Extracting identifying contours for African elephants and humpback whales using a learned appearance model

Hendrik J. Weideman
Bloomberg L.P.
hweideman@bloomberg.net

Charles V. Stewart
Rensselaer Polytechnic Institute
stewart@cs.rpi.edu

Jason R. Parham, Jason Holmberg
Wild Me
{parham, jason}@wildme.org

Kiirsten Flynn, John Calambokidis
Cascadia Research Collective
{kflynn, calambokidis}@cascadiaresearch.org

D. Barry Paul, Anka Bedetti, Michelle Henley
Elephants Alive
{barry, anka, michelephant}@elephantsalive.org

Jerenimo Lepirei, Frank G. Pope
Save the Elephants
{frank, jerenimo}@savetheelephants.org

Abstract

This paper addresses the problem of identifying individual animals in images based on extracting and matching contours, focusing in particular on the trailing edges of humpback whale flukes and the outline of the ears of African savanna elephants. A coarse-grained FCNN is learned to isolate the contour in an image, and a fine-grained FCNN is learned to provide more precise boundary information. The latter is trained by generating synthetic boundaries from coarse, easily-extracted training data, avoiding tedious manual effort. An A algorithm extracts the final contour, which is converted to set of digital curvature descriptors and matched against a database of descriptors using local-naive Bayes nearest neighbors. We show that using the learned fine-grained FCNN produces more accurate contours than using image gradients for fine localization, especially for elephant ears where the boundaries are primarily texture. Matching using contours extracted using the fine-grained FCNN improves top-1 accuracy from 80% to 85% for flukes and 78% to 84% for ears.*

1. Introduction

We address the problem of identifying individual animals from images based on extracting and matching distinguishing contours (Fig. 1). In particular, we focus on the trailing edge of a humpback whale fluke and the outer edge of the ear of an African savanna elephant. The technical focus of the paper is learning to extract these contours with sufficient reliability and accuracy to enable identification.

"Photo-identification" of animals using patterns of stripes, spots or texture, appearance of faces, and body outlines [9, 15, 32, 16, 2, 34] is gaining traction as a potential replacement for capture-mark-recapture techniques, which are expensive, labor intensive, and often dangerous [17]. Given the proliferation of inexpensive, high-quality digital cameras, if photo-id can be made sufficiently automated and accurate, it will enable gathering of animal identity data at high resolution in time and space, revolutionizing population biology and conservation studies.

For a variety of reasons photo-id is still a challenging problem. Animals in their natural habitats are uncooperative photo subjects.The image appearance of distinguishing information changes significantly between sightings due to short-term variations in body position, illumination and occlusion, and due to longer term changes in skin condition, scarring, animal maturation, and aging. This problem is exacerbated by the relative sparsity of curated training data, with many individuals in a population appearing in only one or a small number of images. In this paper we introduce and evaluate a deep-learning based algorithm to extract a single identifying contour — which may be trained without repeated sightings of individual animals — and then compute features from this contour for matching.

Our contour extraction problem requires more than application of traditional methods based on intensity gradients [11]. There are two reasons for this. First, the presence of strong distracting gradients that arise from a variety of potential sources, including trees, waves, the horizon, self-occlusion, scars, and skin pigmentation, makes it difficult to isolate the desired contour. Second, the contour may be visually subtle, appearing as a slight change in texture

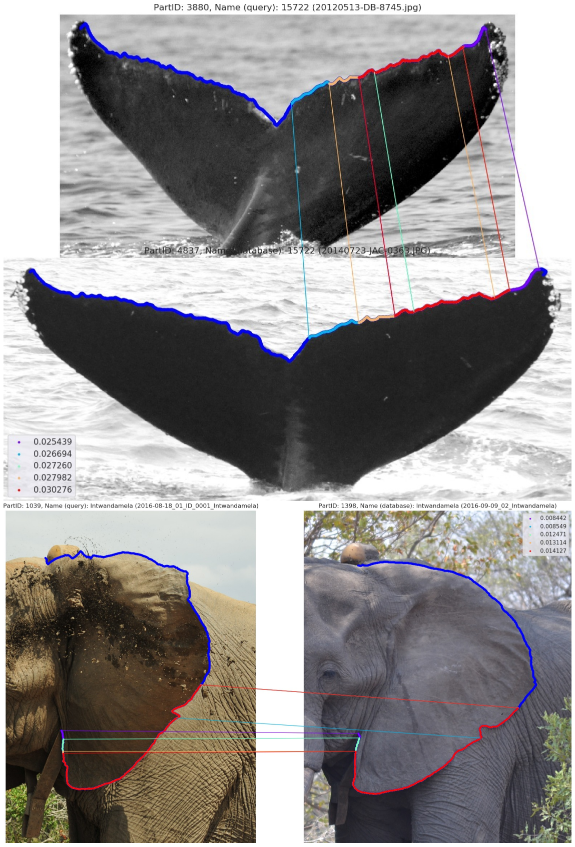
PartID: 3880, Name (query): 15722 (20120513-DB-8745.jpg)
PartID: 4837, Name (database): 15722 (20140723-JAC-0363.JPG)
0.025439
0.026694
0.027260
0.027982
0.030276
PartID: 1039, Name (query): Intwandamela (2016-08-18_01_ID_0001_Intwandamela)
PartID: 1398, Name (database): Intwandamela (2016-09-09_02_Intwandamela)
0.008442
0.008549
0.012471
0.013114
0.014127

Figure 1. Example matching of humpback flukes (top) and elephant ears (bottom). In each case fine-detail contours are extracted using the learned contour model introduced here. Of particular note, the distinguishing notches on the elephant ear appear only as subtle changes in texture. Curvature descriptors are extracted from contour regions and matched using the local naive Bayes nearest neighbor algorithm. Lines and colors (especially red) between images indicate the strongest matching contour segments.

without significant color or intensity gradients. We address these two problems by learning both a coarse and fine contour appearance model: the coarse model is used to suppress distracting information from other contours, while the fine model is used to capture the fine markings that distinguish contours from distinct individuals.

This raises the challenge of extracting training data to drive the learning. We would like to do this without requiring pixel-by-pixel labeling of contour boundaries, an expensive effort akin to the training data needed for semantic segmentation. This is made especially challenging by the subtlety of the actual contours. Requiring such an effort for each new species would limit the practical utility of our algorithms. Instead we propose a self-supervised method for training the contour appearance model based only on coarsely (and easily) traced contour outlines. While our method can exploit dense, high-resolution training data, if available, we are able to show accurate contour extraction and state of the art matching results without it.

The contributions of this paper include:

1. An algorithm for extracting identifying contours based on a learned appearance model that only requires coarse training information.
2. The integration of the results of this algorithm with a matching algorithm based on curvature descriptors and local Naive Bayes nearest neighbor matching.
3. A demonstration of the accuracy of the contour extraction algorithm by comparison with sparsely-extracted ground truth data.
4. Improved recognition for humpback whales and state of the art recognition for African savanna elephants

Two final introductory notes are important. First, we assume a detection algorithm has been trained to locate humpback flukes and elephant ears, placing a bounding box around each [26]. Images cropped to these bounding boxes form the starting point for our work. Second, our complete photo-id algorithm produces a rank-ordered list of the best matching animals from a database of previously-labeled animals. Human users are responsible for final identity decisions. Fully-automatic identification, while important, is beyond the scope of this work.

2. Background

A significant amount of work addresses the problem of automating the photo identification component of an ecological field survey by exploiting identifying markings. These markings include stripe patterns for zebras [8, 9] and toads [25], and the ratio of body part lengths [21] for dolphins. Other methods, including our work, exploit contour markings [2, 32, 15, 16, 34] for identification.

Methods such as DARWIN [31, 32] and Finscan [15] combine edge detection [6] with the active contours algorithm [19] to extract the identifying contour from an image of a dorsal fin. Because contours may be drawn to strong image gradients caused by waves or illumination, points must be repositioned manually. Additionally, the smoothness term used by the active contours algorithm [19] discourages rapid changes of direction in the extracted contour. This conflicts with the goal of accurately representing the jagged nicks and notches that contain identifying information.

A method for extracting dorsal fin contours of sharks is introduced in [16]. A contour map representing likely contour regions is aggregated using [3], before a random forest classifier [5] identifies contour sections belonging to fins. This classifier uses normal and local appearance information [33] from a hand-labeled training set.

In [34], an FCNN trained with pixel-level labels predicts the probability that each pixel in an image is part of the contour. These probabilities are combined with the image gradient to define a cost matrix. Finally, the contour extraction is formulated as a shortest path problem, where the shortest path search is initialized by a neural network trained to predict the start and end points of the contour [18].

In the context of general contour extraction, a contour is defined by a continuous sequence of strong local gradient responses [6], often refined using active contours techniques [19]. To distinguish strong gradient responses from the contour from those of the background, segmentation methods use supervised learning to assign labels to pixels. The level of supervision may vary. For example, in [13, 4, 30], the user labels a region as foreground or background after which the optimal segmentation is computed by minimizing an energy term [12]. This idea is extended by learning the foreground appearance across multiple images from the same class in [1]. It is important to note that methods that rely on global shape are unsuitable for contour extraction in the context of instance recognition, because distinguishing between members of the same population requires an accurate representation of subtle local variations.

3. Learned Contour Extraction

The contour algorithm has three major components. The first two are fully-convolutional neural networks [22] (FCNNs) that each produces a pixel-level probability map. The third is a shortest path contour extraction algorithm that is guided by these maps, similar to [34]. The first FCNN provides coarse grained information about the location of the identifying contour. It is trained by asking annotators to trace a thick brush stroke to cover the contour in each training image, an easily accomplished task. The thickness of the contours learned by the first FCNN prevents us from using them to identify individuals, and therefore the second FCNN provides more precise information about the location of the contour. While training of this fine-grained FCNN (FG-FCNN) could use precisely-annotations of precisely-located boundaries, we show how to train it to accurately locate the boundary from the coarse-grained training data alone. In effect, it learns to recognize the contour boundary without actually having seen a real one.

3.1. Training Data Annotation

Annotators are shown cropped images that frame the body part containing the distinguishing contour — the ear or the fluke. They are asked to trace the entire identifying contour using a single brush stroke, trying to keep the center of the brush close to the true contour as they proceed, but being certain that the contour is covered by the brush stroke. This attempts to balance accuracy against human effort, typically requiring a few minutes for each contour. For a brush with a radius r, this produces a set of points known to be no more than $2r$ from the identifying contour, while ensuring that no point outside this set lies on the true contour. The interface used to collect this coarse grained training data is shown in Figure 2.

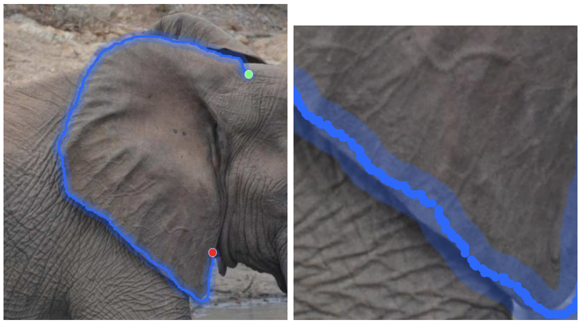

Figure 2. The interface for collecting training data for the coarse appearance model (left) and a close-up view (right).

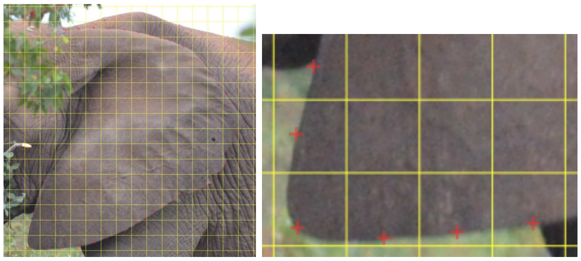

Figure 3. The interface for collecting fine-grained contour points (left) and a close-up view (right). These points are required exclusively for experimental evaluation and are not used in training.

On a subset of images we collect a sparse set of fine-grained contour points. It must be stressed that points are purely for experimental evaluation and are never used for training. Annotators are shown the image with a regularly spaced grid dividing the image into cells. They are asked to find each grid cell that intersects the contour and click the contour point within the cell that is closest to its center. This spreads the samples evenly and avoids bias in the selection of points – e.g. toward notches. This annotation process takes upwards of five to ten times as long as the coarse tracing. The primary reason for this is the existence of regions where the contour is extremely subtle, therefore requiring meticulous effort from the user to separate the contour from the background. The interface used to collect this set of fine-grained contour points is shown in Figure 3.

3.2. Learning the Coarse-Grained Probabilities

The coarse-grained FCNN (CG-FCNN) is trained to predict for each pixel in an image I (of an ear or fluke) the probability that it would be covered by the coarse brush stroke, producing a probability image, C, at the same resolution as I. We employ a U-Net architecture [29] and train the network from random initialization using binary cross-entropy loss. Random rotations are applied to training images and their coarse contours to augment the training data.

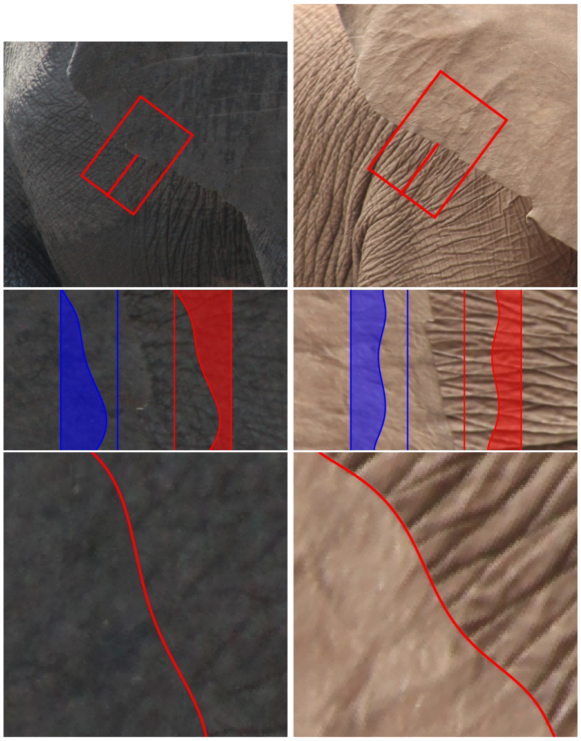

Figure 4. Example generation of synthetic boundaries at a control point for two different ears (left and right). The top shows the initial regions, centered on the control point, and oriented along the normal (red line segment) direction. The middle shows these regions rotated with the normal now horizontal. The blue and red shaded regions, formed from outside the exclusion region between the blue and red lines, are combined along the randomly generated polynomial to form the synthetic boundaries shown at the bottom (zoomed into higher resolution than the middle and top).

3.3. Self-Supervised Learning of the Fine-Grained Probabilities

Supervised training data for the fine-grained FCNN (FG-FCNN) is generated by synthesizing fine-resolution boundary patches from the coarse training data. As illustrated in Fig. 4, at points along the coarse contour, we can step outside the contour's brush region along the perpendicular direction and extract a pair of image regions that we know with high confidence (a) do not intersect the coarse contour and (b) are on opposite sides of the boundary. The synthetic patch is created by overlapping these regions and blending them along a randomly-generated curve. These synthetic boundary patches are used to train the FG-FCNN to predict the probability that a pixel is a boundary pixel.

The first important detail is selecting the "control points" along the coarse contour. These approximate the center of the coarse contour represented by the probability map C. During training, we obtain C from the annotated brush region, with pixels covered by the brush assigned a value of 1 and all others assigned 0. During inference C is the probability map produced by the CG-FCNN. Given C, we compute a distance transform D such that entry d_{ij} in D is the distance from pixel location (i, j) to the closest zero-valued probability pixel in C — pixels confidently labeled as not on the identifying contour. Points on the ridgeline of D become the control points while the direction perpendicular to this ridgeline becomes the normal along which patches are sampled. Examples of these control points are shown as the centers of the oriented rectangles at the top of Fig. 4.

For a particular control point $\mathbf{p}$, we sample a pair of patches opposite each other along the normal at $\mathbf{p}$, excluding the non-zero region of D. This non-zero region is between the blue and red lines in Fig. 4 (center), and the patches are to the left (blue) and right (red) of these lines. The two patches in the pair are then overlapped to form a single rectangular region with a synthetic contour boundary forming the transition between the two combined regions (shaded blue and red regions in the center panels of Fig. 4). The synthetic contour boundary is a linear combination of the first 10 Bernstein polynomial basis functions [23] scaled and stretched to the dimension of the blending region. We randomly sample the coefficients of the basis functions to create a variety of boundary shapes. The y-axis of the generated polynomial is along the normal direction at the contour point and defines the location of the synthetic contour boundary. The alpha-blended transition between the two patches is approximately 4 pixels wide (2 pixels on either side of the contour) to create realistic boundaries.

The FG-FCNN uses the same U-Net architecture [29] as the CG-FCNN, with the pixels in the transition region of the synthetic contour boundary patches playing the role of the brush-stroke pixels in the CG-FCNN. For elephant ears, we use patches of dimensions 256×256 to train the FG-FCNN, while for humpback flukes — that tend to be more rectangular — we use 384×192. We train the FG-FCNN to minimize the cross-entropy loss and L_2 penalty with a coefficient of 10^{-4} by using stochastic gradient descent with a momentum value of 0.9. We have found that initializing the FG-FCNN from the weights of the CG-FCNN and then tuning with a very low learning rate (10^{-5} vs. 10^{-2}) is necessary to obtain good results. Intuitively, the CG-FCNN has already learned the appearance of the region surrounding the contour boundary, but not the contour itself. By initializing the FG-FCNN to the same weights and fine-tuning, we simply train it to suppress those pixels that come from the surrounding region, rather than the contour itself.

When using the FG-FCNN during inference, we crop a patch centered at each control point. The probability map predicted by the FG-FCNN for each patch is used to fill in the corresponding entries in the fine-grained cost matrix. Overlap between probability maps from different control points is handled by interpolation, with Gaussian weighting based on the distance of pixels from control points.

3.4. Extracting the Contours

After generating the coarse and fine probability maps, we combine them into a single cost matrix such that a small entry in the cost matrix corresponds to a pixel with a large contour probability value. If c_{ij} and f_{ij} are the elements of the coarse and fine probability maps at (i, j), respectively, then the corresponding entry w_{ij} in the cost matrix is

$$w_{ij} = \exp\left(\gamma(1 - c_{ij}f_{ij})\right), \tag{1}$$

where the coefficient γ controls the trade-off between traversing a short but expensive region to get to a cheaper region, or avoiding expensive regions altogether. We typically use $\gamma = 5$.

To initialize the shortest path search, we train a neural network that predicts the two end points of the contour [18] based on the endpoints of the hand-traced coarse contours (Sec. 3.1). The A* shortest path algorithm is then used to extract the pixels between these endpoints, guided by cost matrix w. These pixels form the identifying contour.

4. Identification Based on Extracted Contours

After extracting the identifying contour from an image, we need to convert it to a representation for matching. For this, our approach is identical to [34]. Starting from the A* shortest path as an ordered sequence of (x, y) coordinate pairs, we compute an integral curvature representation of the contour by sliding multiple disks of increasing radius along the contour. At each contour point, the ratio of the areas of a given disk on either side of the contour defines the integral curvature at the point for a particular scale. Integral curvature is less sensitive to noise than differential curvature [28], is more robust to changes in viewpoint and pose, and has been shown to be effective for individual identification [18, 34]. Similar to the approach introduced in [16], we define feature keypoints at local extrema of the representation. Between all combinations of pairs of these keypoints we extract a feature descriptor from the corresponding region in the integral curvature representation. Each descriptor is resampled to a fixed length and normalized. The result is a set of overlapping curvature descriptors that densely cover the contour at multiple scales.

For each query image, we combine the feature descriptors extracted from the integral curvature representation with the local naive Bayes nearest neighbors (LNBNN) algorithm [24] to define a ranking of previously-labeled individuals from a database. This method was previously shown to be effective for identifying individuals in [9, 16, 34], because it effectively ignores information common to multiple members of a population and focuses on what distinguishes individuals.

5. Experimental Results

We evaluate the proposed algorithm for contour extraction in the context of its ability to accurately represent the contour and of its effect on the accuracy of the rankings produced by a matching algorithm.

5.1. Data

For humpback whales, we use a real-world photo identification data set provided by the Cascadia Research Collective. This data set contains 3,572 distinct humpback whales across 6,912 encounters.[1] For elephants, we use a real-world photo identification data set provided by Elephants Alive. This data set contains 132 distinct elephants across 508 encounters. An “encounter“ is defined as a set of one or a few images taken of a particular individual at the same time and place. The entire data set is used to evaluate the ranking performance, while a subset of the images are annotated with fine-grained ground truth data as described in Section 3.1. The latter is used for a quantitative evaluation of the contour extraction algorithm. Importantly, for training the FCNNs, we take images from animals and encounters that are distinct from the identification data sets, ensuring a clean separation between training and test sets.

For both humpbacks and elephants, detected regions of interest around the fluke and the ear are resampled while preserving the aspect ratio, producing a width of 1152 pixels for humpback whales and 1024 pixels for elephants. The brush radius for coarse contour training data is 10 pixels. The dimension of the grid cell is 5% of the smaller of the height and width of the region, typically varying between $s = 20$ and $s = 58$ pixels.

5.2. Contour Extraction Results

To evaluate the contour extraction algorithm, we compare the extracted contours to the sparse ground truth. We would like to understand the frequency of missing and spurious contour sections and, where the extracted and ground truth contours are close, the accuracy of the extracted contour. Since there is indeed a reasonable amount of overlap — 90% of humpback flukes and 75% of elephant ears have coverage of at least 90% — the latter is the most important measure because it suggests how well the contour is described for matching. Hence, we focus on accuracy here.

As a baseline for the evaluation, we use the method from [34]. Although the first stage of [34] is very similar to the CG-FCNN, the authors derive the fine-grained cost for the A* algorithm from the gradient magnitude rather than our new FG-FCNN.

[1] This dataset, also used in [34], differs from the recent Kaggle competition https://www.kaggle.com/c/humpback-whale-identification by having at most two encounters per individual.

The challenge in measuring accuracy is the sparsity of the ground truth points. We therefore measure the distance between each ground truth contour point and the closest extracted point, and consider the distribution of these distances. The ideal distribution would be a step function. For each contour i that has ground-truth points, let $\mathcal{G}_i$ be these points, and let $\mathcal{H}_i$ be the set of contour points extracted using the proposed algorithm. For (sparse) ground-truth point $\mathbf{x}$, let $d(\mathbf{x}, \mathcal{H}_i)$ be the distance from $\mathbf{x}$ to the closest point on the extracted contour. Letting s be the aforementioned dimension of each grid cell, to measure accuracy we restrict our attention to the subset of $\mathcal{G}_i$ where $d(\mathbf{x}, \mathcal{H}_i) < s$, and refer to this subset as $\mathcal{G}'_i$. Ground-truth points in $\mathcal{G}'_i$ are the locations where we measure the accuracy of the extracted contour "true positives". For contour i we form the distribution as the function

$$FTP(i, \delta) = \frac{1}{|\mathcal{G}'_i|} \Big| \{\mathbf{x} \in \mathcal{G}'_i \mid d(\mathbf{x}, \mathcal{H}_i) \leq \delta\} \Big|. \quad (2)$$

This is the "fraction of true positive" contour cells in the grid where the extracted contour passes within δ of the cell's ground-truth point. By computing the means of this measure over all N ground truth contours, we obtain our summary distribution:

$$MFTP(\delta) = \frac{1}{N} \sum_{i=1}^{N} FTP(i, \delta). \quad (3)$$

Function $MFTP$ is plotted for our new algorithm and for the gradient-driven baseline in Fig. 6 for humpback whale flukes and Fig. 7 for elephant ears. Fig. 5 illustrates the significance of various values of δ on a fluke. For very small distances, i.e., $\delta < 3$, using the baseline with the image gradient magnitude outperforms the new algorithm using the FG-FCNN. This is actually expected because whenever the true contour coincides with sharp intensity discontinuities, we should expect the image gradient magnitude to provide a more reliable signal than the learned FG-FCNN contour model. This occurs more often for humpback flukes, which have frequent high contrast boundaries against the water or the sky, than it does for elephants. Above these small distances, when $\delta \geq 3$, using the FG-FCNN has the same cumulative level of accuracy as the gradient-based method for flukes and produces substantially better results for elephants. The range $\delta = 3$ to $\delta = 5$, where the FG-FCNN results catch up to (and pass for ears) the gradient-based results, is particularly important because beyond this we begin to see subtle switches between following the correct and incorrect contours (see Fig. 5). Examples illustrating this are shown in Figs. 8 and 9 where the FG-FCNN enables the contour extraction algorithm to distinguish between the true contour and distracting gradients — changes in skin pigmentation on flukes, and leaves and branches in the immediate background for elephant ears — keeping the extracted contour close to the true boundary.

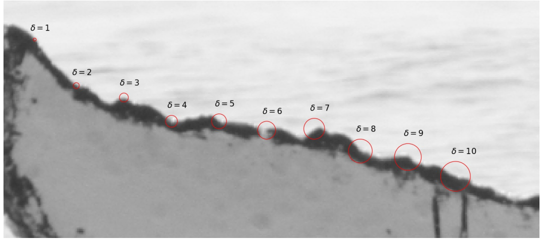

Figure 5. For each circle, all points inside are closer than or equal to the center than the given value of δ.

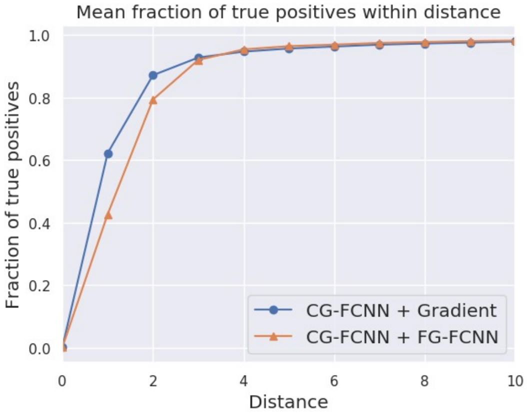

Figure 6. The mean fraction of *true positives* within a given distance for humpback whales.

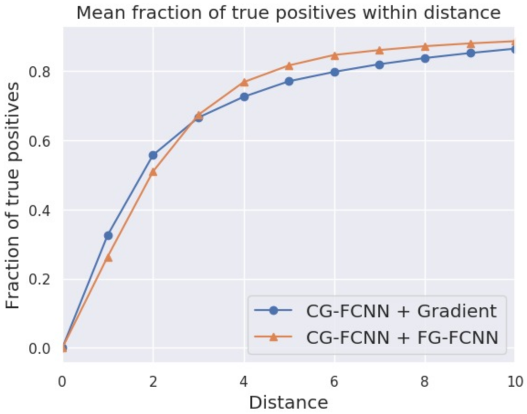

Figure 7. The mean fraction of *true positives* within a given distance for elephants.

We conclude that use of the FG-FCNN produces nearly equivalent numerical results to gradient-based methods for humpback flukes, better numerical results for the more subtle contours outlining elephant ears, and often successfully avoids errors due to following incorrect contours with strong gradients in both cases.

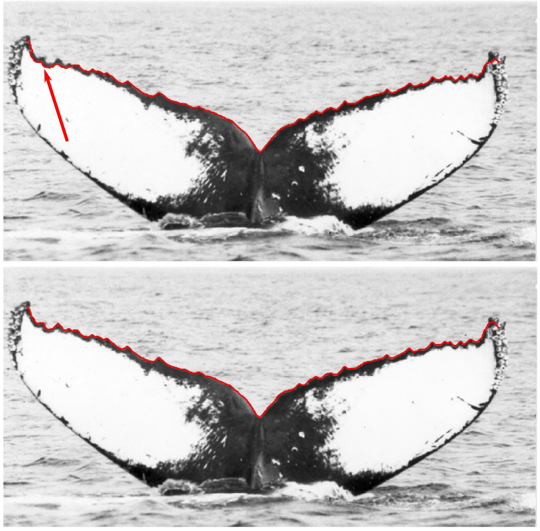

Figure 8. The fluke contour extracted using the image gradient (top) and using the FG-FCNN (bottom). The red arrow indicates a section of the contour where the image gradient method followed pigmentation rather than the actual fluke boundary.

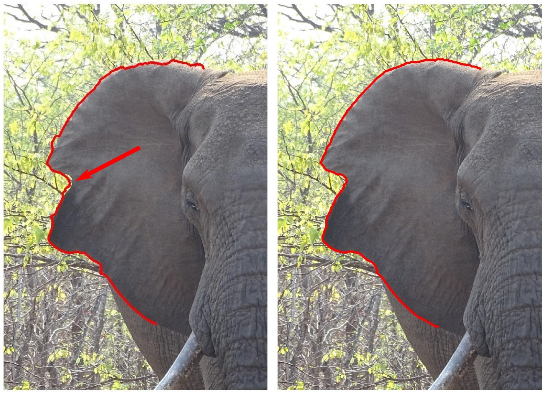

Figure 9. The ear contour extracted using the image gradient (left) and using the FG-FCNN (right). The red arrow indicates a section of the contour where the gradient-based method followed strong background signal, but the method using the FG-FCNN stayed close to the true contour.

5.3. Identification Results

The final test of the significance of the new contour extraction method is its impact on matching performance. Since for each query image, the result of matching is a ranked list of the potentially matching individuals in the database, we plot the cumulative match characteristic (CMC) curve — the fraction of queries for which the correct match has rank $\leq k$, for $k = 1, 2, 3, \ldots$.

The baseline results for humpbacks are computed using the work from [34], which combines a method similar to the CG-FCNN from this work with gradient magnitude information for contour extraction. For elephants, the algorithm from [34] serves as one baseline, but we also include recently reported results from [20] as a second. This algorithm uses a ResNet50 [14] architecture pretrained on ImageNet [10] to extract a feature vector from a bounding box placed around an elephant's head. These feature vectors are used for classification by means of dimensionality reduction [27] and a support vector machine [7]. In using this algorithm here, we restrict its use to identification based on the ear, providing a direct comparison between algorithms.

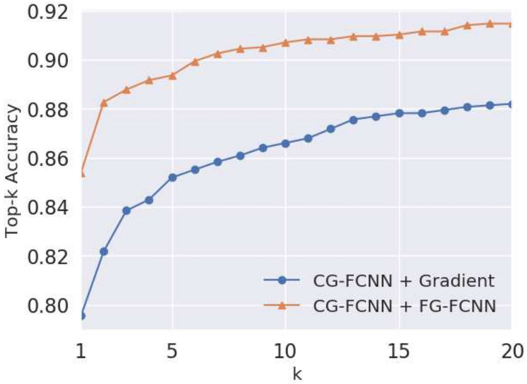

Figure 10. Using the FG-FCNN, which learns a more sophisticated contour appearance model, instead of the gradient improves the top-1 accuracy from 80% to 85% and the top-5 accuracy from 85% to 89% when using the LNBNN matching algorithm for humpback whales.

Figures 10 and 11 plot the CMC curves for humpback flukes and elephant ears, respectively, for matching based on the FG-FCNN contours, and for the baseline algorithms. This shows that for both humpback whales and elephants replacing the gradient-based term with the FG-FCNN improves the ranking performance. For humpback whales the top-1 accuracy improves from 80% to 85% and the top-5 accuracy from 85% to 89%, while for elephants the top-1 accuracy improves from 78% to 84% and the top-5 accuracy from 88% to 93%. We attribute this to the ability of the contour extraction algorithm based on the FG-FCNN to stay close to the true contour and avoid distracting information that distorts the identifying information. Interestingly, this occurs even for humpback flukes where the numerical performance of the two contour extraction methods is essentially equivalent. Figure 12 shows an example where the matching algorithm correctly identifies an elephant when the contours are extracted using the FG-FCNN, but not when they are extracted using the gradient.

For elephants, these methods outperform the non-contour baseline from [20], which achieves a top-1 accuracy of 34% and a top-5 accuracy of 63% on our data set. One reason for this is that the data set is small and unbalanced with respect to the number of images per individual. This makes it difficult to learn a representation that is invariant to a wide range of appearance changes, such as coverage

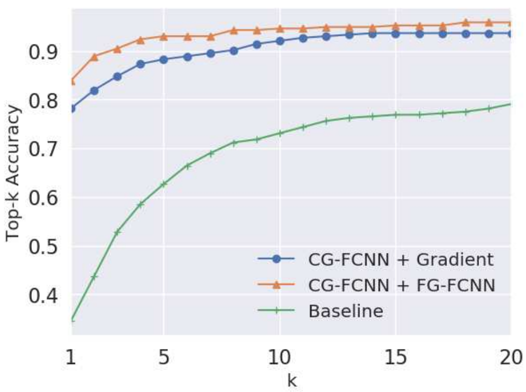

Figure 11. Using the FG-FCNN, which learns a more sophisticated contour appearance model, instead of the gradient as in [34] improves the top-1 accuracy improves from 78% to 84% and the top-5 accuracy from 88% to 93% when using the LNBNN matching algorithm for elephants.

by mud or water, illumination variations, and deformations. Unfortunately, sample imbalances are almost inevitable for many wild animal populations. Our algorithm does better in this regard, but is still limited as our top-1 matching rates rise quickly from under 40% with a single encounter in the database to nearly 80% for three. This is not as much of an issue for humpback flukes where we achieve 85% top-1 rates despite only having one database encounter per animal. Clearly, elephant ear recognition is currently more difficult, in part due to ongoing challenges of contour extraction and in part because the identifying information is more localized and subtle. An important avenue of future work is to combine the identifying information from the ears with identifying information from other parts of the elephant.

6. Summary and Conclusion

We have developed an algorithm for learning a fine-grained appearance model for contours that distinguish individual animals, training it using boundaries synthesized from coarse annotation data. The model captures boundary information from transitions in color and texture as well as intensity. We have integrated the model into a complete contour extraction algorithm that also includes a coarse-grained contour model and an A* search algorithm. The contours produced by this algorithm are more accurate than the contours produced using gradient information, especially for the subtle boundaries of elephant ears. When integrated into an existing matching algorithm based on curvature descriptors and LNBNN matching, these contours produce approximately 5% improvement in top-1 ranking results for both humpback whale flukes and the ears of African savanna elephants. Matching works despite having a small number of encounters per individual animal, an important consideration for real-world use.

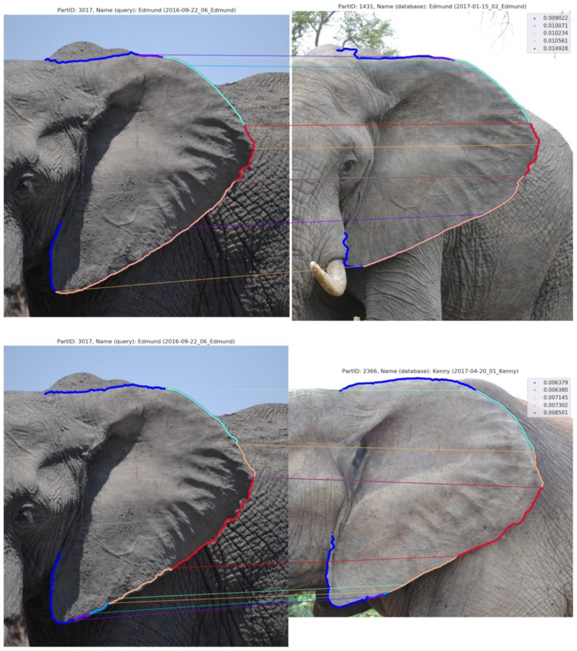
PartID: 3017, Name (query): Edmund (2016-09-22_06_Edmund)
PartID: 1431, Name (database): Edmund (2017-01-15_02_Edmund)
0.009022
0.010071
0.010234
0.010561
0.014928
PartID: 3017, Name (query): Edmund (2016-09-22_06_Edmund)
PartID: 2366, Name (database): Kenny (2017-04-20_01_Kenny)
0.006379
0.006380
0.007145
0.007302
0.008501

Figure 12. The top row shows a query image (left) matched to the correct individual from the database (right) when using the FG-FCNN for contour extraction, while the bottom row shows the same query image (left) matched to an incorrect individual (right) when using the gradient.

Acknowledgements

This work was completed while the first author was at Rensselaer Polytechnic Institute (RPI). Weideman and Stewart would like to thank H2O.ai, Save the Elephants, Wild Me, and the RPI School of Science for their financial support, and Cascadia, Elephants Alive, and Save the Elephants for providing the image data. Flynn and Calambokidis were supported by subaward 16-08003 from WDFW under NOAA Sect 6 grant NA16NMF4720061. They would like to thank Ian Flynn-Thomas and Elana Dobson for their work on image annotation. Paul, Bedetti and Henley would like to thank Save the Elephants, Transfrontier Africa and in particular Craig Spencer, The Oak Foundation and the USFWS for long-term support of Elephants Alive, and thank APNR and all private landowners for logistical support. Finally, they would also like to thank Amy Clark, Cathy Greyling, Christin Winter, Hannah Malin, Jennifer Blondeau, Jessica Wilmot, Joel Sithole, Julie Kern, Michellene Munro, Robin Cook, Ronny Makukule, Scott Chui and Tammy Eggeling for long hours collecting and processing photos. Lepirei and Pope would like to thank their Save the Elephants team, the Samburu County Government, Isiolo County Government and the Kenya Wildlife Service.

References

[1] B. Alexe, T. Deselaers, and V. Ferrari. Classcut for unsupervised class segmentation. In *Eur. Conf. on Comput. Vision*, pages 380–393, 2010. 3

[2] B. Araabi, N. Kehtarnavaz, T. McKinney, G. Hillman, and B. Würsig. A string matching computer-assisted system for dolphin photoidentification. *Ann. of Biomed. Eng.*, 28(10):1269–1279, Oct. 2000. 1, 2

[3] P. Arbeláez, J. Pont-Tuset, J. T. Barron, F. Marques, and J. Malik. Multiscale combinatorial grouping. In *Comput. Vision and Pattern Recognition*, pages 328–335, 2014. 2

[4] Y. Y. Boykov and M.-P. Jolly. Interactive graph cuts for optimal boundary & region segmentation of objects in nd images. In *Int. Conf. on Comput. Vision*, volume 1, pages 105–112, 2001. 3

[5] L. Breiman. Random forests. *Mach. Learning*, 45(1):5–32, Oct. 2001. 2

[6] J. Canny. A computational approach to edge detection. *IEEE Trans. on Pattern Anal. and Mach. Intell.*, (6):679–698, Nov. 1986. 2, 3

[7] C. Cortes and V. Vapnik. Support-vector networks. *Mach. Learning*, 20(3):273–297, Sept. 1995. 7

[8] J. P. Crall. *Identifying Individual Animals Using Ranking, Verification, and Connectivity*. PhD thesis, Department of Computer Science, Rensselaer Polytechnic Institute, Troy, NY, 2017. 2

[9] J. P. Crall, C. V. Stewart, T. Y. Berger-Wolf, D. I. Rubenstein, and S. R. Sundaresan. Hotspotter — patterned species instance recognition. In *Winter Conf. on Appl. of Comput. Vision*, pages 230–237, 2013. 1, 2, 5

[10] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A large-scale hierarchical image database. In *Comput. Vision and Pattern Recognition*, pages 248–255, 2009. 7

[11] R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis*. Wiley, New York, NY, 1973. 1

[12] L. R. Ford Jr and D. R. Fulkerson. A suggested computation for maximal multi-commodity network flows. *Manage. Sci.*, 5(1):97–101, Oct. 1958. 3

[13] D. M. Greig, B. T. Porteous, and A. H. Seheult. Exact maximum a posteriori estimation for binary images. *J. of the Roy. Statistical Soc. Series B (Methodological)*, 51(2):271–279, Jan. 1989. 3

[14] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Comput. Vision and Pattern Recognition*, pages 770–778, 2016. 7

[15] G. Hillman, N. Kehtarnavaz, B. Würsig, B. Araabi, G. Gailey, D. Weller, S. Mandava, and H. Tagare. 'Finscan', a computer system for photographic identification of marine animals. In *Eng. in Med. and Biol. Soc.*, volume 2, pages 1065–1066, 2002. 1, 2

[16] B. Hughes and T. Burghardt. Automated visual fin identification of individual great white sharks. *Int. J. of Comput. Vision*, 122:542–557, Sept. 2016. 1, 2, 5

[17] A. Irvine, R. Wells, and M. Scott. An evaluation of techniques for tagging small odontocete cetaceans. *Fishery Bull.*, 80(1):135–143, 1982. 1

[18] Z. Jablons. Identifying humpback whale flukes by sequence matching of trailing edge curvature. Master's thesis, Department of Computer Science, Rensselaer Polytechnic Institute, Troy, NY, 2016. 3, 5

[19] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *Int. J. of Comput. Vision*, 1(4):321–331, Jan. 1988. 2, 3

[20] M. Körschens, B. Barz, and J. Denzler. Towards automatic identification of elephants in the wild. In *Federated Artificial Intell. Meeting Workshops*, 2018. 7

[21] A. Kreho, N. Kehtarnavaz, B. Araabi, G. Hillman, B. Würsig, and D. Weller. Assisting manual dolphin identification by computer extraction of dorsal ratio. *Ann. of Biomed. Eng.*, 27(6):830–838, Nov. 1999. 2

[22] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Comput. Vision and Pattern Recognition*, pages 3431–3440, 2015. 3

[23] G. G. Lorentz. *Bernstein Polynomials*. American Mathematical Society, Providence, RI, 2012. 4

[24] S. McCann and D. G. Lowe. Local naive Bayes nearest neighbor for image classification. In *Comput. Vision and Pattern Recognition*, pages 3650–3656, 2012. 5

[25] T. A. Morrison, D. Keinath, W. Estes-Zumpf, J. P. Crall, and C. V. Stewart. Individual identification of the endangered Wyoming toad Anaxyrus baxteri and implications for monitoring species recovery. *J. of Herpetology*, 50(1):44–49, Mar. 2016. 2

[26] J. Parham, C. Stewart, J. Crall, D. Rubenstein, J. Holmberg, and T. Berger-Wolf. An animal detection pipeline for identification. In *Winter Conf. on Appl. of Comput. Vision*, pages 1075–1083, 2018. 2

[27] K. Pearson. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosoph. Mag. and J. of Sci.*, 2(11):559–572, Nov. 1901. 7

[28] H. Pottmann, J. Wallner, Q.-X. Huang, and Y.-L. Yang. Integral invariants for robust geometry processing. *Comput. Aided Geometric Des.*, 26(1):37–60, Jan. 2009. 5

[29] O. Ronneberger, P. Fischer, and T. Brox. U-Net: Convolutional networks for biomedical image segmentation. In *Int. Conf. on Med. Image Comput. and Comput.-Assisted Intervention*, pages 234–241, 2015. 3, 4

[30] C. Rother, V. Kolmogorov, and A. Blake. Grabcut: Interactive foreground extraction using iterated graph cuts. In *ACM Trans. on Graphics (TOG)*, volume 23, pages 309–314, 2004. 3

[31] R. Stanley. *DARWIN: identifying dolphins from dorsal fin images*. Bachelor's thesis, Department of Computer Science, Eckerd College, St. Petersburg, FL, 1995. 2

[32] J. Stewman, R. Stanley, and M. Allen. DARWIN: A system to identify dolphins from fin profiles in digital images. In *Proc. 8th Annu. Florida Artificial Intell. Res. Symp.*, 1995. 1, 2

[33] K. Van De Sande, T. Gevers, and C. Snoek. Evaluating color descriptors for object and scene recognition. *IEEE Trans. on Pattern Anal. and Mach. Intell.*, 32(9):1582–1596, Aug. 2010. 2

[34] H. J. Weideman, Z. M. Jablons, J. Holmberg, K. Flynn, J. Calambokidis, R. B. Tyson, J. B. Allen, R. S. Wells, K. Hupman, K. Urian, et al. Integral curvature representation and matching algorithms for identification of dolphins and whales. In *Int. Conf. on Comput. Vision Workshops*, pages 2831–2839, 2017. 1, 2, 3, 5, 7, 8