conclusion

Despite of the high test accuracy on the sample pool, on the actual raw submarine image, the initial performance is not as good, as shown in first picture in Figure 7. The reason is that the samples used to train the classifiers are manually selected, and they are either hard positive or hard negative; however in the raw submarine image, there are ambiguous windows that contains only part of the larvacea, which are not seen in the samples, and a lot of false positive are produced from this ambiguity. Iterative hard negative mining has significantly increased the chance of correctly marking larvacea location on raw image, and also continues to contributes to the diversity of the negative sample pool. This is now the first step towards autonomous tracking of larvaceans. For future work, first, the model hyperparameters should be more rigorously chosen for better model performance; second, the efficiency of the object detection algorithm need to be improved to be able to run fast enough for real-time tracking (currently, it takes about 1 min to apply classification across all windows slided on an image); third, the motion characteristic of larvaceans need to be further explored to gain more information from videos instead of separate images; lastly, the object detection algorithm needs to be able to tell the relative position of the larvacea to the underwater vehicle, and maintain observation range with the target by applying feedback control laws incorporating the vehicle dynamics [1], for a fully autonomous mission. Other machine learning techniques, such as Convolutional Neural Networks (CNN) might also be worth exploring for this application, especially it has demonstrated a lot of success in the object detection field.

Acknowledgments

My sincerest gratitude toward Professor Andrew Ng and Dan Boneh for an amazing quarter of teaching, project TAs for offering invaluable help, Professor Stephen Rock for providing guidance and ideas, and MBARI for granting usage of the image database.

References

[1] J. H. Rife and S. M. Rock, “Design and validation of a robotic control law for observation of deep-ocean jellyfish,” *IEEE Transactions on Robotics*, vol. 22, pp. 282–291, April 2006.

[2] J. Rife and S. M. Rock, “Segmentation methods for visual tracking of deep-ocean jellyfish using a conventional camera,” *IEEE Journal of Oceanic Engineering*, vol. 28, pp. 595–608, Oct 2003.

[3] J. Rife and S. M. Rock, “A pilot-aid for rov based tracking of gelatinous animals in the midwater,” in *MTS/IEEE Oceans 2001. An Ocean Odyssey. Conference Proceedings (IEEE Cat. No.01CH37295)*, vol. 2, pp. 1137–1144 vol.2, 2001.

[4] T. Malisiewicz, A. Gupta, and A. A. Efros, “Ensemble of exemplar-svms for object detection and beyond,” in *2011 International Conference on Computer Vision*, pp. 89–96, Nov 2011.

[5] L. Juan, O. Gwun, L. Juan, and O. Gwun, “A comparison of SIFT, PCA-SIFT and SURF,” *International Journal of Image Processing (IJIP)*.

[6] D. G. Lowe, “Distinctive image features from scale-invariant keypoints,” *International Journal of Computer Vision*, vol. 60, no. 2, p. 91, 2004.