A Comparison of Background Subtraction Algorithms for Detecting Avian Nesting Events in Uncontrolled Outdoor Video

Kyle Goehner, Travis Desell
Department of Computer Science
University of North Dakota
Grand Forks, North Dakota 58202
Email: kyle.goehner.2@my.und.edu, tdesell@cs.und.edu

Rebecca Eckroad, Leila Mohsenian, Paul Burr,
Nicholas Caswell, Alicia Andes, and Susan Ellis-Felege
Department of Biology
University of North Dakota
Grand Forks, North Dakota 58202
Email: rebecca.eckroad@outlook.com,
leila.mohsenian@my.und.edu, pcb124@msstate.edu,
nicholas.caswell@my.und.edu, alicia.andes@my.und.edu,
susan.felege@email.und.edu

***Abstract*—This paper examines the use of three different background subtraction algorithms — Mixture of Gaussians (MOG), ViBe, and Pixel-Based Adaptive Segmentation (PBAS) — to detect events of interest within uncontrolled outdoor avian nesting video for the Wildlife@Home project. Many computer vision techniques are unsuccessful in this domain due to low framerates and resolution of battery powered surveillance cameras in combination with the cryptic coloration (camouflage) of the animals. Modifications to ViBE and PBAS are presented which provide more robust results in this challenging video, and address issues caused by the cryptic coloration of the species being monitored by the project. These algorithms were run on over 250 hours of video and compared to human observations generated by Wildlife@Home's project scientists and volunteer citizen scientists. All three algorithms provide accurate detection of events however we see much fewer false postives from the modified versions of the ViBe and PBAS algorithms. This is especially true for Interior Least Tern (*Sternula antillarum*) and Piping Plover (*Charadrius melodus*) video, which do not suffer from as much moving vegetation as the Sharp-Tailed Grouse (*Tympanuchus phasianellus*) footage. These results provide initial justification for utilizing Wildlife@Home's 2,000+ volunteered computers to analyze the project's 85,000 hours of avian nesting video, so that this information can be integrated into the Wildlfe@Home user interface. Further, the videos and human observations used to test these algorithms have been made available as part of Wildlife@Home's first data release, to encourage future study by computer vision researchers.**

I. Introduction

Wildlife@Home[1] [1], [2] is a volunteer computing project in which citizen scientists and wildlife experts are presented videos taken from at the nests of various species of birds. Currently, users have the option of viewing Sharp-Tailed Grouse (*Tympanuchus phasianellus*, an *indicator species* which can represent the ecological health of a region), Interior Least Tern (*Sternula antillarum*, a federally endangered species), or Piping Plover (*Charadrius melodus*, a federally threatened species). Each of these species have different nesting behaviors and users are tasked with classifying them. Examples of behaviors are *On Nest*, *Off Nest*, *Brooding*, *Flying*, *Foraging*, and *Feeding*. While users are observing the nets, they create a time-series for each video specifying when these events begin and end. Each event in the time-series has a type, start time and end time (see Figure 1).

Fig. 3. Sample Sharp-Tailed Grouse footage. The nest is marked with a white oval.

Such camera studies are popular in the field of avian ecology as they can reduce researcher impacts on animal behavior and also monitor animals in remote locations [3], [4]. Unfortunately, many of these studies are hampered by small sample sizes, where few have studied more than 100 nests [4], limiting the biological inferences that can be made. In order to overcome these challenges, Wildlife@Home has been developed to employ both volunteer computing and crowd sourcing to quickly analyze wildlife video, as well as to investigate automated video analysis strategies using computer vision techniques.

The Wildlife@Home project has accumulated over 85,000 hours of 24/7 uncontrolled outdoor surveillance video. This amount of data becomes problematic for humans to classify, even with software tools to help create and store event data.

[1] http://volunteer.cs.und.edu/csg/wildlife/

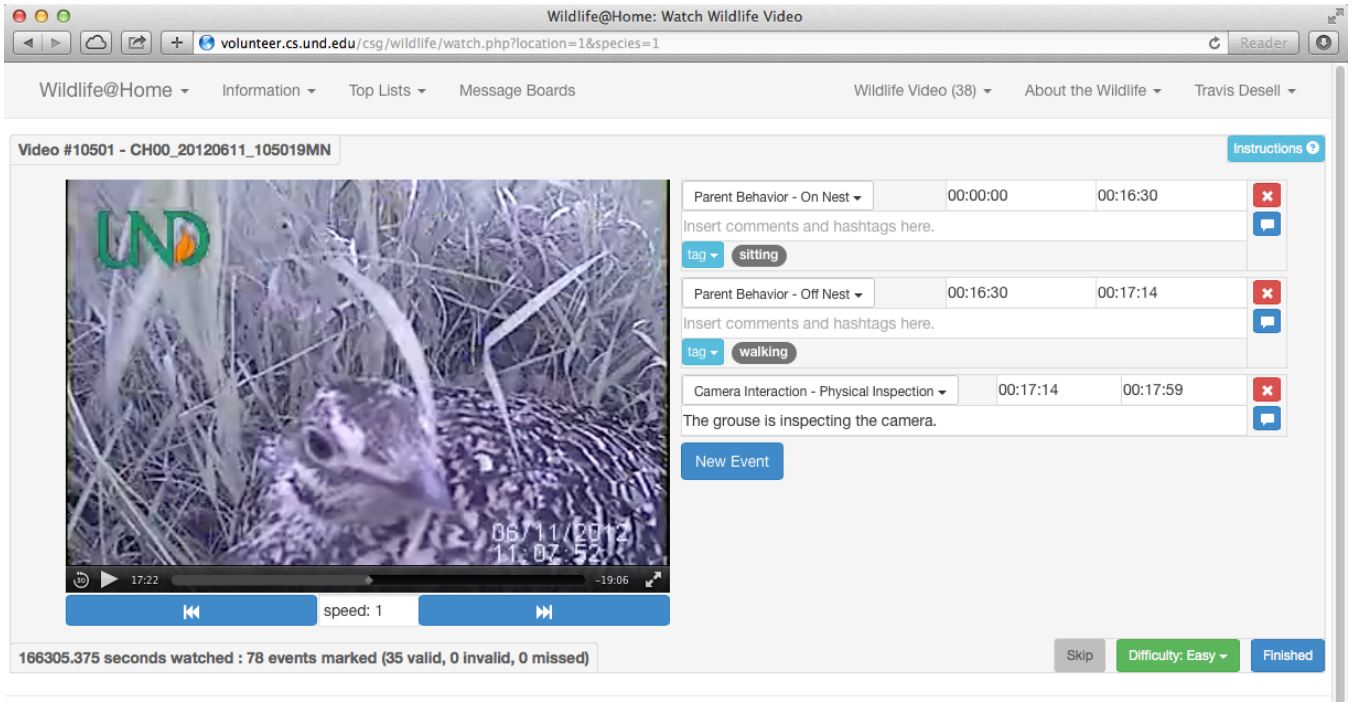


Fig. 1. An example of Wildlife@Home's video viewing interface. Users are shown 30 minute to 2 hour long nesting videos, and can specify the start and end time for various events of interest, and provide tags and comments for additional detail. Users can also specify how difficult it was to determine events for the video and discuss segments of the video on the project's message boards.

(a) Sample I (b) Sample II (c) Sample III

Fig. 2. Sample sunrise Interior Least Tern footage. The nest is marked with a white oval.

A lot of time is spent viewing regions of the video where the birds are not present at all or where a bird is present but highly inactive for long periods of time. This can lead to problems with scientist precision and focus. Users watching the video use the scrub bar to move more quickly through the video, especially uninteresting potions, and this can cause missed events. Scientists tasked with classifying long periods of uninteresting video can more quickly tire and lose focus.

This paper investigates the use of various background subtraction techniques for the detection of avian nesting behaviors, especially in highlighting interesting or active portions of the video. Background subtraction is commonly used in surveillance video as a technique for segmenting objects of interest from a scene [5], [6]. By extracting segments of the collected video with an abnormal amount of foreground activity, it is possible to algorithmically present scientists with video containing classifiable events and filter out video where no events occur.

Given the diversity of species and nest locations, preliminary results find that algorithm performance is highly dependent on the amount of background movement, camera brightness, and cryptic coloration in a video. Using modern background subtraction techniques, such as Mixture of Gaussians (MOG) [7], and modified versions of the ViBe [8] and Pixel-Based Adaptive Segmentation (PBAS) [9] algorithms, it is possible to show a strong correlation between scientist observed events and those calculated with background subtraction. By confidently narrowing the amount of video scientists are watching, it will be possible to focus on showing worthwhile video segments and increase user incentive and focus.

Section II presents modern techniques used for common background subtraction problems. Section III covers the approach we took to extracting regions of the video with activity and how this is implemented. Performance results and limitations of the algorithms are in described Section IV. The data release and open source code developed for this work are given in Section V. Finally, Section VI concludes with future work and a discussion of the next steps to collecting more results, improving the algorithms, and use of the data.

II. Related Work

This section discusses approaches for background subtraction, or as it is sometimes referred, foreground segmentation. The most common methods along with more modern approaches are presented. This includes the running Gaussian average (II-A), Mixture of Gaussians (II-B), ViBe (II-C), pixel-based adaptive segmentation (II-D), and a couple techniques used in a similar problem domain (II-E).

A. Running Gaussian Average

Running Gaussian average is one of the most basic background subtraction techniques [5], [6] and has also been effective in applications with a static background such as traffic cameras [10], [11]. This technique works by storing a model of the background $\mathbf{B}_t$ and calculating the distance of each new image $\mathbf{I}_t$ from the background model. If this distance is larger than a provided threshold, τ, then the pixel at that location is marked as foreground. This threshold can be seen in Equation 1.

$$|\mathbf{I}_t - \mathbf{B}_t| < \tau \tag{1}$$

The background model can then be updated by using an exponential moving average which slowly adapts to changes:

$$\mathbf{B}_{t+1} = \alpha \cdot \mathbf{I}_t + (1 - \alpha) \cdot \mathbf{B}_t \tag{2}$$

Where α is the rate at which the model adjusts and t is the current frame index.

There are a few effective methods for cleaning the results from a simple running Gaussian average as pointed out in [5]. The first is to clean up the foreground mask with some type of filter. Both a median filter and a open/close filter work well. If a pixel has been marked as foreground for too many consecutive frames it can be set in the background model to prevent long standing false detection in the vent of a sudden lighting change. Finally if a pixel is rapidly changing from foreground to background it can be masked to prevent sporadic and unreliable detection.

B. Mixture of Gaussians (MOG)

MOG is a widely used and robust background subtraction algorithm used in OpenCV [12]. It is based on modeling the background pixels as a combination of surfaces [7] which is further described as a Gaussian mixture model. The probability of a pixel belonging to the background is described as a sum of Gaussians:

$$f_{\mathbf{X}}(X|\Phi) = \sum_{k=1}^{K} P(k) \cdot f_{\mathbf{X}|k}(X|k, \theta_k) \tag{3}$$

where $P(k)$ is the probability of the surface k appearing in the pixel view and $f_{\mathbf{X}|k}(X|k, \theta_k)$ is the Gaussian distribution for surface k with Φ being the set of theta input parameters ($\theta_k = \mu_k, \sigma_k$) for the Gaussian distributions describing each surface.

Power and Schoonees note that $P(k)$, μ_k, and θ_k are typically estimated with running averages calculated at each frame [7]. Also, $f_{\mathbf{X}|k}(X|k, \theta_k)$ for a pixel value x can be estimated by a Boolean value, true if x is within 2.5 standard deviations of the mean, false otherwise.

With MOG, similar techniques to those in Section II-A can be used to clean results. The use of an open/close filter is especially useful for removing noise.

C. ViBe

ViBe [8] is a background subtraction algorithm based on random substitution and spatial diffusion. Van Droogenbroeck *e*t al. approach background model formulation with stochasticity in order increase the robustness of their algorithms and increase the range of background pixels stored in the model. Since ViBe does not rely on statistical modeling of pixel history the authors believe it can better match a pixels true history by actually using past pixel values. This means ViBe can fit multi-modal pixel histories and better adapt to slight background movement.

To model the background, ViBe stochastically stores 20 previous pixel values and compares new pixel values to this pixel history. If a pixel value matches (see Equation 1) two of the stored values then it is classified as part of the background, otherwise it is masked as foreground. This method of classification allows for up to 10 different background models to be fit by ViBe.

As alluded to earlier, updating the background model is a stochastic processing in ViBe. Each new observed pixel value has a 1/16 chance to overwrite a random position in the 20 previously stored pixel values. Previous pixel values are not stored as a FIFO queue since this implies some linearity to background pixel occurrence which is typically not the case in real world data. If a pixel history is updated there is another 1/16 chance to update one randomly selected neighboring pixel. This random update process allows for an adaptive model that can will slowly absorb foreground object that have become part of the static background.

ViBe employs the use of an open/close filter to remove noise from the foreground mask as in II-A. Van Droogenbroeck *e*t al. also suggest using the filtered mask as the update mask such that ViBe will add the unwanted noise to the background model.

D. Pixel-Based Adaptive Segmentation (PBAS)

PBAS, introduced by Hofmann *e*t al. [9], is a foreground segmentation algorithm that uses the stochastic portions of

ViBe [8] along with pixel-based adaptive thresholding and updating. PBAS adjusts thresholds to the pixel variance in the image by dynamically setting the threshold, τ, as shown in Equation 1, and the probability of pixel update from Section II-C.

Hofmann *e*t al. measure background dynamics by calculating the mean from a stored array of previously observed minimum pixel differences [9]. When background dynamics are high, a larger threshold, τ, can be used to reduce noise and the probability for updating the background model can be increased to allow for quicker absorption of false foreground detection. By contrast, when background dynamics are low, a smaller and more precise τ can be used with a smaller update probability to keep foreground detections in the foreground longer. This means PBAS allows for strong foreground segmentation on pixels with a highly static background while simultaneously using a more lenient set of parameters on highly dynamic regions of the image such as water or foliage.

E. Background Subtraction on Distributions

Work in a similar domain, the observation of avian behaviors, has been done by researching background subtraction techniques as a method for observing birds visiting a feeder [13], [14]. This environment naturally has an active background with foliage movement, however birds drawn to feeders are not typically in their ideal environment for camouflage and since they are feeding tend to be more active than when on the nest. The technique proposed in [13] was designed to solve noise generated by background movement by looking at pixel neighborhood distributions but is more computationally expensive than pixel-based approaches.

F. MotionMeerkat

MotionMeerkat is a general use tool to detect motion in ecological environments created by Ben Weinstein [15]. The tool is used to alleviate the process of video stream data analysis by extracting frames with motion from a video file. MotionMeerkat can either use MOG (Section II-B) or a version of Running Gaussian Average (Section II-A) for foreground segmentation and then uses blob detection and thresholding to determine if a foreground object it present. Weinstein's results show that MotionMeerkat is successful in many ecological environments but is still subject to problems such as rapid lighting changes, and camouflage.

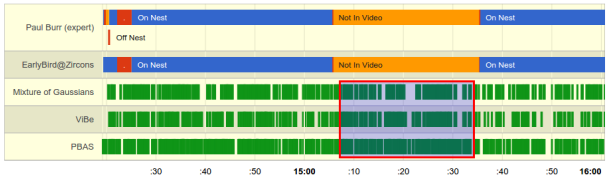
III. Methodology

The goal of this research is to determine which algorithms can best highlight regions of uncontrolled outdoor video with *interesting* events. This ideally can act as a filter and help scientists focus on segments of video that require their attention and letting them skip less interesting segments of video. The background subtraction methods need to be resistant to noise and handle quick correction of camera lighting problems while still being sensitive enough to detect the motion of a small to medium sized animal with cryptic coloration. The usefulness of these algorithms is sensitive to the number false positives and false negatives. Too many false positives and there many not be a significant length of video that can be classified as *uninteresting*, while too many false negatives may leave many *interesting* events unclassified and unwatched. An example of this is observed when comparing scientists' observations to positive events from the algorithms, as in Figure 4. An almost continuous stream of false positives can occur when vegetation moves in the wind when the grouse is not even at the nest (see Figure 4a), but on less windy days we see increased agreement between the two classifications (see Figure 4b).

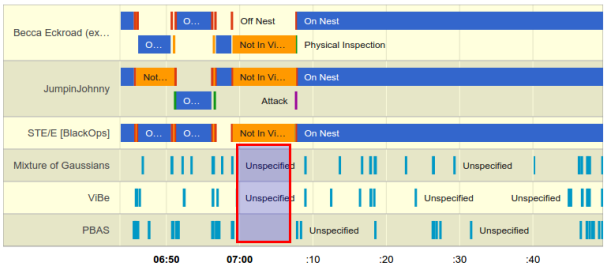
Three different algorithms were evaluated for their ability to accurately detect motion in Wildlife@Home's Interior Least Tern, Piping Plover and Sharp-Tailed Grouse video. MOG (see Section II-B) was chosen as the baseline for performance, as it is used as a standard for many new background subtraction implementations [5], [6], [9], [13], [17], [18] and has been successfully used in real world applications [19].

Modified versions of ViBe [8] and PBAS [9] were implemented and compared to MOG. ViBe is a good fit for this problem space as it is non-parametric and can be quickly initialized to prevent a large number of initial false positives. PBAS is an algorithm that adjusts its thresholding and update parameters on a pixel-by-pixel basis. PBAS is also good for this problem, where certain parts of the image are very noisy and at times entire sections of the video are polluted with dynamic lighting changes. PBAS will dynamically increase the foreground classification threshold during portions of a video affected by lighting changes and can learn to ignore regions of a video with large background variance such as in the grouse video (see Figure 3) where grass movement will span a large area of the video (100's of pixels) and pixel neighborhoods are not enough to detect the movement.

Modifications were made to improve performance on the noisy video and subjects with cryptic coloration. Initialization of ViBe and PBAS were adjusted to be second-frame-ready by adding the minimum number of values to the background model to match the first frame and filling the rest of the background model with values from random locations in the frame. This initialization allows for fast adaptation to subsequent frames if the background has a lot of motion while maintaining the minimum requirement to match the likely similar following frame. An open/close filter was also added to reduce foreground detection noise in the output mask. The mathematical morphology removes small unconnected bits of noise while maintaining the larger connected regions. This prevents many false detections due to video compression and camera induced noise. Depending on the video resolution filter size, this may be adjusted accordingly. Finally, in order to improve detection of birds, we use the convex hull of any connected foreground regions as the foreground mask. Since much of the birds are a similar color to their environment, generally only small areas are detected such as the head, tail feathers, and shadow, and much of the body can remain missing or segmented. The addition of a convex hull to connected foreground regions highlights bird movements and increases algorithm confidence. The convex hull may also be



(a) Timeline I



(b) Timeline II

Fig. 4. Timelines showing the number of false positives in a windy grouse video (4a) against those in a less windy grouse video (4b). The highlighted regions show time segments from the background subtraction results where there is no bird on the nest. These timelines were created using the Google Charts API [16] and are easily embedded in the Wildlife@Home user interface.

used in the future to detect extreme lighting changes since this will also emphasize large scene changes.

The conversion from the foreground mask to calculated events is done with time-series analysis. An event is defined as a specified video segment marked with a start and an end time. Foreground pixel counts are taken as a series of data points, and these are smoothed by using an exponential moving average. This further reduces detection noise and sporadic peaks. Once the data is smoothed its mean (μ) and standard deviation (σ) are calculated and used to determine which frames have more than 3σ foreground pixels using the inequality in Equation 5. If this is the case, it is marked as a significant event, otherwise it is ignored. Experimentation can be done to determine a good threshold for the standard deviation.

The equation for the exponential moving average is:

$$m_t = \alpha \cdot x_t + (1 - \alpha) \cdot m_{t-1} \tag{4}$$

where m_t is the mean at time unit t, x_t is the number of foreground pixels at time t, and α is the weighted decrease or learning rate. As $\alpha \rightarrow 1$ the new data is more heavily weighted. An example this time-series data compared to when scientist marked events occurred can be seen in Figure 5.

The calculation of significant foreground events is done using the following threshold inequality:

$$x_t > \mu + 3\sigma \tag{5}$$

This threshold is a good indication of foreground activity even when the video has a moderate amount of noise since noisy regions are either smoothed or taken into consideration in the time-series mean. The calculated foreground activity can then be compared to scientists to determine algorithm accuracy, as shown in Figure 4. By calculating events from regions with an abnormal amount of foreground pixels, a measure for the amount of foreground activity taking place is provided. This activity can then be compared to scientists to determine the accuracy of each background subtraction algorithm as shown in Figures 4 and 5.

An example of the correlation between background subtraction events and scientist observed events can be see in Figure 5.

TABLE I
ALGORITHM ACCURACY VS EXPERT SCIENTISTS ON TERN AND PLOVER NESTS

Event Type	Event Count	MOG	ViBe	PBAS
Preen	180	170	138	147
Scratch	4	4	2	2
Not In Video	732	632	578	607
Nest Exchange	22	16	16	16
Foraging	82	71	52	56
Adult-to-Adult Feed	20	6	6	6
Nest Defense	4	4	4	4
Predator	12	10	7	9
Non-Predator Animal	22	16	15	15
Unspecified	350	93	66	78
On Nest	932	665	582	608
Off Nest	2312	1960	1775	1876

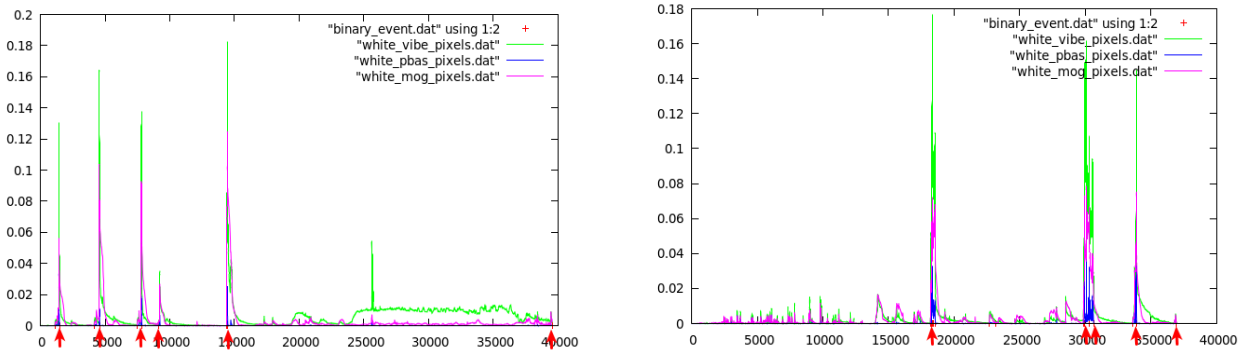
The arrows indicate human observed events in comparison with the time-series for each of the three algorithms. The data in these two examples are highly correlated with little noise and very few false detections. It can also be observed that PBAS is very quick to adapt to changes while ViBe has the largest detection emphasis among the three algorithms.

IV. RESULTS

The three background subtraction algorithms were run against a set of 105 tern and plover videos and 109 grouse videos. The plover and tern video totals 77.05 hours, and the grouse video totals 205.39 hours. Video lengths range anywhere from 30 minutes to 2 hours in length. Each algorithm runs at more than 10 frames per second (the recording frame rate) on a hyperthreaded 3.5 GHz core and is considered capable of real-time processing. Results were collected using a Mac Pro and 12 logical cores, which took approximately 48 hours. They were compared to observations made by project expert scientists and volunteer citizen scientists to determine algorithm accuracy.

A. Detecting Events with Background Subtraction

Tables I, III, II, and IV present how well each algorithm matched up to project scientists and volunteers for sharptailed grouse, and piping plover and least tern combined. Piping plover and least tern results were combined as the birds and



(a) Sample I (b) Sample II

Fig. 5. Example of event and foreground pixel count correlation. Red arrows indicate a scientist observed event and lines indicate foreground pixel count for each algorithm.

TABLE II
ALGORITHM ACCURACY VS CITIZEN SCIENTISTS ON TERN AND PLOVER NESTS

Event Type	Event Count	MOG	ViBe	PBAS
Not In Video	82	79	79	79
Nest Exchange	4	2	2	4
Adult-to-Adult Feed	14	14	14	14
Non-Predator Animal	16	16	14	14
Unspecified	10	10	10	10
On Nest	140	138	112	112
Off Nest	146	144	143	143

TABLE III
ALGORITHM ACCURACY VS EXPERT SCIENTISTS ON GROUSE NESTS

Event Type	Event Count	MOG	ViBe	PBAS
Not In Video	284	274	258	270
Eggshell Removal	6	4	5	5
In Video	130	128	129	129
Predator	6	5	5	5
Unspecified	2	2	2	2
Attack	2	2	2	2
Physical Inspection	60	52	56	56
Observation	44	41	39	41
On Nest	216	196	174	178
Off Nest	492	470	439	461

TABLE IV
ALGORITHM ACCURACY VS CITIZEN SCIENTISTS ON GROUSE NESTS

Event Type	Event Count	MOG	ViBe	PBAS
Not In Video	308	298	261	274
Nest Defense	2	2	2	2
Predator	14	12	10	12
Non-Predator Animal	2	2	1	2
Unspecified	2	0	2	2
Attack	22	18	20	21
Physical Inspection	46	46	45	46
Observation	8	7	7	8
On Nest	340	317	249	253
Off Nest	588	576	506	532

TABLE VI
ALGORITHM FALSE POSITIVES VS EXPERT SCIENTISTS

	MOG		ViBe		PBAS	
Species	μ	σ	μ	σ	μ	σ
Grouse	139.67	144.76	74.31	95.92	73.83	100.64
Tern	5.78	35.37	2.76	15.86	1.58	6.89
Plover	4	7.63	0.50	1.07	0.63	1.41

environments are highly similar, and both species are being observed for the same set of events. The *Event Count* column shows the total number of each event that occurred in the set of videos analyzed, and the following columns present how many of those events the background subtraction algorithm found.

Any background subtraction detected events that occur within 30 seconds of the start or end time of a scientist observed event are marked as a match. Multiple matches to the same start and end event from the same scientist are ignored. Since all three algorithms are adaptive, learning takes place in each algorithm where it will begin to ignore bird presence and absence on the nest. Event start and end times that take place within the first 10 seconds of the beginning of the videos were ignored as the algorithms did not have time to learn an initial background yet.

Table V compares results from combining all three background subtraction algorithms. The *Any Alg* column shows the number of events that matched any one of the three algorithms, and the *All Alg* column shows the number of events that matched all three algorithms. Using events marked by any algorithm provided a small increase in events detected over PBAS for all event types, however using a consensus showed a dramatic decrease in the number of events found. This decrease is indicative that the three different algorithms are not finding overly similar areas of activity within the videos.

B. Analysis of False Positives

Tables VI and VII provide an analysis of false positives generated by the background subtraction algorithms. False

TABLE V
ALGORITHM ACCURACY WITH CONSENSUS VS EXPERT SCIENTISTS ON TERN AND PLOVER NESTS

Event Type	Event Count	Any Alg	All Alg	MOG & ViBe	MOG & PBAS	ViBe & PBAS
Preen	180	174	137	138	143	137
Scratch	4	4	2	2	2	2
Not In Video	732	635	576	576	606	576
Nest Exchange	22	16	16	16	16	16
Foraging	82	73	51	52	54	51
Adult-to-Adult Feed	20	6	6	6	6	6
Human	2	0	0	0	0	0
Nest Defense	4	4	4	4	4	4
Predator	12	11	6	6	8	7
Non-Predator Animal	22	19	12	12	14	13
Unspecified	350	94	66	66	77	66
On Nest	932	669	572	580	606	572
Off Nest	2312	1974	1763	1769	1868	1763

TABLE VII
ALGORITHM FALSE POSITIVES VS CITIZEN SCIENTISTS

	MOG		ViBe		PBAS	
Species	μ	σ	μ	σ	μ	σ
Grouse	118.27	136.17	53.14	74.65	53.90	82.10
Tern	0.41	1.74	0.22	0.80	0.15	0.46

positives were counted by the number of computer classified events that occur during a user classified *Not In Video* event. Results are reported as the mean (μ) and standard deviation (σ) of false positives during any *Not in Video* event by any scientist over all videos tested for that species. Videos without a *Not In Video* event were ignored to prevent padding the results. A 10 second buffer is used after the start and before the end of the *Not In Video* events to avoid counting edge case movement as a false positive. This was used as a measure for false positives since at any other time a detection may correspond to an unmarked event, such as motion from the bird on the nest.

C. Effectiveness of Background Subtraction

The initial background subtraction results in Tables I, II, III, and IV show that background subtraction is accurate enough to be a reliable detection method for this type of video. Especially in the case of the *Not In Video*, *On Nest*, and *Off Nest* events, the detection accuracy is high enough to be useful for decision making. The other event sample sizes are still too small, requiring more results to be collected. MOG has the highest accuracy on both the tern and plover video however we also see the highest false postive rates from MOG across all species types. Due to MOG's high rate of false positives, PBAS is likely the best overall performing algorithm due to its low false positive rate and high accuracy. Utilizing results from any algorithm (Table V) shows a slight improvement over in performance over any individual algorithm. We also see than PBAS has a low number of false positives on the tern and plover observations (Tables VI and VII).

The Sharptailed Grouse have the highest average number of false positives (Tables VI and VII) and by far the highest standard deviation of false positives. The high variance in the grouse results suggest that some videos may have a low number of false positives, presumably indicating better precision on less windy videos. This also indicates that the high accuracy on the grouse videos (Tables III and IV) may not solely be false positives due to moving foliage.

Another major cause for algorithm inaccuracy and large variance in false positives (especially in the Least Tern samples) is from camera lighting autocorrection discussed in Section III and seen in Figure 6. Changes in scenery brightness from transitions in time of day or significant overhead cloud movement cause the camera to adjust brightness and can cause large scale false foreground detection. If the camera rapidly and repeatedly changes the brightness we see regions of video that the foreground algorithms cannot adapt to as shown in Figure 6. Due to the nature of PBAS, it adjusts to the rapid brightness changes but this still causes false negatives if a scientist observed event does occur during or shortly after the brightness adjustments.

Other detection errors are caused by video compression noise, and species cryptic coloration. The original archival Wildlife@Home videos taken by the field cameras are compressed by the hardware in part due to storage reasons. With these background subtraction algorithms working on moderate to heavy compression, false positives are caused during transitions between intra coded frames. More sensitive events such as preens and scratches can be difficult to detect due to the small amount of motion involved (typically just body rotation and head movement) given the camera distance, along with the cryptic coloration of the species. With the surrounding area taking on such a similar color to the bird a simple preen or scratch may easily go undetected by a background subtraction algorithm.

It is also worth noting that many detected events may not line up with the start or end time of a scientist observation but may still be a cause of bird motion. For example in Figure 4b, no events occur while the bird is off the nest but we see sporadic events while it is on the nest, this could be caused by bird adjustment on the nest or unmarked bird grooming events. The frequency of events occurring during a video may also

(a) Sample I (b) Sample II (c) Sample Issue Foreground Count

Fig. 6. Rapid and repeating brightness adjust caused by overhead cloud movement. Brightness is alternated multiple times per second creating a messy foreground pixel timeline show in Figure 6c. ViBe fails to adapt to the rapid changes and both MOG and PBAS become ignorant to small pixel changes required to detect bird movement.

serve as an additional indicator of bird presence, and merits further investigation.

V. Reproducibility

All the videos and human observations used to generate these results have been made available as part of the first Wildlife@Home data release[2]. The data is being made available not only to allow external researchers validate these results, but also because it is an extremely valuable resource for computer vision researchers interested in uncontrolled outdoor video. We hope this data release will encourage further study in detecting animals and events in this type of video. Further, all the Wildlife@Home source code for both the project's webpages and video analysis applications is made freely available on GitHub[3] for use by other researchers, both in computer science and wildlife ecology.

VI. Conclusion

This paper presents a preliminary analysis of the use of different background subtraction algorithms (Mixture of Gaussians, modified Pixel-Based Adaptive Segmentation and modified ViBe) for detecting events within uncontrolled outdoor avian nesting video. The effectiveness of these algorithms was obtained using over 250 hours of video along with human observations gathered by project experts and volunteer citizen scientists at the Wildlife@Home project [1], [2]. Initial results show that MOG has the highest accuracy but suffers from the largest number of false positives, while PBAS and ViBe have good accuracy while maintaining a low false postive rate. Both PBAS and ViBe reach high enough accuracy to be a promising technique for detecting video segments that are most interesting and important for an expert and citizen scientist to observe and classify. This opens up the possibility of using them as filters to dramatically reduce the amount of time spent by scientists analyzing the 85,000 hours of video at Wildlife@Home.

The videos used in this work were processed on a Mac Pro across 12 logical cores and proved adequate for retrieving this sample of results in just over 48 hours, however processing all 85,000 hours of video at Wildlif@Home is unfeasible. Due to these promising initial results, we are currently using the Berkeley Open Infrastructure for Network Computing (BOINC) [20] to harness Wildlife@Home's 2,000+ volunteered computers to collect background subtraction data for the entire data set.

In addition to analyzing more videos, improvements need to be made in order to accurately process detect segments of interest within the videos. Rapidly changing brightness inhibits the background subtraction algorithms. Possibilities for normalizing scene brightness, such as Retinex [21], [22] or adjusting the exponential moving average filter to mark video segments with extreme foreground detection (*e*.g., larger then 20% to 30% of the frame) remain as future work. More in-depth improvements could involve taking nest location into consideration and increasing the importance of foreground pixels located around the nest. Since cameras are placed strategically facing the nests we can safely assume nest location is close to the center of the frame and can easily scale foreground pixel importance accordingly.

Finally, these background subtraction results will be integrated into the interface used by project and citizen scientists to gain human feedback about how the correct algorithms are about marking event occurrences. This will not only help confirm the computed results but will also notify users to a possible upcoming event, which could improve human accuracy. This will provide a first step towards fully using automated strategies as a filter before showing the Wildlife@Home videos to scientists, allowing them to reliably skip segments of the videos where there is no animal activity.

Acknowledgments

We appreciate the support and dedication of the Wildlife@Home citizen scientists who have spent significant amounts of time watching video. This work has been partially supported by the National Science Foundation under Grant Number 1319700. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation. Funds to collect data in the field were

[2]http://volunteer.cs.und.edu/csg/wildlife/data_releases.php
[3]https://github.com/travisdesell/wildlife_at_home

provided by the U.S. Geological Survey and North Dakota Game and Fish.

REFERENCES

[1] T. Desell, R. Bergman, K. Goehner, R. Marsh, R. VanderClute, and S. Ellis-Felege, "Wildlife@Home: Combining crowd sourcing and volunteer computing to analyze avian nesting video," in *eScience (eScience), 2013 IEEE 9th International Conference on*. IEEE, 2013, pp. 107–115.

[2] T. Desell, K. Goehner, A. Andes, R. Eckroad, and S. Ellis-Felege, "On the effectiveness of crowd sourcing avian nesting video analysis at Wildlife@Home," in *The 15th International Conference on Computational Science*, Reykjavk, Iceland, June 2015.

[3] W. A. Cox, M. S. Pruett, T. J. Benson, J. C. Scott, and F. R. Thompson, "Development of camera technology for monitoring nests," *Video surveillance of nesting birds. Studies in Avian Biology*, vol. 43, pp. 185–210, 2012.

[4] S. N. Ellis-Felege and J. P. Carroll, "Gamebirds and nest cameras: present and future," *Video surveillance of nesting birds. Studies in Avian Biology*, vol. 43, pp. 35–44, 2012.

[5] A. M. McIvor, "Background subtraction techniques," *Proc. of Image and Vision Computing*, vol. 4, pp. 3099–3104, 2000.

[6] M. Piccardi, "Background subtraction techniques: a review," in *Systems, man and cybernetics, 2004 IEEE international conference on*, vol. 4. IEEE, 2004, pp. 3099–3104.

[7] P. W. Power and J. A. Schoonees, "Understanding background mixture models for foreground segmentation," in *Proceedings image and vision computing New Zealand*, vol. 2002, 2002, pp. 10–11.

[8] M. Van Droogenbroeck and O. Barnich, "Vibe: A disruptive method for background subtraction," *Background Modeling and Foreground Detection for Video Surveillance*, 2014.

[9] M. Hofmann, P. Tiefenbacher, and G. Rigoll, "Background segmentation with feedback: The pixel-based adaptive segmenter," in *Computer Vision and Pattern Recognition Workshops (CVPRW), 2012 IEEE Computer Society Conference on*. IEEE, 2012, pp. 38–43.

[10] D. Koller, J. Weber, T. Huang, J. Malik, G. Ogasawara, B. Rao, and S. Russell, "Towards robust automatic traffic scene analysis in real-time," in *Pattern Recognition, 1994. Vol. 1-Conference A: Computer Vision & Image Processing., Proceedings of the 12th IAPR International Conference on*, vol. 1. IEEE, 1994, pp. 126–131.

[11] J. Heikkilä and O. Silvén, "A real-time system for monitoring of cyclists and pedestrians," *Image and Vision Computing*, vol. 22, no. 7, pp. 563–570, 2004.

[12] G. Bradski, "Opencv," *Dr. Dobb's Journal of Software Tools*, 2000.

[13] T. Ko, S. Soatto, and D. Estrin, "Background subtraction on distributions," in *Computer Vision–ECCV 2008*. Springer, 2008, pp. 276–289.

[14] ——, "Warping background subtraction," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. IEEE, 2010, pp. 1331–1338.

[15] B. G. Weinstein, "Motionmeerkat: integrating motion video detection and ecological monitoring," *Methods in Ecology and Evolution*, 2014.

[16] Google Inc. (2015) Google charts. [Online]. Available: http://developers.google.com/chart/

[17] A. Elgammal, D. Harwood, and L. Davis, "Non-parametric model for background subtraction," in *Computer VisionECCV 2000*. Springer, 2000, pp. 751–767.

[18] K. Kim, T. H. Chalidabhongse, D. Harwood, and L. Davis, "Real-time foreground–background segmentation using codebook model," *Real-time imaging*, vol. 11, no. 3, pp. 172–185, 2005.

[19] S. C. Sen-Ching and C. Kamath, "Robust techniques for background subtraction in urban traffic video," in *Electronic Imaging 2004*. International Society for Optics and Photonics, 2004, pp. 881–892.

[20] D. P. Anderson, "Boinc: A system for public-resource computing and storage," in *Grid Computing, 2004. Proceedings. Fifth IEEE/ACM International Workshop on*. IEEE, 2004, pp. 4–10.

[21] E. H. Land and J. McCann, "Lightness and retinex theory," *JOSA*, vol. 61, no. 1, pp. 1–11, 1971.

[22] D. J. Jobson, Z.-U. Rahman, and G. A. Woodell, "A multiscale retinex for bridging the gap between color images and the human observation of scenes," *Image Processing, IEEE Transactions on*, vol. 6, no. 7, pp. 965–976, 1997.