Orientation Invariant Autonomous Recognition of Individual Snow Leopards

Sara Beery
Advisors: Agnieszka Miguel, McLean Sloughter, Rana Bayrakcismith
Fellow Researchers: Erica Flores, Loren Klemesrud, Nevan Wichers

***Abstract*—Camera trapping is used by conservation biologists to study snow leopards. In this research, we introduce techniques that find motion in camera trap images. Images are grouped into sets and a common background image is computed for each set. The background and superpixel-based features are then used to segment each image into objects that correspond to motion. The proposed methods are robust to changes in illumination due to time of day or the presence of camera flash. After the motion is detected, the images are classified by type of animal and then the snow leopard images are sorted by individual animal based on spot pattern.**

***Keywords*—Computer Vision, Pattern Recognition, Machine Learning, Image Processing, Camera Trapping, Conservation Biology**

I. Introduction

Snow leopards (*Panthera uncia*) are listed as endangered on the International Union for the Conservation of Nature Red List of Threatened Species [5]. Their range of more than 2 million km^2 spans 12 countries in Central Asia. Snow leopards are very elusive and seldom seen by people. In 2003, scientists estimated that there are between 4,500 and 7,350 snow leopards in the wild, although this estimate may be too low because as population studies have expanded over the past decade, scientists are often finding more cats than expected [10]. Conservation activities include methods to better understand this species. Researchers use ”camera traps” by placing cameras in remote areas inhabited by snow leopards. Such cameras capture photographs when a source of heat passes in front of them. These pictures are then used to recognize specific cats in order to track populations and movement patterns. Because each snow leopard has a unique coat, snow leopards are identified based on the characteristics of their spot patterns such as their size, shape, orientation, and coloration [8].

Currently, memory cards from cameras are retrieved and mailed to researchers who spend many hours analyzing the pictures by hand. Their main task is to classify the images of snow leopards into sets corresponding to each individual. In this research, we use techniques from mathematics, image processing, pattern recognition, and machine learning to aid researchers in their work with camera pictures. We first sort the images obtained from one camera based on rectangular samples from inside areas of movement within the image. The assumption is that the sample will correspond to a section of the animals fur, which can then be evaluated in order to eliminate images without leopards. Next we analyze the sorted photos to search for matches among the many different spot patterns, characterized by mathematical models of each individual leopards spots. Close match between an image and a known pattern indicates the same individual. Images taken at different times are processed and compared to a database of spot patterns. Every time a new batch of pictures is analyzed, the database will be updated since there will be new animals recognized or new views of the same animal. The database will be used to classify new sets of images as they become available.

As a result of this project, conservation biologists will be able to shift the focus of their work from inspecting every picture to verifying the classification decisions made by the pattern recognition program and to drawing conclusions from these findings. The outcomes of this work can be generalized to research on other large cats with patterned fur, such as tigers and jaguars. With the latest advancements in imagining technologies, pattern recognition and classification (machine learning) is becoming very important and even central to many computer science and engineering applications. This research will contribute to the body of knowledge in pattern classification by addressing the challenging topic of identifying non-uniform objects that are viewed from many different angles and distances.

II. Prior Work

There are many different methods of unsupervised image segmentation. We are interested predominantly in those that have been applied to the segmentation of camera trap images. Most notably, Reddy et.al. [11] use a method based on both texture and color features in order to segment tigers in camera trap images. Their method is based on multi-level nonlinear diffusion as described in [12]. In [15], Zhelezniakov et. al. propose a unsupervised two-step segmentation method. The first step uses the method of superpixel classification described in [3] and [2], then texture analysis of each superpixel is input into a Support Vector Machine in order to classify that pixel as belonging to the seal or background.

The image data used in these papers are characterized by greater differentiation in animal texture and pixel intensity from the surrounding environment than in the case of snow leopard images. The methods also do not take into account potential information provided by the continuity of the background in each image. Our method proposes to use the fact that the images in our data are taken in sets after each trigger of the motion-sensor to recognize areas of motion in each image. This information will add to that provided by texture characteristics, in order to improve segmentation results when working with very well-camouflaged animals in grayscale images.

III. Data

A camera trap is a remotely-triggered camera that is used as a non-invasive method to determine population data for a species difficult to otherwise quantify in the wild. Camera traps are invaluable in analyzing snow leopard populations, as the cats are notoriously elusive and seldom seen by researchers. The cameras are placed in known snow leopard habitats, and use either a motion or infrared sensors to trigger the shutter. To take pictures at night, some types of cameras use white or incandescent flash while others record pictures with only an infra-red light [8].

The camera trap images used in this research were provided by Panthera, a nonprofit wild cat conservation organization. The images were taken during a three year period (2009-2011) in Tost Mountains, South Gobi, Mongolia. The image dimensions are 1280 x 1024 pixels. There were between 11,000 to over 27,000 files produced each year. The images are stored in folders that correspond to different camera locations. Cameras were placed in 40-41 locations each year. The cameras used to take these images were the RECONYX RapidFire Professional Digital Infrared Cameras.

The cameras were programmed to take a sequence of five photos after the sensor is triggered. The cameras can be triggered by not only a snow leopard but by any other animal that passes in front of the sensor, as well as sudden weather changes, or wind moving the vegetation. This results in many images in each data folder that do not contain snow leopards. Fig. 1 shows examples of images used in this research. There are three daytime images of a snow leopard that are part of a sequence of 10 images taken within a very short period of time. The snow leopard moves slowly across the camera's field of view (Fig. 1(a)-(c). We also show two examples of night images. In the snow leopard image (Fig. 1(d)), the flash illuminated mainly the body of the animal. In the fox image (Fig. 1(f)), both the ground and the fox have been illuminated by the flash. Finally, Fig. 1(e) shows an example of a daytime image of an ibex which is a part of a sequence of 45 images.

IV. Proposed Motion Detection Method

To locate the areas of motion in each image, we use concepts from background subtraction [7]. Traditional background subtraction methods used to separate background from foreground in video sequences perform well when the background is static, for example indoors where the source of illumination does not vary. In the outdoor environments of camera trap images, changing weather (sun, rain, and wind) makes these methods very challenging to use. To account for the environment-related changes in the background, our first step is to sort all images from one location into *sets*. For each set, we then compute a common background image. Following, we use one of three methods to find the location of motion in each image (feature thresholding, k-means, and fuzzy k-means clustering).

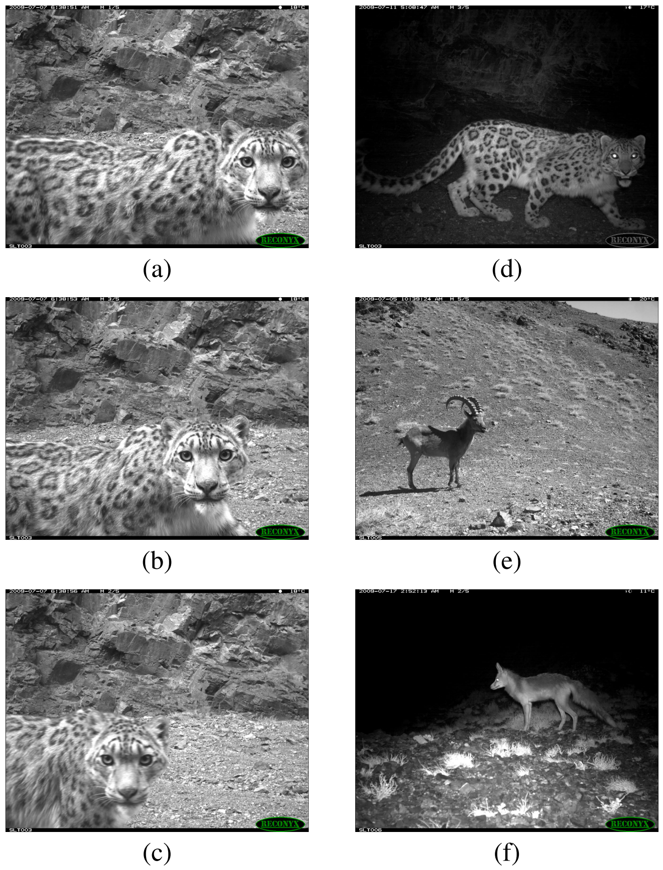


Fig. 1. Examples of images used in our research. (a)-(c) Snow leopard images that are part of a sequence of 10 images. (d) Snow leopard in a night picture. (e) Ibex. (f) Fox in a night picture.

A. Sets of Images

We determine which images should be grouped into a single set for background computation by comparing the conditions under which each photo was taken. We then group daytime images into sets according to the time when these images were taken. Initially, all daytime images that were taken within 90 seconds of each other belong to the same set. Following, we split daytime image sets using information about the ISO and exposure settings of the camera. We assume that a change in camera settings results in significantly different image that should be part of a different set. We group nighttime images into sets based only on the camera settings. To reduce the run time of our background computation, no sets can have more than 30 images. The minimum number of images in a set is 3.

B. Computing Backgrounds

We compute a background image for each set of images using median filtering which was shown to be very robust compared to higher complexity methods. The background image B_k for set k containing K images is defined as

$$B_k(x,y) = \text{median}\{I_1(x,y), I_2(x,y), ..., I_i(x,y), ..., I_K(x,y)\}. \quad (1)$$

Fig. 2 shows examples of backgrounds computed for sets of camera trap images. The first background, computed for a set of 17 nighttime images (Fig. 2a), is clear of foreground objects. The second background, computed for a set of 10 daytime images (Fig. 2b), contains a faint outline of an animal which illustrates how it is sometimes challenging for the median filter to deal with slow motion.

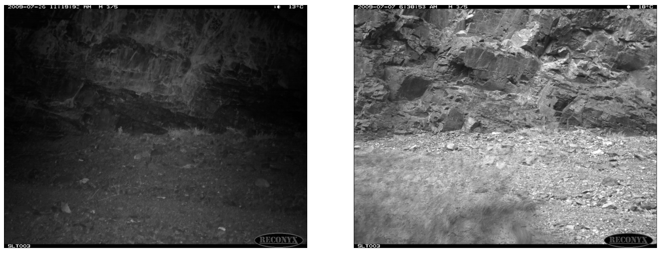

(a) (b)

Fig. 2. Examples of background images computed from a sequence of (a) 17 nighttime images and (b) 10 daytime images.

C. Motion Estimation

Background subtraction in which each image is subtracted from its corresponding background is a typical approach to obtain the foreground mask. However, for snow leopard camera trap images, such approach proves to be inadequate and leads to very noisy results where large sections of motion are missing. Instead, we use the concept of superpixels and image features.

1) Superpixels: Superpixels were developed as an alternative to the traditional pixel grid. Superpixel algorithms group pixels into perceptually meaningful regions. In this work, we use the Simple Linear Iterative Clustering with Zero Parameters (SLICO) algorithm to form 10,000 superpixels in our camera trap images [1]. The SLICO algorithm clusters pixels in the combined five-dimensional color and image plane space to efficiently generate compact, nearly uniform superpixels. You can see an example of how and where superpixels are found in Fig. 3.

We found that this large number of superpixels, though computationally expensive, results in more accurate final motion masks.

2) Features: We use several types of superpixel-based texture features. Assume that image $I_i(x,y)$ has been segmented into N superpixels. In this work, $N = 10000$. Subscript i indicates that image $I_i(x,y)$ is the ith image in set k. Its corresponding background image is $B_k(x,y)$. Let $S^n(I_i)$ be the set of all pixel coordinates that belong to the nth superpixel computed for image $I_i(x,y)$. We then define $I_i^n(x,y)$ as the segment of image $I_i(x,y)$ that corresponds to the nth superpixel:

$$I_i^n(x,y) = \{I_i(x,y) : (x,y) \in S^n(I_i)\}$$

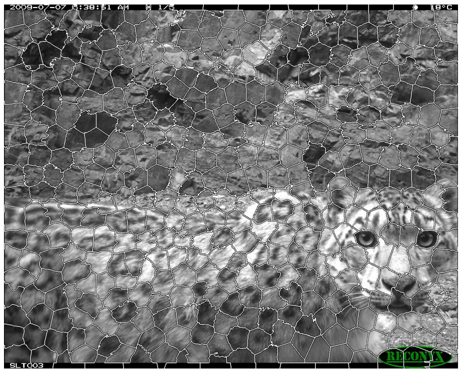


Fig. 3. Example of 200 superpixels found within a snow leopard image

and

$$I_i(x,y) = \bigcup_n I_i^n(x,y).$$

Next, we define the operation of creating a new image by assigning constant values to all pixels in each superpixel. Let

$$m = \{m(1), m(2), ..., m(n), ..., m(N)\}$$

be a vector of length N and

$$I_i^n(x,y) \leftarrow m(n)$$

correspond to replacing all pixel values in $I_i^n(x,y)$ with $m(n)$. We can create a new image with N values as

$$M_i(x,y) = \bigcup_n \{I_i^n(x,y) \leftarrow m(n)\}.$$

The *Mean of Differences* motion feature is computed as follows:

$$\text{MoD}\{I_i(x,y)\} = \bigcup_n \left\{ I_i^n(x,y) \leftarrow \frac{1}{|S^n(I_i)|} \times \sum_{(x,y)\in S^n(I_i)} |B_k(x,y) - I_i(x,y)| \right\},$$

where $|S^n(I_i)|$ is the number of pixels in superpixel n.

The *Difference of Means* motion feature is defined as:

$$\text{DoM}\{I_i(x,y)\} = \bigcup_n \left\{ I_i^n(x,y) \leftarrow \frac{1}{|S^n(I_i)|} \times \left| \sum_{(x,y)\in S^n(I_i)} B_k(x,y) - \sum_{(x,y)\in S^n(I_i)} I_i(x,y) \right| \right\}$$

In addition to MoD and DoM, we also use mean, range, and median of superpixels as our features:

$$\text{Mean}\{I_i(x,y)\} = \bigcup_n \left\{ I_i^n(x,y) \leftarrow \frac{1}{|S^n(I_i)|} \times \sum_{(x,y)\in S^n(I_i)} I_i(x,y) \right\}$$

$$\text{Range}\,\{I_i(x,y)\} = \bigcup_n \left\{ I_i^n(x,y) \leftarrow \left(\max_{(x,y)\in S^n(I_i)} I_i(x,y) - \min_{(x,y)\in S^n(I_i)} I_i(x,y) \right) \right\}$$

$$\text{Median}\,\{I_i(x,y)\} = \bigcup_n \left\{ I_i^n(x,y) \leftarrow \text{median}_{(x,y)\in S^n(I_i)} I_i(x,y) \right\}$$

D. Feature Thresholding

Our first method of finding motion in camera trap images computes MoD and DoM features for each image. Then, for each pixel, motion decision is made according to the values of MoD and DoM. If one or both of these features are greater than their corresponding threshold, the pixel is classified as a motion pixel. If both features are lower than their respective thresholds, the pixel is classified as a background. The thresholds are found empirically for each given camera trap data set.

E. K-means Clustering

Our second method of finding motion uses k-means clustering. For daytime images, our feature set includes only the DoM and the MoD. For nighttime images, we increase our feature set by adding the range, mean, and median. These metrics are used as features for the k-means clustering algorithm, and each superpixel is classified as belonging to one of two clusters (foreground or motion and background). Given a set of observations $x_1, x_2, ..., x_n$, where each observation is a d-dimensional vector (in our case the vector of features), k-means clustering aims to partition the n observations into $k <= n$ sets $\mathbf{S} = S_1, S_2, ..., S_k$, minimizing the within-cluster sum of squares. We choose k to be 2, so that our regions are split into two sets, representing motion and background. This algorithm can be expressed mathematically as:

$$\arg\min_{\mathbf{S}} \sum_{i=1}^{k} \sum_{x\in S_i} \|x - \mu_i\|^2$$

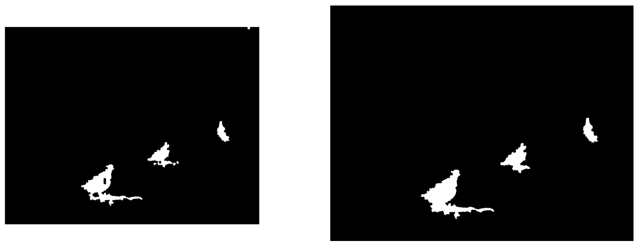
where μ_i is the mean of points in S_i

F. Fuzzy K-means Clustering

Our third method of finding motion uses fuzzy k-means classification. We use the same features as in the case of traditional k-means clustering above. Fuzzy k-means classification returns the probability of whether each superpixel belongs to motion within the image. We chose which classified portion belongs to the foreground by taking the sum of the confidence values for each label over all superpixel regions. The label with larger sum of confidence values is considered the background, and the smaller set is considered the foreground. We then threshold the fuzzy k-means results based on a percentage of the average confidence value of the animal label across the entire image.

G. Method Selection and Postprocessing

We process each image using the three methods described above. To determine which motion detection result is preferred, we assume that the desired motion template should contain a small number of smooth objects and should be noise-free. Therefore, we count the number of objects in the template, analyze the smoothness of the chain code that describes their boundaries, and compute a frequency-based measure of noise in the template. To improve the results from our classification algorithms, we use morphological operations to fill all holes smaller than the size of five superpixels and remove all binary objects smaller than the size of five superpixels. The resulting objects are treated as a mask for the image, with each object corresponding to one animal in the image. The mophological effects can be seen in Figure 10.



(a) Thresholding Results (b) Mask After Morphology

Fig. 4. Development of Image Mask

V. Results of Segmentation

We present examples of our promising results in Figures 5 and 6. For each original image, shown in Fig. 5(a, e) and Fig. 6(a, e), we apply the three proposed methods and show results in the corresponding rows of these two figures. No one method produces the best results for all types of images. In the case of the birds and the ibex images, the feature thresholding method produces superior results. Fuzzy k-means method gives the best results for the daytime snow leopard image while the traditional k-means technique is best in the case of the nighttime snow leopard image. This variability is present across all camera trap images and underscores the need to explore different segmentation methods for this particular application.

VI. Classification

After motion is recognized, classification methods are used to determine whether images contain snow leopards. These methods develop features to use in a machine learning algorithm in order to classify each image as either ”snow leopard” (1) or ”no snow leopard” (0).

A. Spot Recognition

Our algorithm uses a learning method called a Cascade Object Detector, based apon the Viola-Jones algorithm [14]. This object detector recognizes certain image features in

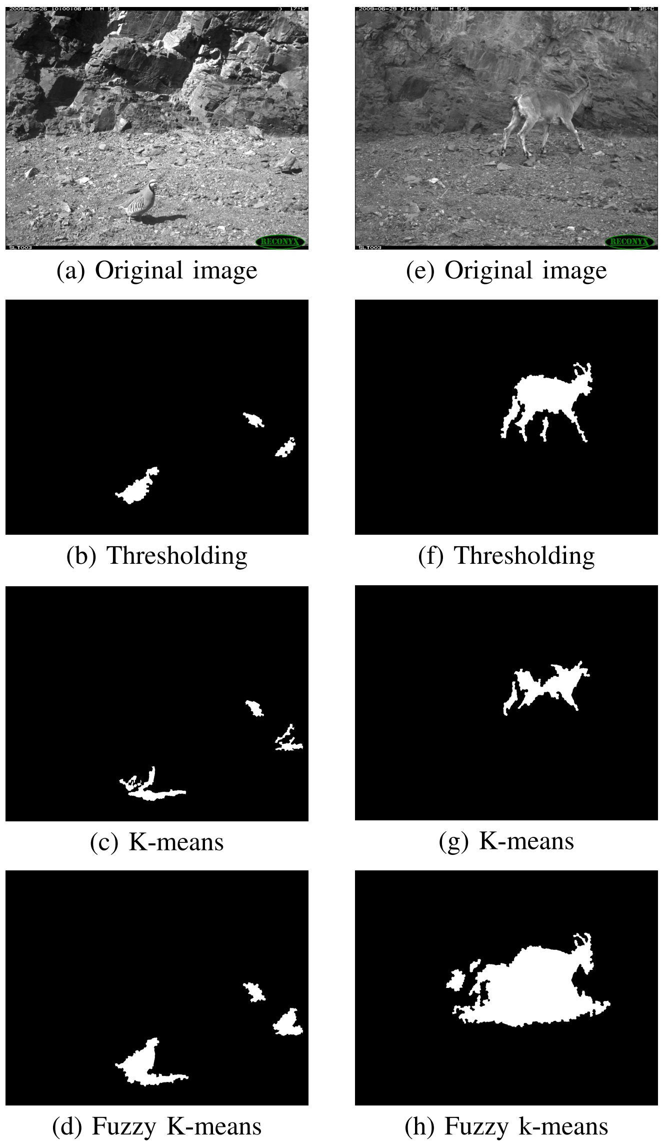


Fig. 5. Results for daytime birds and ibex images.

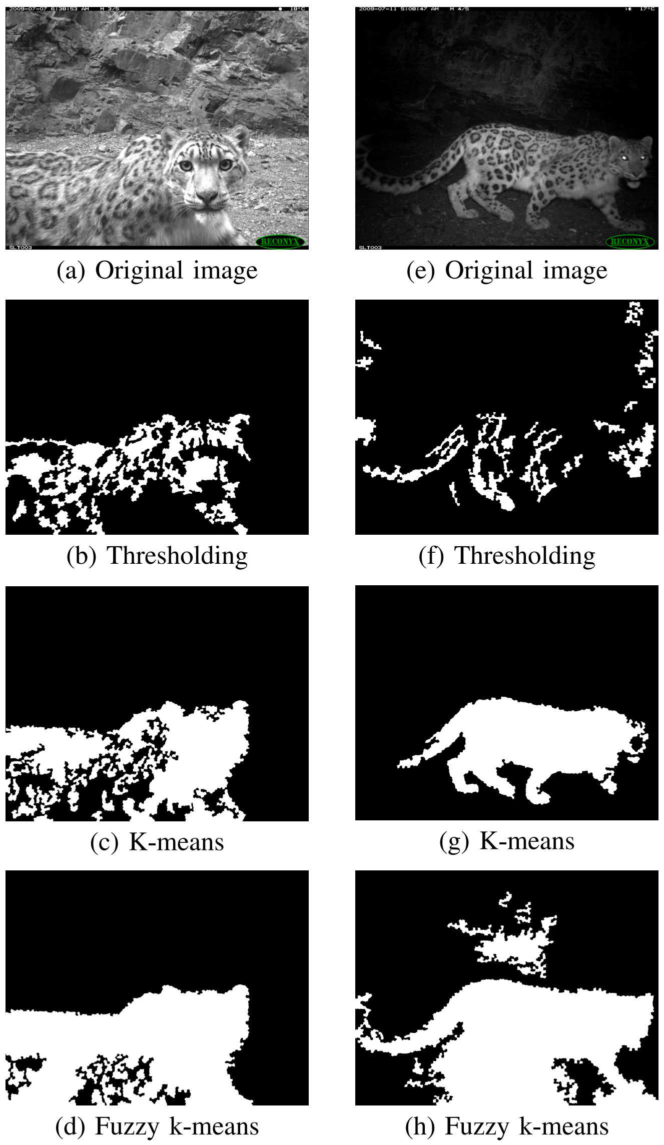


Fig. 6. Results for daytime and nighttime snow leopard images.

similar relationships in order to recognize specific objects in new images robust to differences in scale or illuminations. It is robust to scale, as it looks for similar image feature blocks at cascading scales across the image. The image feature blocks used can be seen in Fig. 7.

However, it is not robust to rotation, so the object detector must to be trained to recognize each possible rotation of a spot. This is obviously an impossible task, and considering that there are many different types and variations on types of spots the accuracy of our spot detector is very dependent on the number of samples it has been trained on. We worked to develop a training set of close to 3000 spots from three different directories of images, which gives us the ability to recognize almost all spots in the image. We get many false positives, where spots are recognized in the background (as they have similar image feature blocks to the shadows around rocks). We minimize the effect of these false positives by only counting the spots that are located within known motion areas in the image.

B. Eye Recognition

We use a similar classifier developed though a cascade object detection algorithm in order to recognize snow leopard eyes at night [14]. The characteristic and specific reflection of light off of predatory eyes from image flash is easily recognizable in images. The presence of one or two eyes in nighttime images are valuable features for machine learning.

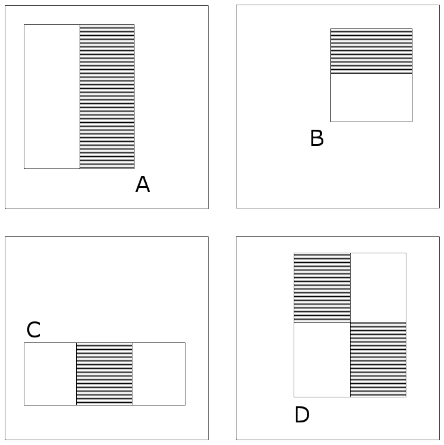


Fig. 7. Features used in the Viola-Jones classification algorithm [14]

C. Machine Learning

We use a Support Vector Machine (SVM) as a machine learning framework for our snow leopard recognition method [13]. This SVM will use information such as number of spots in the image that correspond with areas of motion, number of eyes during night images, and texture characteristics within the areas of motion. The SVM will be trained on a data set by sending in all of these characteristics as well as whether or not each image corresponds to a leopard. It will create delineations between regions in the multidimensional plot data based on whether there is a snow leopard in the image or not. When future images are analyzed, the SVM will predict whether they contain a leopard or not based on the region of the location of that image's data point.

D. Accuracy

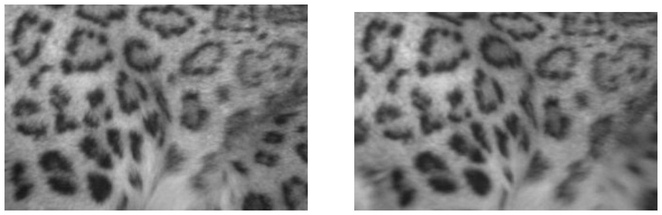
The accuracy of the machine learning algorithm is determined using the following metrics:

Precision = $\frac{TP}{TP+FP}$
Recall = $\frac{TP}{TP+FN}$
Accuracy = $\frac{TP+TN}{TP+TN+FP+FN}$

where TP is the number of true positives, FP is the number of false negatives, TN is the number of true negatives, and FN is the number of false negatives.

VII. Individual Spot Pattern Recognition

Once the images are sorted and the snow leopard images are separated from those containing other animals, we are developing a method of recognizing individual leopards based on spot types and spot relationships. This method will run autonomously on large sets of images and classify each image as belonging to an individual leopard. We use samples of the images from within areas classified as belonging to snow leopards in order to analyze and recognize the same pattern of spots. Examples of the sampled images can be seen in Figure 8.

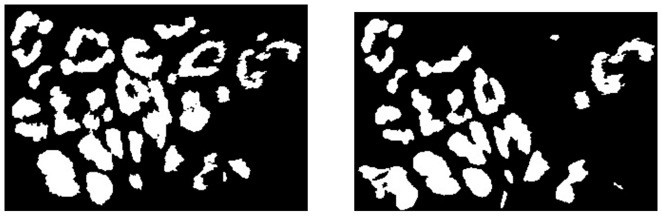


(a)Sample from image 153 (b) Sample from image 155

Fig. 8. Sample of leopard's spot pattern from two similar images

A. Thresholding

The samples of the spot regions of each snow leopard region will be binarized using an adaptive threshold, which can be seen in Figure 9. The individual spots will be treated separately, and placed in a data structure that will maintain their pixel location, the location of their center of mass, the type of spot based on its curvature, and a mathematical model of the spot curve.



(a)Sample from image 153 (b) Sample from image 155

Fig. 9. Spot samples after thresholding

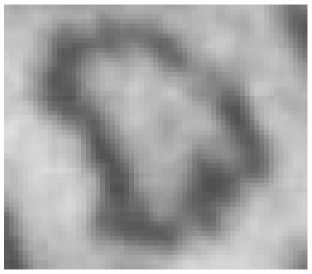
B. Skeletonization

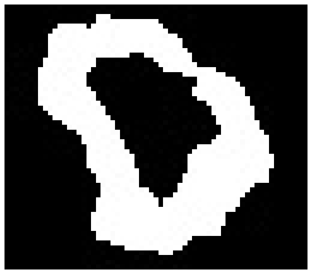
Once the snow leopard motion regions are recognized, the external regions are set to zero and an adaptive threshold is used on the regions corresponding to motion in order to binarize the images with spots as objects. Then each object is separated and saved as an individual object. Each object is skeletonized using a skeletonization algorithm [9] that creates a skeleton of the binary object. This skeleton is developed by storing the locations of the centers all maximal disks within the binary object. A disk B is *maximal* in a set of disks A if $B \subseteq A$ and if another disk D contains B then $D \nsubseteq A$.

Once the skeleton has been developed, all extraneous branches are removed. This is done by removing all branchpoints from the skeleton, and then comparing the length of the branches at each point. The smallest branch is removed, and the branchpoint is replaced. The process is repeated until there are no more branches on the skeleton.

C. Spot Curvature

I am currently using a curvature recognition method based on curve fitting of the skeletonizations of spots [6]. This


(a) Original Spot


(b) Spot Mask

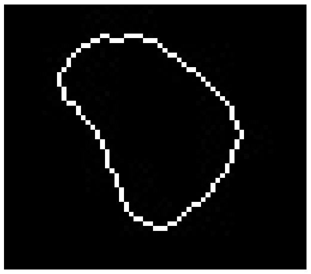

(c) Skeleton Without Branches

Fig. 10. Spot Skeletonization

method uses each pixel locations within the skeleton as data points and then fits a parametric curve through the data. A sinusoidal curve fit is used, which can be replicated along the real line and then shifted in order to match the same curve with parametric data that begins at a different point on the curve.

D. Groth's Algorithm

Groth's algorithm is a pattern recognition method developed by Edward Groth in 1986 for astrophysicists in order to recognize similar patterns of stars in different time zones and from different telescopes. It has been used recently to match patterns on whale sharks in order to identify individual animals [4]. I am developing a MATLAB implementation of Groth's Algorithm to use to recognize spot patterns, with the center of mass of each spot being sent into the algorithm to be compared to previously recognized spot patterns. This is done by comparing the triangle created by every possible combination of three spots within the image, and using the ratios between the sidle lengths and the angles at each corner to match triangles in order to recognize the same triangle at different scales and angles.

1) Triangle Matching: Every possible combination of three spots is considered, where:

- r_2 is the length of the shortest side of the triangle
- r_3 is the length of the longest side of the triangle
- ϵ is the error tolerance, we used 0.01
- R is the triangle ratio: $R = \frac{r_3}{r_2}$
- C is the cosine of the angle at the vertex between the longest and shortest side
- S is the sine of the angle at the vertex between the longest and shortest side

Triangle A and Triangle B are considered a match when:

$$(R_A - R_B)^2 < t_{RA}^2 + t_{RB}^2 \tag{2}$$

and

$$(C_A - C_B)^2 < t_{CA}^2 + t_{CB}^2 \tag{3}$$

where:

$$t_R^2 = 2R^2\epsilon^2\left(\frac{1}{r_3^2} - \frac{C}{r_3 r_2} + \frac{1}{r_2^2}\right)^2 \tag{4}$$

and

$$t_C^2 = 2S^2\epsilon^2\left(\frac{1}{r_3^2} - \frac{C}{r_3 r_2} + \frac{1}{r_2^2}\right)^2 + 3C^2\epsilon^4\left(\frac{1}{r_3^2} - \frac{C}{r_3 r_2} + \frac{1}{r_2^2}\right)^2$$

2) Pattern Matching: Once we have our list of matched triangles, we need to remove potentially false matches. In order to do this, we use a filtering method based on the difference in the logarithm of the perimeter of each set of matched triangle. We define

$$log(p_A) - log(p_B) = logM \tag{5}$$

where $log(p_A)$ is the logarithm of the perimeter of triangle A, and $log(p_B)$ is the logarithm of the perimeter of triangle B. We then calculate the average and standard deviation of the $log(M)$ values for each matched pair of trangles, and remove all triangle matches that have $log(M)$ outside of a factor times the standard deviation of the average.

That factor is determined using the number of clockwise and counter-clockwise defined triangle matches in the set. We let the number of same-sense matches (meaning both triangles are defined in the same direction) be n_+ and the number of opposite-sense matches be n_-. Since all true matches must be the same sense, we can estimate how many of the matches are true and false by defining

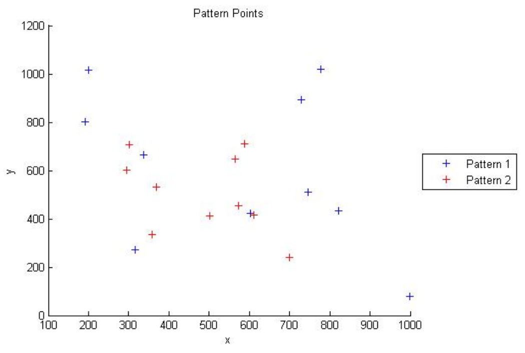
$$m_t = |n_+ - n_-| \tag{6}$$

$$m_f = n_+ + n_- - m_t. \tag{7}$$

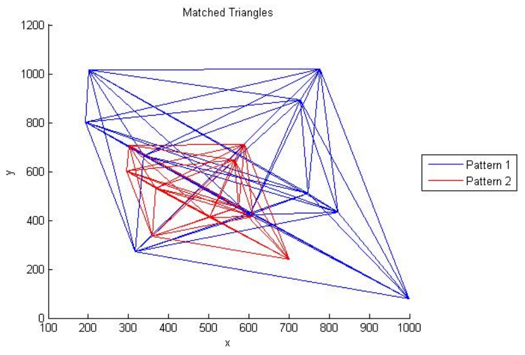
If $m_f > m_t$, we let the factor be 1; if $0.1m_t > m_f$, we let the factor be 3; otherwise, we let the factor be 2. This ensures that matches are discarded at a rate that is comparable to the relationship between m_t and m_f.

We repeat this filtering process until no more matches are discarded, or until all matches are discarded (which will occur when two patterns do not match).

We then use a voting method to decide which points within the patterns match. Each matched triangle casts a vote at each vertex. After all the votes are cast, we sort the vote array from most to least votes. If no points received more than one vote, then the patterns do not match. Otherwise, we match successive pairs of points in the array. until the total vote count drops by a factor of 2.



(a) Points to match



(b) Matched triangles

Fig. 11. Toy problem example of the Groth algorithm

E. *Individual Recognition*

Once the pattern and the individual points have been matched by the Groth algorithm, I plan to iterate through the pattern and verify that the matched spots are of the same or similar type. This ensures that we have accurate matches of the individual leopard spot pattern and not just the pattern of spot locations. The iteration will be done recursively, starting with the spot on the top left of all the matches and then moving to the vetices of the triangles that were matched using that spot. A list of all the matched spots will be maitained, and the spots that have already been matched and verified will be removed from the list until the list is empty or until an incorrect spot type has been found.

VIII. Conclusion and Future Work

In conclusion, the method described will allow conservation biologists to optimize their use of time by removing the necessity of manual image sorting and recognition. This will allow them to process astronomically larger data sets, which will in turn improve the quality of their data analysis.

There is much more work to be done before this method will be put into use. The classification of leopard vs. non-leopard images must be improved, and there are bugs in the proposed pattern recognition method that must be dealt with: primarily, how to deal with the stretching and skewing of the leopard's fur, and therefore the pattern of spots, as the animal moves. This causes the largest number of issues with the current system. The spot type recognition and pattern matching must also be integrated together in order to use the information from both for accurate pattern matching. These questions will continue to be pursued by Dr. Miguel and her future undergraduate research students.

Acknowledgments

I would like to thank all of my advisors for their mentorship throughout the project, the WildBook authors for providing source code for the whale shark annotated recognition algorithm; and Rana Bayrakcismith, Snow Leopard Program Manager at Panthera Corporation for her guidance.The funding for this work has been provided by Center for Environmental Justice and Sustainability at Seattle University (SU), Clare Booth Luce Scholar Foundation, SU Summer Research Fellowship Program, and Francis P. Wood Chair in Electrical and Computer Engineering at SU. Thank you to Panthera, SLT, and SLCF Mongolia for providing the camera trap images used in this research.

References

[1] Radhakrishna Achanta et al. "SLIC superpixels compared to state-of-the-art superpixel methods". In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 34.11 (2012), pp. 2274–2282.

[2] Pablo Arbelaez et al. "Contour detection and hierarchical image segmentation". In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 33.5 (2011), pp. 898–916.

[3] Pablo Arbelaez et al. "From contours to regions: An empirical evaluation". In: *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE. 2009, pp. 2294–2301.

[4] Z. Arzoumanian, J. Holmberg, and B. Norman. "An Astronomical Pattern-matching Algorithm for Computer-aided Identification of Whale Sharks Rhincodon typus". In: *Journal of Applied Ecology* 42.6 (2005), pp. 999–1011.

[5] International Union for the Conservation of Nature. *The IUCN Red List of Threatened Species. Version 2015-14*. http://www.iucnredlist.org. Accessed: 2016-01-08.

[6] A. Fizgibbon, M. Pilu, and R.B. Fisher. "Direct Least Square Fitting of Ellipses". In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 21.5 (1999), pp. 476–480.

[7] H. Hassanpour, M. Sedighi, and A. Manashty. "Video Frames Background Modeling: Reviewing the Techniques". In: *Journal of Signal and Information Processing* 2.2 (2011), pp. 72–78.

[8] R. M. Jackson et al. "Estimating Snow Leopard Population Abundance Using Photography and Capture-Recapture Techniques". In: *Wildlife Society Bulletin* 34.3 (2006), pp. 772–781.

[9] B. Kegl and A. Krzyzak. "Piecewise Linear Skeletonization Using Principal Curves". In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 24.1 (2002), pp. 59–74.

[10] Thomas McCarthy and Guillaume Chapron. *Snow Leopard Survival Strategy*. Seattle, USA: ISLT and SLN, 2003.

[11] K Reddy and R Aravind. "Segmentation of camera-trap tiger images based on texture and color features". In: *Communications (NCC), 2012 National Conference on*. IEEE. 2012, pp. 1–5.

[12] Mikaël Rousson, Thomas Brox, and Rachid Deriche. "Active unsupervised texture segmentation on a diffusion based feature space". In: *Computer vision and pattern recognition, 2003. Proceedings. 2003 IEEE computer society conference on*. Vol. 2. IEEE. 2003, pp. II–699.

[13] Johan AK Suykens and Joos Vandewalle. "Least squares support vector machine classifiers". In: *Neural processing letters* 9.3 (1999), pp. 293–300.

[14] Paul Viola and Michael Jones. "Rapid object detection using a boosted cascade of simple features". In: *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*. Vol. 1. IEEE. 2001, pp. I–511.

[15] Artem Zhelezniakov et al. "Segmentation of Saimaa Ringed Seals for Identification Purposes". In: *Advances in Visual Computing*. Springer, 2015, pp. 227–236.