Object Classification from Aerial Visual Imagery

Corey Ippolito
Intelligent Systems Division
NASA Ames Research Center
Moffett Field, CA 94035, USA
corey.a.ippolito@nasa.gov

Ara Nefian
Carnegie Mellon University Silicon Valley
NASA Ames Research Center
Moffett Field, CA 94035, USA
ara.nefian@nasa.gov

Abstract— **Aerial oil pipeline inspection is a dangerous endeavor in the current practice, where a pilot flying in a general aviation class aircraft flies slowly at low altitudes while concurrently looking at the ground for pipeline hazards with the unaided eye; high pilot workload in a dangerous low-speed, low-altitude environment results in an unacceptable number of accidents and loss of life each year. Automation of image acquisition and threat recognition has the potential to reduce pilot workload, improving the safety of the pilots and increasing efficiency. Towards these goals, this paper describes an image classification architecture and algorithm that utilizes several classifiers on different features extracted from the image to automate the threat detection process. The resulting classifier meets the requirement of greater than 80% accuracy in classification. The results will be discussed, and improvements will be proposed for continued research.**

TABLE OF CONTENTS

I. INTRODUCTION

Oil pipeline operators are mandated by federal regulation to monitor oil pipelines on a regular basis. The current practice in the industry is to perform monitoring using piloted manned aircraft. Pilots in general aviation aircraft are required to cruise at low-altitude for long durations, ideally maximizing the time they are visually inspecting the pipeline by looking out the cockpit window. In general, low altitude flight and maneuvering carries a substantially greater risk of fatality than flight at higher altitude; pilots face a number of challenges, including potential for disorientation due to visual illusions, turbulence, decreased response time to correct for upset events, and increased pilot workload – pilots must detect and avoid ground obstacles, such as trees, power lines, towers and mountains. Pilots performing pipeline inspection not only operate continuously in this hazardous environment, but the operational efficiency of this model dictates maximizing the amount of time pilots can keep their eyes off their instruments and focus on locations on the ground. Low-altitude flight dangers are exacerbated due to the high pilot workload, and an unacceptable number of accidents and loss of life occur each year.

Automation of the inspection process has the potential to substantially increase the safety and efficiency of this operation. Complete automation of the inspection process (e.g. through autonomous robotic aerial survey) is not feasible given the current regulatory environment of the FAA. Automation has the potential to reduce the workload of pilots at low altitudes, allowing pilots to focus on operating their aircraft safely. This technology should be as minimally invasive in the cockpit as possible. It should minimize pilot distractions and increase the safety of pilots by reducing their workload while maneuvering at low altitudes.

The purpose of a joint endeavor between NASA Ames Research Center and the Pipeline Research Council International is to explore advanced autonomous technologies that can be fielded on manned aircraft, particularly fixed-wing, to provide remote sensing and real-time threat identification. A large number of candidate regions are currently being produced by an existing coarse-classifier, but the performance is not meeting the objectives of the project. The success of this research is determined by achieving a classification accuracy rate of greater than 80% on the test data. This paper describes a vision processing algorithm in support of automatic identification and classification of pipeline threats. This paper aims to present the design of a system to perform object classification from aerial images. This paper will also present the results and accuracy statistics on a set of hand labeled data and describe follow-on work and improvements that can be made to this system.

A. Related Research

There is much research interest in vision-based object classification as evidenced by a significant amount of literature in related applications, such as image recognition from satellite sources, traffic flow monitoring and automated

car recognition from tower mounted cameras. Detection and classification in aerial imagery is particularly challenging due the following characteristics of the domain [1]:

(1) Rotation variance of targets in the images

(2) Poorly defined object boundaries that are often buried in the background

(3) Lack of an obvious set of features to be used because of the complex and unpredictable characteristics of the scene

(4) Camera vibration and blur

(5) Image congestion

(6) Background variance

(7) Uncertain lighting conditions

(8) Visual variation in targets

Many approaches in the literature utilize a two-stage processing approach, with a coarse filter that can be applied quickly to the scene, and a fine filter to perform more exact classification. Classification is difficult because a single feature set has not been identified that can produce good results in object detection from aerial imagery. Most recent research investigates combining a set of weak classifiers, where each classifier need only do better than 50%. The goal is to produce a strong classifier by supervised training over a labeled set of data that is indicative of the images produced. Popular ensemble methods include cascaded classifiers and boosting.

A popular approach derived from Viola and Jones [2] utilizes classifiers that are trained on features extracted from fast simple filters such as Haar-like features [2][3]. This approach is very popular for facial object recognition due its speed and performance in controlled environments, but it is challenging for aerial recognition due to variability of features [4]. Authors have explored similar techniques using additional filters in different domains, including triangle filters, Gabor filters [5], and Gaussian derivative filters, with moderate success [4]. Zernike moments and integral polar space transforms have been applied towards rotational invariance [1]. In most approaches, single classifiers do not provide acceptable results, and ensemble techniques on several classifiers are required to reach acceptable accuracy [6][7][8].

II. ARCHITECTURE

A coarse processing algorithm already exists that can identify regions of likely objects which will then be the input into a fine processing stage. The goal of this research is to develop a second fine processing stage that will produce greater than 80% accuracy. The overall pipeline is shown in Figure 1.

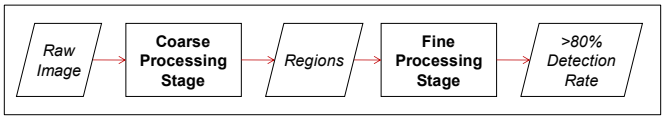

Figure 1. Processing Pipeline

To classify the images, the following feature sets were considered.

TABLE I. LIST OF FEATURE SETS

Description	ID
Gradient Orientation Distribution	GOD
Gradient Magnitude Distribution	GMD
Canny Edge Orientation Distribution	COD
Distributions Entropies	ED
Canny Pixel Ratio	CPR
Spatial Moments	SM
Central Moments	CM
Hu Moments	HM
Haar-Like Features	HLF

The following classification sets were created and utilized throughout the training.

TABLE II. LIST OF CLASSIFICATIONS

Classifier ID	Classifier Description
V	Vehicle
B	Background
T	Tractor

Unfortunately there were an insufficient number of tractors identified in the test images. In these tests, all tractors were classified as vehicles.

A. Demonstrative Test Image Example

The following test image is referred to in this document as the Demonstrative Test Image (DTI) example. The DTI was used to develop and test the algorithms, as it contains a number of vehicles and a number of false positive images. The hand-labeled image is shown below. A second set of data was also created for testing. This test set is comprised of actual identified regions from the coarse-classifier.

Figure 2. Demonstrative Test Image (DTI) Example

Figure 3. Regions in the DTI Training Set

Figure 4. Regions in the DTI Test Set

B. *Data Sets*

The classifiers were trained and tested on a set of labeled data. The full data set included 185 vehicle regions and 1552 background regions. The vehicles and background images were stored in two separate data sets. The remaining vehicle regions and background regions were incorporated into a mixed training set. This training set included the DTI regions and had a total 53 vehicle regions, with 584 regions in total.

III. CLASSIFIERS

A. *Gradient Images*

The gradient orientation histogram (GOH) filter calculates a gradient image in the x and y directions, then processes the images to characterize the overall gradients in the image.

The GOH algorithm converts the region image to grayscale, and then computes the gradients using a Sobel filter. A Sobel filter utilizes a convolution kernel applied on a per-pixel basis to quickly produce the gradient images. The results of a 3x3 and 7x7 kernel matrix were compared and an L2 norm was used to compute the magnitude. As a result of the testing, a 3x3 matrix was utilized, as the 7x7 didn't produce noticeably better results.

```
pImage = cvLoadImage ( pFilename, 1 );
cvSetImageROI( pImage, cvRect( x, y, w, h ) );
pRegionImage = cvCreateImage ( ... );
cvCopyImage ( pRawImage, pRegionImage );
cvConvertImage( pRegionImage, pCannyImg);
cvSobel ( pCannyImg, dx, 1, 0, gAptr );
cvSobel ( pCannyImg, dy, 0, 1, gAptr );
```

Figure 5. Gradient Filter Images.
Showing color region image (left), grayscale image (left-center), gradient x image (right-center) and gradient y image (righ-center).

The orientation and magnitude from gradient images dx and dy were then computed. The orientations and magnitudes were processed into a histogram of frequencies.

```
ori_deg = (atan2 ( _dy, _dx ) + CV_PI) *
    180.0 / CV_PI
mag = sqrt ( _dy*_dy + _dx*_dx )
binIndex = ( value - valuemin ) *
    ( nbins / valuerange )
```

B. *Gradient Orientation Distribution (GOD)*

The probability distribution of gradient orientations was utilized as a feature for classification. The gradient orientation angles were assembled into a histogram. The histogram contained k=18 buckets, where each bucket is assigned 20 degrees for a total range of 360 degrees. To avoid problems with zero observations, a pseudo count was added to each bin. Effectively this assumes a flat prior distribution over the orientations before the observations are added, and each added observation moves the distribution closer to the observed distribution. The distribution was computed by normalizing the histograms over the total number of observations, yielding a probability distribution satisfying:

$$p(\theta_i) = \frac{n_{\theta_i}}{N} \quad ; \quad \sum_{i=1}^{K} p(\theta_i) = 1 \quad ; \quad p(\theta_i) > 0 \qquad (1)$$

Here, $p(\theta_i)$ is the probability of a particular orientation θ_i occurring, N is the total number of observations, and $n_{\theta i}$ is the occurrence of observation occurrences with orientation θ_i. The feature vector is given by

$$V_{god} = \{p(\theta_1) \ldots p(\theta_k)\} \qquad (2)$$

C. *Gradient Magnitude Distribution (GMD)*

Similar to the gradient orientation distribution, the magnitudes of the gradients over the image were calculated.

These gradients were accumulated into a histogram with k=18 buckets and converted to a probability distribution in the same manner as mentioned previously. Let p(mi) be the probability of a gradient magnitude falling into bin I, then the feature vector is given by

$$V_{GMD} = \{p(m_1)\ldots p(m_k)\} \quad (3)$$

The V_{GMD} filter was tested on the DTI training set. The following parameters were used.

```
Number Of Bins:              18
cfil:gCannyAptr:             3
cfil:gCannyLoPos:            15.0
cfil:gCannyHiPos:            40.0
cfil:theshold_Lo:            135.0
cfil:theshold_Hi:            360.0
cfil:gMagRangeMin:           101.3
cfil:gMagRangeMax:           432.0
cfil:gMagRangeRange:         330.8
```

The magnitude histograms that resulted from the vehicles (positive) and the background (negative) are shown below. Qualitatively, the gradient magnitudes in background images are not as large as in vehicle images, and consequently this classifier does well in identifying natural background characteristics (such as grass and foliage).

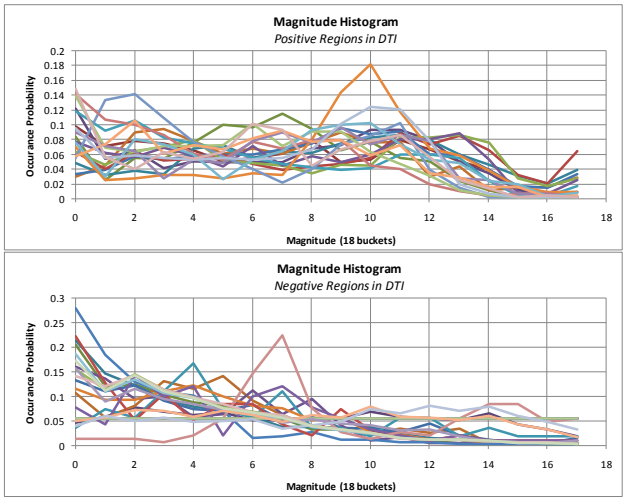

Figure 6. Magnitude Probability Distribution.
These graphs compare positive vehicle images (top) and negative background images (bottom) on the DTI

D. Canny Edge Orientation Distribution (COD)

The gradient orientation and magnitude histograms computed in the GOD above were used to compute the orientation probabilities over the entire image. Images with man-made objects, such as cars, typically have rectangular edges, and the orientation of these edges can be utilized as a feature. The Canny edge detection algorithm is a popular method for quickly determining edges in an image. The Canny edge detection algorithm implemented in OpenCV utilizes the same Sobel operators as mentioned above. However, it has a number of additional features, including a hysteresis threshold implemented in the function cvCanny, non-maxima suppression, and a stack implementation for checking neighboring cells to find edges. The Canny edge filter applied to an image is shown in Figure 7.

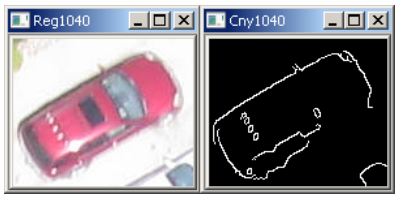

Figure 7. Canny Edge Detection

It is algorithmically expensive to extract edge directions. Instead, a Canny edge detector was used to find edges. The distribution was created exactly the same as the GOD feature, except only the gradients at vertices that coincide with Canny edges were considered. The effect of the Sobel kernel size did not have a noticeable qualitative difference in the resulting image (Figure 8), so a smaller 3x3 kernel was selected.

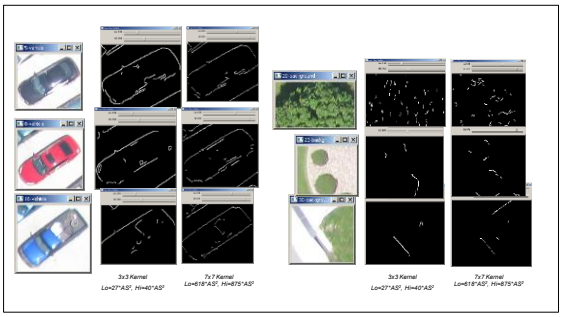

Figure 8. Canny Edge Detection Comparison of a 3x3 and 7x7 Sobel Kernel.

The resulting classification vector is given by the following, where $\hat{\theta}_i$ is the probability of an edge gradient occurring in bin i.

$$V_{COD} = \{p(\hat{\theta}_1)\ldots p(\hat{\theta}_k)\} \quad (4)$$

The orientation probability distribution in Figure 9 was generated from regions in the DTI example. Qualitatively, the resulting probability distribution over the background test cases exhibits a flatter distribution than that over vehicles. The distribution over vehicles in the DTI showed a distinct likelihood for having lines aligned at certain orientations. The DTI example contained vehicles that were all oriented in the same direction, which is why all the lines tended to group in two main locations with such strong correlation.

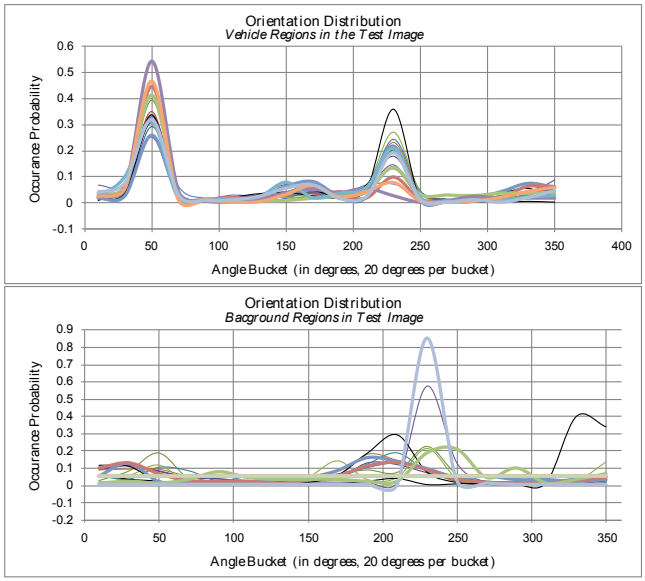

Figure 9. Canny Edge Orientation Distributions

The negative image distribution shows a few samples that correlate between edge direction and probability of occurrence. For instance, some of the negative regions contained man-made objects (see Figure 3), and the histogram reflected the trend for the edges to be aligned.

The orientation distributions performed well in identifying vehicles, but man-made objects in the scene would cause the error rate to increase.

The classifier was trained on the training data images. The training and the classification were performed with the following settings.

```
CFilter Properties:
  cfil:gCannyAptr:      3
  cfil:gCannyLoPos:     15
  cfil:gCannyHiPos:     40
  cfil:theshold_Lo:     135
  cfil:theshold_Hi:     360
  cfil:gMagRangeMin:    101.250000
  cfil:gMagRangeMax:    432.000031
  cfil:gMagRangeRange:  330.750031
  cfil:CannyFilterKnnK: 10
```

The results are shown below.

TABLE III. RESULTS OF CLASSIFICATION

Set	Matches	Mismatches	Score
Train	323	36	89.97%
Vehicles Only	146	30	82.95%
Mixed	307	115	72.75%

E. *Canny Pixel Distribution (CPD)*

The Canny Pixel Distribution is computed from a histogram over the image, comparing the ratio of edge pixels to the ratio of non-edge pixels in areas on the image. The pixel ratio R_c here is defined by

$$R_c(i,j) = \frac{P_{edge}(i,j)}{P_{total}(i,j)} \quad (5)$$

Here Pedge(i,j) and Ptotal(i,j) are the number of pixels located in a Canny edge and the number of total pixels in the image, respectively, in the (i,j) bin. In these tests we set I=J=5, for a total of 25 bins evenly distributed across the image, as shown in Figure 10.

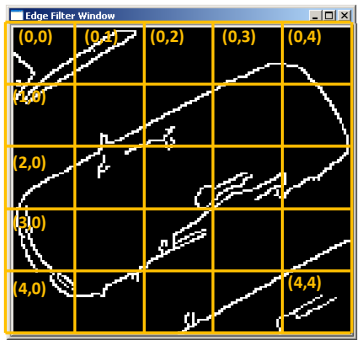

Figure 10. Pixel Ratio Histogram

The histogram is then converted to a normalized distribution. The feature vector is given by the twenty five elements in the distribution matrix.

$$V_{CPR} = \{R_c(0,0)..R_c(5,5)\} \quad (6)$$

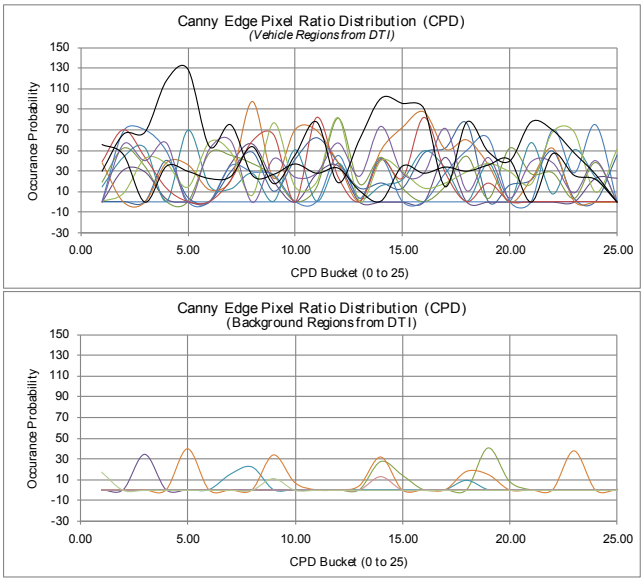

Figure 11. CPD Distributions of Test Image Regions.

The CPD distributions are shown in Figure 11 above for the first ten vehicles and background images in the test image.

F. *Distribution Entropy (DE)*

The entropy of the various distributions was computed, where entropy is given by the following:

$$E(x) = \sum_{x} p(x)^T \log_2\big(p(x)\big) \tag{7}$$

The resulting classification vector is given by the V_{DE} below, where E_v is the entropy over a vector V.

$$V_{DE} = \{E_{V_{GOD}}, E_{V_{GMD}}, E_{V_{COD}}, E_{V_{CPD}}\} \tag{8}$$

The following image shows the entropy values for E_{VGOD} as computed over regions in the DTI, shown in Figure 12. Vehicle regions tended to have consistent entropy, whereas background objects had entropy measures that fluctuated.

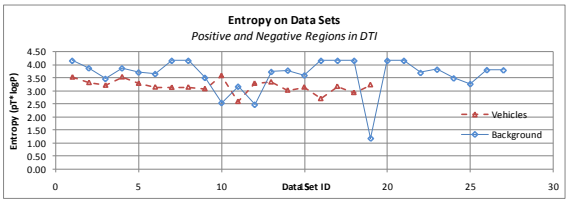

Figure 12. Entropy Comparison for Regions in the DTI

The entropy classifier was tested on the training set and a mixed test set. The results are shown below.

TABLE IV. RESULTS OF ENTROPY CLASSIFIER

Set	Matches	Mismatches	Score
Train	276	83	76.88%
Mixed	287	135	68.01%

G. Integral Spatial Moment Feature (SM)

Integral moments are a weighted average of pixel intensities integrated over an image. Consider an image whose intensity value f(x,y) is given over pixel location (x,y), then the moment over an image is given by

$$m_{p,q} = \sum_{x,y \in Image} x^p y^q f(x,y) \tag{9}$$

For instance, the value of m_{00} is the sum of intensities over the image. The feature vector for the „Spatial Moment Filter" is created by considering the following set:

$$V_{sm} = \{m_{00}, m_{10}, m_{01}, m_{20}, m_{11}, m_{02}, m_{30}, m_{21}, m_{12}, m_{03}\} \tag{10}$$

H. Integral Central Moment Feature (CM)

Central moments refer to the intensity integral relative to a point $C=(x_c,y_c)$, where the central point C is located at the „center of intensity". About a point, the moment is specified by

$$\mu_{p,q} = \sum_{x,y \in Image} (x - x_c)^p (y - y_c)^q f(x,y) \tag{11}$$

$$x_c = \frac{m_{10}}{m_{00}} \quad ; \quad y_c = \frac{m_{01}}{m_{00}} \tag{12}$$

The feature vector is composed of

$$V_{cm} = \{\mu_{20}, \mu_{11}, \mu_{02}, \mu_{30}, \mu_{21}, \mu_{12}, \mu_{03}\} \tag{13}$$

I. Hu Moment Features (HM)

The preceding moment integrals will change when an image is translated, rotated or scaled, but can be used to compute the Hu moments [9] that are invariant to these transforms. The Hu moments are given by:

$$\begin{aligned}
I_1 &= \eta_{20} + \eta_{02} \\
I_2 &= (\eta 20 - \eta 02)^2 + 4\eta 11^2 \\
I_3 &= (\eta 30 - 3\eta 12)^2 + (3\eta 21 - \eta 03)^2 \\
I_4 &= (\eta 30 + \eta 12)^2 + (\eta 21 + \eta 03)^2 \\
I_5 &= (\eta 30 - 3\eta 12)(\eta 30 + \eta 12)[(\eta 30 + \eta 12)^2 \\
&\quad - 3(\eta 21 + \eta 03)^2] \\
&\quad + (3\eta 21 - \eta 03)(\eta 21 \\
&\quad + \eta 03)[3(\eta 30 + \eta 12)^2 \\
&\quad - (\eta 21 + \eta 03)^2] \\
I_6 &= (\eta 20 - \eta 02)[(\eta 30 + \eta 12)^2 - (\eta 21 + \eta 03)^2] \\
&\quad + 4\eta 11(\eta 30 + \eta 12)(\eta 21 + \eta 03) \\
I_7 &= (3\eta 21 - \eta 03)(\eta 21 + \eta 03)[3(\eta 30 + \eta 12)^2 - (\eta 21 \\
&\quad + \eta 03)^2] - (\eta 30 - 3\eta 12)(\eta 21 \\
&\quad + \eta 03)[3(\eta 30 + \eta 12)^2 - (\eta 21 \\
&\quad + \eta 03)^2]
\end{aligned} \tag{14}$$

These set of moments are captured in the Hu moment feature vector:

$$V_{hm} = \{I_1, I_2, I_3, I_4, I_5, I_6, I_7\} \tag{15}$$

J. Haar-Like Features

Utilizing Haar-like features in a boosted, cascaded classifier is a very popular approach for fast object detection and has been very successful in detecting human faces. The original algorithm suggested by Viola and Jones has been extended over larger sets of basis features many places in the literature [4][5][10][11]. Our choice of implementation for a boosted cascade of simple features is predicated on the availability of the algorithm in common open source repositories, particularly in OpenCV.

This project implements a boosted cascade of simple Haar-like filters using the OpenCV implementation. Each classifier provides a yes/no decision based on boosting, which implements a weighted voting approach. Our implementation uses Gentle Adaboost implementation. The following features are supported through the library. Our implementation follows the standard established by the authors of the library.

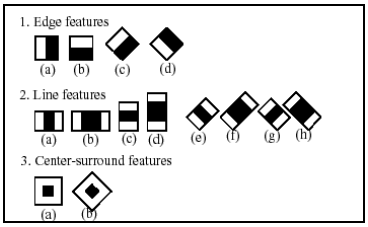

Figure 13. Set of Haar-Like Features [12]

IV. CLASSIFIER TRAINING

A K-nearest neighbor classifier was created based on the three classification types in Table II. There were an insufficient number of tractors to include in the training dataset, but it was included in the classification.

This algorithm has the option of setting parameters in the nearest neighbors search which has an effect on the accuracy of the results. The sensitivity of the results to K nearest neighbors are shown below. The test was performed by training on the DTI set, and then retesting on the same DTI set, a set of reduced positives, and a set of reduced negatives. The value of K was changed from 1 to 32, and the accuracy of the results are shown below. Interestingly, the negative set has better accuracy using less neighbors, while the positive set has better accuracy with increased K. A value of K=10 was chosen, which is the value of K where the accuracy begins to flatten out for both the positive and negative sets.

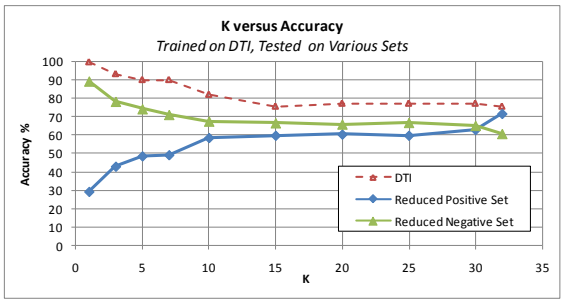

Figure 14. Sensitivity of Accuracy to K

A. Rotation Variance

Although our classifier includes a few rotation invariant metrics, several of the better performing classifiers, such as the orientation distributions, are not invariant.

To address rotation variance, each image was rotated by a number of rotations over a full 360 degree range. With a Knn classifier, we expect the addition of rotational images to provide better matches when vehicles are rotated with respect to the training images. This also has a beneficial effect of increasing the number of training samples by an order of magnitude. The number of rotations was set to 10, with 36 degrees per rotation.

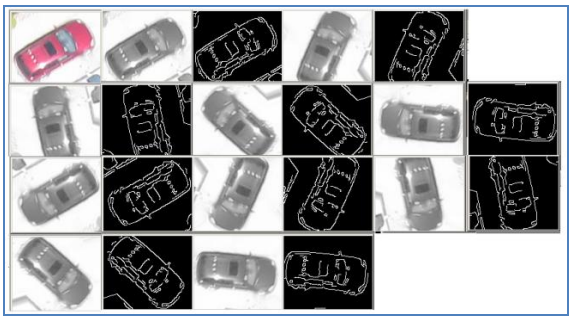

Figure 15. Rotation of Sample Image and Processing

The rotation data gave a way of testing the orientation histogram processing algorithms. Given the image rotated by 36 degrees per image, the edge histogram should reflect this by showing a translation of the histogram curve by 36 degrees, or 2 buckets (since each bucket is 18 degrees). This is reflected in the resulting histogram for the image in Figure 15 above.

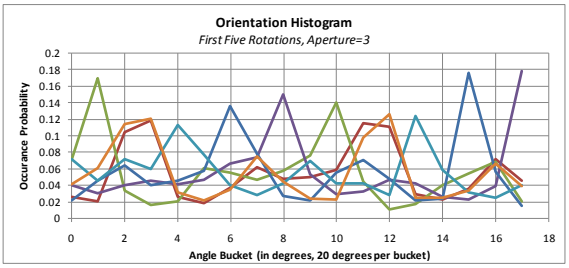

Figure 16. Edge Orientation Histogram with Rotation

The resulting matches resulted in a much better result from the gradient orientation classifier, as shown in Table V.

TABLE V. MATCHES AND MISMATCHES WITH ROTATION

Set	Matches	MisMatch	Score
Train	570	14	98.12%
Vehicles	175	10	94.59%
Backgrounds	1480	72	95.36%

B. Performance with Real Data Set

The output from the coarse system does not always center the regions on the object of interest. The coarse system is based on corner features, and the images tend to be slightly off-centered. To improve identification rate for off-centered images, the images were perturbed by 1/3 of the width of the image in the image x and y directions. Along with rotation of the images, this provides the additional benefit of enlarging the training data.

Figure 17. Shifted Image

V. COMBINING CLASSIFIERS

Individually, these classifiers performed with varying amounts of accuracy and performance; none of the individual classifiers would consistently meet the requirements of the project. The next phase of this research was to implement each possible feature into separate classifiers and combine the results using an ensemble learning technique.

In this first implementation, each classifier individually is based on the K nearest neighbor classifier, with an L2 norm distance specification determining distance to the closest neighbor over the feature vectors. We consider 10 nearest neighbors for each classifier.

A. Ensemble Approach

Our current approach to combine classifier outputs utilizes a weighted voting scheme that is driven by an external supervisor learning algorithm. The iterative algorithm generates a set of weights based on historic performance of the learners. Our ensemble classifier combines the weight of the vote from a supervised weighting scheme. A simple gradient descent operates over an L2 norm in an attempt to minimize the historic classification error. Future work will combine the rest of the features with the Haar cascaded boosting algorithm. Until these systems are combined, the results are not expected to be optimal.

B. Voting Results on the DTI

The first test was performed by training on the DTI training set. Classification accuracy results are shown in Table V. The first set of data shows classification results on the DTI training set, which provides an evaluation of the training error. The classification results were tested against the unseen training data for the DTI example.

TABLE VI. VOTING ACCURACY ON DTI
CLASSIFIERS TRAINED ON DTI IMAGE.

Set	Classifier Name	+	-	Accur
DTI	Gradient Mag Distrib	44	2	95.65%
DTI	Gradient Ori Distrib	44	2	95.65%
DTI	Distrib Entropy	44	2	95.65%
DTI	Canny Pixel Ratio	32	14	69.57%
DTI	Canny Ori Histogram	39	7	84.78%
DTI	Spatial Moments	43	3	93.48%
DTI	Central Moments	40	6	86.96%
DTI	Hu Moments	45	1	97.83%
DTI	Voting Scheme	46	0	100.0%
DTI Test	Gradient Mag Distrib	16	0	100.0%
DTI Test	Gradient Ori Distrib	6	10	37.50%
DTI Test	Distrib Entropy	6	10	37.50%
DTI Test	Canny Pixel Ratio	9	7	56.25%
DTI Test	Canny Ori Histogram	11	5	68.75%
DTI Test	Spatial Moments	12	4	75.00%
DTI Test	Central Moments	7	9	43.75%
DTI Test	Hu Moments	12	4	75.00%
DTI Test	Voting Scheme	13	3	81.25%

To characterize the training error, the first test was performed against the same set it was tested against, the DTI training set. None of the classifiers by themselves was able to perfectly train, showing various amounts of training error. However, by applying the voting scheme, the ensemble achieved 100% accuracy against the training set.

Against the DTI testing set, the voting scheme brought the overall accuracy to 81%, which was a promising improvement from the single classifier. The most accurate classifier was the gradient magnitude histogram. All other classifiers performed marginally to poorly on the unseen test set.

C. Voting Results on the Labeled Sets

The voting scheme increased the overall accuracy of the estimation for the DTI set as compared to the single classifier, so the test was reapplied to the full labeled data set. The classifiers were trained against the training set. The trained classifiers were applied against the same training set, the labeled vehicle set, and the labeled background set.

The results of the individual classifiers and the voting ensemble are shown below.

TABLE VII. VOTING ACCURACY ON FULL TEST DATA
CLASSIFIERS TRAINED ON THE TRAINING SET.

Set	Classifier Name	+	-	Accur
Train	Gradient Mag Distrb	556	28	95.21%
Train	Gradient Ori Distrb	571	13	97.77%
Train	Distrib Entropy	564	20	96.58%
Train	Canny Pixel Ratio	470	114	80.48%
Train	Canny Ori Histogram	488	96	83.56%
Train	Spatial Moments	532	52	91.10%
Train	Central Moments	524	60	89.73%
Train	Hu Moments	482	102	82.53%
Train	Voting Scheme	573	11	98.12%
Vehicles	Grd Mag Distrib	151	34	81.62%
Vehicles	Grd Ori Distrib	173	12	93.51%
Vehicles	Distrib Entropy	165	20	89.19%
Vehicles	Canny Pixel Ratio	98	87	52.97%
Vehicles	Canny Ori Distrib	122	63	65.95%
Vehicles	Spatial Moments	150	35	81.08%
Vehicles	Central Moments	126	59	68.11%
Vehicles	Hu Moments	77	108	41.62%
Vehicles	Voting Scheme	169	16	91.35%
Background	Grd Mag Distrib	1495	57	96.33%
Background	Grd Ori Distrib	1496	56	96.39%
Background	Distrib Entropy	1485	67	95.68%
Background	Canny Pixel Ratio	1321	231	85.12%
Background	Canny Ori Distrib	1319	233	84.99%
Background	Spatial Moments	1409	143	90.79%
Background	Central Moments	1447	105	93.23%
Background	Hu Moments	1368	184	88.14%
Background	Voting Scheme	1517	35	97.74%

The voting scheme overall brought the training rates past 90%, and accomplished the goals of this project.

VI. CONCLUSION

Object identification and classification in aerial images is a challenging problem due to the characteristics of the domain. Classification needs to be performed quickly, images may be blurred, there is significant background noise, objects may be partially hidden, the boundaries may not be well defined, the background will vary, etc. This project has shown an approach that gives over 90% success rates over our labeled set of regions.

The results of any single classifier were insufficient to meet the consistency and accuracy requirements of this project. Through combination of multiple classifiers, the consistency and accuracy were able to meet these objectives. Unfortunately, the results could only be tested on the limited data set derived from the relatively small number of actual aerial images available. Though the results are encouraging, the training and test set is not large enough to sufficiently draw conclusions about the algorithm in terms of accuracy or robustness.

A. Future Work

The success of this initial set of classifiers is encouraging, but the results need to be verified on a larger test set. In addition, there are many improvements that will be made to improve the training accuracy. The ensemble technique will be improved with automated random selection, towards a boosting algorithm. The initial implementation of the Haar classifier in the boosting framework resulted in a successful algorithm, if not slow, but more work needs to be done to evaluate its performance on the overall system. Additional labeled training data must be obtained. Additional classes can be added once a large enough training set is established. The system must be tested with additional output from the coarse filter, to test against more than hand-labeled data. The number of classifications needs to be increased, and a decision tree approach might be applied for part of the classifier. For instance, the orientation distribution features (COD and GOD) could identify „man-made" objects with great accuracy. The magnitude distribution feature (GMD) was able to identify natural background objects well. Additional features will also be investigated.

REFERENCES

[1] Greenberg S, Guterman H, Rotman S. An unsupervised neural network classifier for automatic aerial image recognition. Electrical and Electronics Engineers in Israel, 1996:3-6

[2] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in Accepted Conference on Computer Vision and Pattern Recognition, 2001.

[3] P. Viola and M. Jones, "Robust real-time face detection", Int. j. Comput. Vis., vol. 57, no. 2, pp. 137-154, May 2004.

[4] Haselhoff A, Kummert A. A Vehicle Detection System Based on HAAR and Triangle Features. 2009 IEEE Intelligent Vehicle Symposium (IVS 2009). 2009:261-266.

[5] Z. Sun, G. Bebis, and R. Miller, "On-road vehicle detection using Gabor filters and support vector machines," presented at the IEEE Int. Conf. Digital Signal Processing, Santorini, Greece, Jul. 2002.

[6] Tsai L, Hsieh J, Fan K. Vehicle Detection Using Normalized Color and Edge Map. Image (Rochester, N.Y.). 2007;16(3):850-864.

[7] Nguyen TT, Grabner H, Bischof H, Gruber B. On-line Boosting for Car Detection from Aerial. October. 2007:87-95.

[8] A. Broggi, P. Cerri, and P. C. Antonello, "Multi-resolution vehicle detection using artificial vision," in Proc. IEEE Intelligent Vehicles Symp., Jun. 2004, pp. 310–314.

[9] M. K. Hu, "Visual Pattern Recognition by Moment Invariants", IRE Trans. Info. Theory, vol. IT-8, pp.179–187, 1962

[10] R Lienhart and J Maydt. An Extended Set of Haar-like Features for Rapid Object Detection. IEEE ICIP 2002, Vol. 1, pp. 900-903, Sep. 2002.

[11] Alexander Kuranov, Rainer Lienhart, and Vadim Pisarevsky. An Empirical Analysis of Boosting Algorithms for Rapid Objects with an Extended Set of Haar-like Features. Intel Technical Report MRL-TR-July02-01, 2002.

[12] OpenCV User's Manual. Copyright 2009, Willow Garage. Downloaded January 2010, from http://opencv.willowgarage.com/documentation/cpp/index.html.

BIOGRAPHY

Corey Ippolito *is a research scientist at NASA Ames Research Center. He leads the Exploration Aerial Vehicles (EAV) research laboratory and several research tasks in support of the NASA Fundamental Aeronautics Program. His research focus includes intelligent control methods for next-generation and autonomous vehicle systems. He leads the Decentralized Control research project, Payload Directed Flight research project, and Polymorphic Control Systems research project. He has developed several autonomous vehicle platforms at NASA, including the Swift UAS, Exploration Aerial Vehicle (EAV), the eXperimental Sensor Controlled Aerial Vehicle (X-SCAV), and the MAX 5.0A unmanned ground vehicle. He has created several software applications and libraries, include the Reflection Architecture Reflection Architecture for autonomous vehicle systems, NAET-Gen optimization engine, SAVANT-ML for stochastic fluid modeling, the Perception Engine for physics-based simulation, the Self-Assembling Brokering Object Architecture, and the Component Graphics Library. He has received several awards and commendation at NASA, include a NASA Group Achievement Award, an Award of Excellence, an Award for Superior Accomplishment, and several recognition awards for contributions to other programs. Mr. Ippolito is currently pursuing a PhD at Carnegie Mellon University in Electrical and Computer Engineering, has an M.S. and B.S. in Aerospace Engineering from the Georgia Institute of Technology, and a graduate certificate in Space Systems Engineering from the Stevens Institute of Technology*.

Ara Nefian *holds a BS from Politehnica University Bucharest (1993) and a MSEE and PhD from Georgia Institute of Technology (1999). In the past he was with the Intel Research Labs in Santa Clara, CA, involved in several research projects including face and gesture recognition, audio-visual speech processing, web image clustering and bioinformatics. In 2005 Dr. Nefian was part of the computer vision group within the Stanford racing team (Stanley) that won the DARPA Autonomous Navigation Grand Challenge. He is currently with the Intelligent Robotics Group at NASA Ames Research Center working in 3D image modeling from planetary data. His general interests are in the area of computer vision, machine learning and robotics. He co-authored more than 30 research papers and holds ten US and international patents.*