

An Adaptive Real-Time Statistical Algorithm for the Estimation of the Local Steelhead Population in Freshwater Creek

by

Ken Owens and Mike Haley

**Humboldt State University
Dept. of Mathematics
Arcata, CA. 95521**

kdo10@humboldt.edu
jmh17@humboldt.edu

March 15, 2003

Abstract

To count steelhead trout appearing on underwater video, we have developed a real-time adaptive motion detection algorithm. This algorithm begins by viewing video void of fish images and producing an estimate of this non-fish state. As successive images are viewed at 30 frames per second, decisions are made as to whether or not motion has been detected. If not, images are used to update the estimate of the non-fish state, otherwise they are saved for later analysis. In this paper, we report details of this algorithm and results from applying it to still images and videotape of migrating steelhead trout.

Introduction

Estimating salmonoid abundance in streams, in order to gain an estimate of the overall population, is fundamental in managing effective and judicious fisheries. Current methods of population estimation include electrofishing, tagging, and visual methods requiring divers. The California Department of Fish and Game's Northern California North-Coast Steelhead Research and Monitoring Program's management is interested in estimating the size of the steelhead population in Freshwater Creek for use in management decisions. In the winter of 2001, a sample of steelhead swimming upstream to spawn was captured and tagged. As the fish returned to the sea a second sample was taken using an existing weir to force the fish into a trap. The ratio of tagged to untagged fish was then obtained which allowed managers to estimate the population of the steelhead. During the second capture the water flow on Freshwater Creek became low and clear. When the DFG realized that a large number of steelhead were remaining upstream to avoid the obvious trap, the agency opted to remove the trap and eschew further data collection.

In order that this situation might not happen again, DFG replaced the trap with a ten-inch diameter tube that is several feet long. Mounted to the side of the pipe is a lighting system and a watertight camera box that captures images of fish as they swim through the tube. On the upstream mouth of the tube an antenna collects data from previously tagged fish. The camera records six frames per second which allows 36 hours of monochrome images to be captured on a two-hour and forty-minute tape.

The difficulty that now arises is the amount of data that is collected. The tagged fish that swim through the pipe can be found on the tape, since they register a time signal as they pass near the antenna, but the untagged fish need to also be found and counted. Preliminary videotapes that have been analyzed typically returned ten seconds of fish footage per four hours of videotape. Under these circumstances, a technician must watch 60 hours of videotape for approximately 2.5 minutes of fish crossing footage. DFG biologists acquired 50 VHS tapes during the winter and spring of 2002, and are currently collecting new data. The predicament lies not in the difficulty of the task, rather the sizeable amount of data and the tedious nature of viewing it. The task of counting the fish on tape is a job where the use of a computer capable of deploying an adaptive statistical technique would prove invaluable.

Previous efforts, including work on fire detection in aircraft engine compartments and dry bays [1] and [2], have developed machine vision systems capable of real-time fire detection. This work lends validity to the possibility of developing a real-time fish detection system capable of characterizing no-motion images and separating them from motion images. For a general discussion of machine vision techniques see [3] and [4].

Methods

I) Algorithm

The basic task of the algorithm is to detect motion. However, our approach is to develop an algorithm to detect the lack of motion and label everything else as motion. The data upon which we will base these decisions is a real-time stream of successive difference images (Fig. 0).

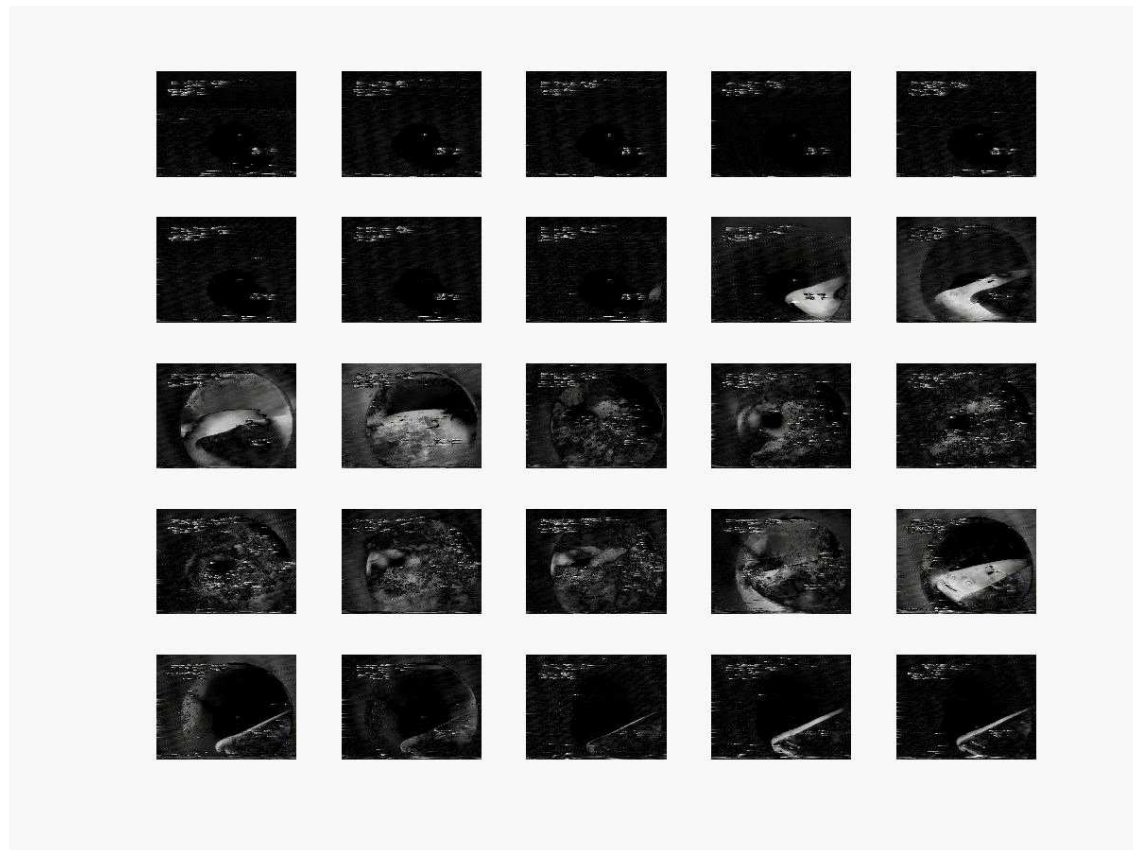


Figure 0. Sample difference images taken from 26 consecutive frames.

Instead of working with the entire image, we choose to work with the mean absolute value of the pixel differences between adjacent images (MPV), which characterizes the change between images and can be used to detect motion [1]. Image differencing is also

computationally inexpensive making it a good candidate for use in real-time data processing.

This algorithm results in a functioning program that is capable of tracking the mean pixel values of difference images (See Appendix A to view the source code). The purpose of using a moving average vector is to smooth the data out, while the process-estimate-vector acts as an adaptive filter which updates image statistics for slow system fluctuations while classifying and cataloging images with greater fluctuations as fish occurrences. The adaptability of this program is crucial for several reasons: the system naturally exhibits fluctuations which are a function of ambient lighting, water turbidity, disturbances in the water, image capture/playback noise as well as fluctuations due to technicians maintaining the system resulting in camera movement and lighting changes. The algorithm estimates the lack of motion by comparing a no-motion process estimate with the current moving average of the mean pixel values. The no-motion process estimate is the average component of the process-history-vector. This process-history-vector is computed from the moving averages of the mean pixel values and is updated when the current moving average is within a given decision threshold of the no-motion process estimate (Fig. 1).

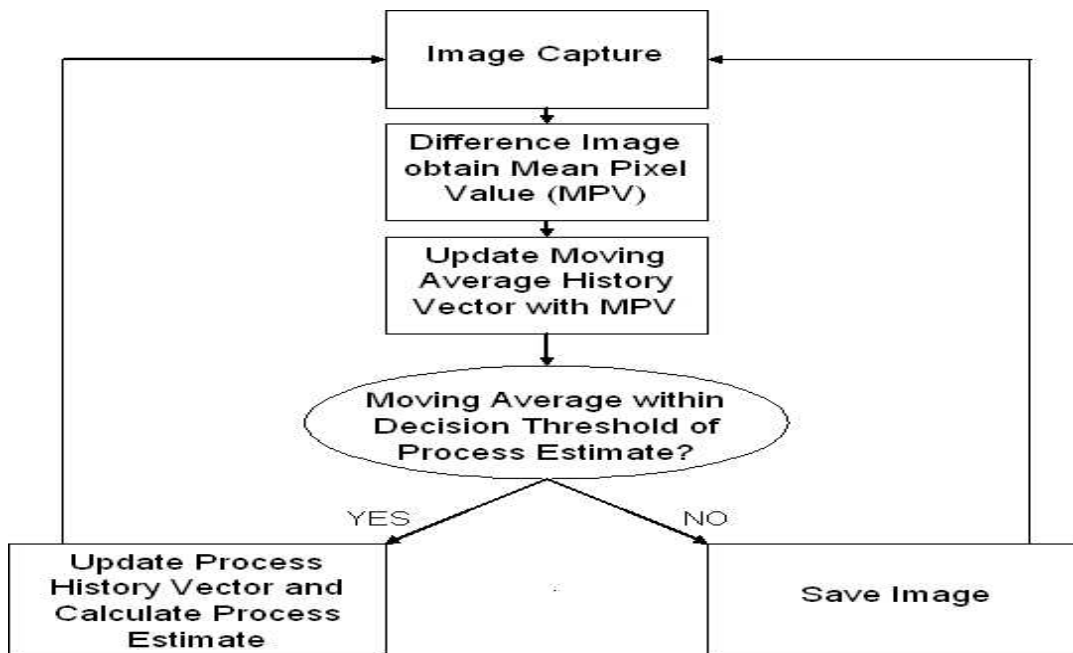


Figure 1. Flow chart of the fish detection algorithm that obtains a process estimate from images classified as non-motion images.

Mathematically the no-motion process estimate is equivalent to the least squares estimate over the process history window. To see this, we model the measured mean pixel values as $mPV_i = mPA + \eta_i$ where mPA is the theoretical mean pixel value of the difference

images and η_i are independently identically distributed mean zero Gaussians. Then we can write the squared error over the process history window as

$J = \sum_{j=0}^{k-1} \sum_{i=1+j}^{k+j} (mPV_i - p_{2k})^2$. In this equation p_{2k} is the process estimate at time $2k$, the

inner sum is the error of the moving average estimates and the outer sum gives the total error over the process history window. Then setting $\frac{dJ}{dp_{2k}} = 0$ implies that the least

squares estimate of the no-motion process at time $2k$ is $\hat{p}_{2k} = \frac{1}{k} \sum_{i=1}^k mA_i$, where mA_i is the

i th moving average in the process history window. Thus, the least squares estimate of the non-motion non-fish process is the average of the moving averages in the process history window. Further, this estimator is an unbiased estimate of the no-motion process.

II) Data

All of the preliminary data analysis was done using Matlab and the Image Processing Toolbox [5]. At this stage in the process a functional real-time frame grabber was not operational and so images were obtained using a Super VHS machine that was able to pause at each frame in order to manually extract specific images. Approximately 400 images from two different videotapes taken from data acquired on February 8, and March 8, 2002. This data is characterized by three segments containing fish crossings surrounded by no motion images.

Initial analysis found that each difference image could be represented in feature space by the mean pixel value and the standard deviation of the mean pixel value [6]. When the data was evaluated, several important observations were obtained: no-motion images were contained in a relatively small area in feature space with a small deviation from one image to another and the fish images were found outside of this region with much larger deviations between the images. This meant that images could be classified as either fish or non-fish based upon their proximity in feature space to previous images. Another important observation was that there was a near linear relationship between the mean pixel value and the standard deviation of the mean pixel value, suggesting that these measurements are highly correlated and that the basic fish/no-fish classification could be done with the mean pixel value only [6].

III) C Processing

Following our computational strategy of, “develop code in Matlab and port to C for processing speed”, we ported the above Matlab decision algorithm to C running under Linux [6], [7]. However, to use it to process video, we needed to get images from videotape into computer memory. This issue had been solved by the open source Video for Linux (V4L) community. As with all open source code, it was developed by many people and made freely available to anyone who wants to use it. We purchased an

inexpensive (\$40.00 US) video card based on the Brooktree Bt84xx video processing chip and wrote our own custom real-time frame grabber based on code and support from V4L web pages. After setting the task priority as low as possible, the real-time (30 frames per second) video grabbing, image processing and decision algorithm use approximately 30% of the computational cycles of a Pentium4 processor running Linux 7.3.

Results

I) Matlab Results

Figure 2 contains the results of applying this algorithm to three separate fish occurrences. In these three situations the decision threshold has been set at 3.5. The results obtained using this data in Matlab will be used to set the parameters in the C code and provide Matlab estimates of type I and II detection errors.

In Figure 2a, the Fish 1 data set is comprised of 45 images where the steelhead passes by the camera in only 3 frames, frames 20-22. Thus the difference images should have large pixel deviations from frame 19 to 23. Figure 2a shows a plot of the mean pixel value for each difference image. This measurement exhibits small fluctuations which are inherent in the system. At image 19 the mean pixel value of the difference image is considerably greater for several images as the fish passes through the system, before returning to the mean pixel values observed before the fish occurrence.

A moving average of the mean pixel of the difference image is acquired which smoothes the fluctuations in the system. The moving average can only be computed beginning on the tenth frame because it is acquired by taking the average value of the nine previous mean pixel values as well as the current mean pixel value. Thus the current moving average is dependent upon ten mean pixel values: the current mean pixel value and the nine previous pixel values. Because the moving average is acquired in this manner it appears to be shifted in relation to the unmodified mean pixel values. This can be clearly seen at approximately the 25th frame where the mean pixel value of the images is back to a level seen before the fish occurrence and yet the moving average is considerably greater than this value because it represents the previous images as well.

An estimate of the no-motion process is then obtained by using the previous ten moving average values, which is why the process estimate does not begin until the 19th frame. Thus the process estimate displayed at frame 19 is dependent upon the ten moving average values obtained at frames 10, 11, ..., 19. This process estimate will be used to decide if the moving average at frame 20 is a no-motion image or a fish occurrence image. This decision is based upon taking the absolute value of the difference of the moving average and the process estimate, if this value is within the predefined decision threshold then the no-motion process estimate will be updated, otherwise the image is recorded for further analysis. The decision threshold is represented on the Figure 2 by the region which surrounds the process estimate. As can be seen at frame 20 of Figure 2a, the moving average is within the decision threshold of the process estimate and is thus

used to update the process estimate. However, in frame 21 the absolute value of the difference between the moving average and the process estimate is greater than the threshold, thus the process estimate is not updated with this value. Instead the image is classified as a fish occurrence. This is shown with the motion detection indicator graph in Figure 2a, where zero represents no-motion process and five represents motion. This graph demonstrates that the process estimate is not updated, remaining constant, until the moving average of the system is within the decision threshold, which occurs at frame 31. Therefore, this fish occurrence during frames 20-22 results in classification of fish occurrences at images 21-30.

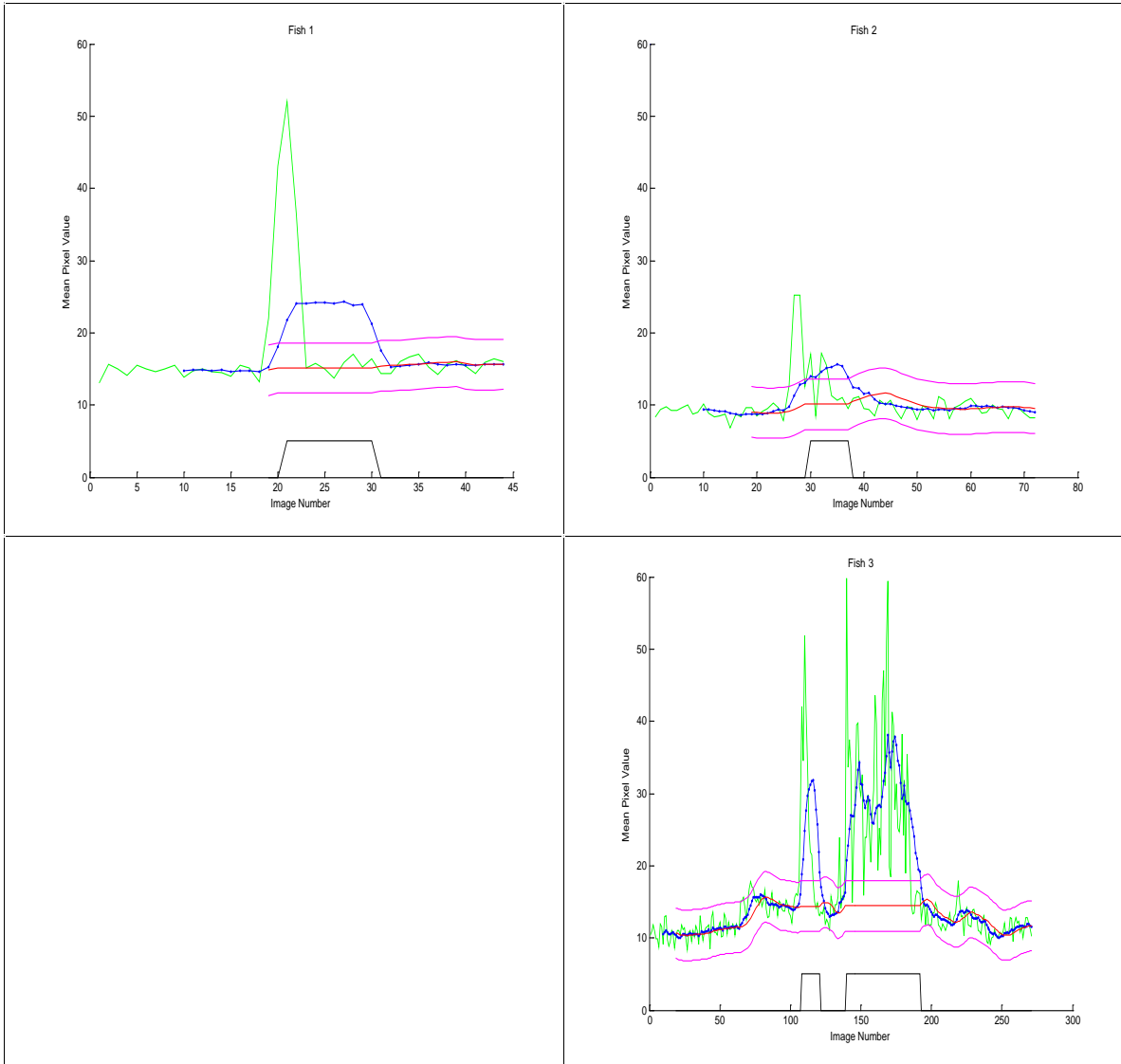


Figure 2 a,b,c shown clockwise from the upper left: Plots of the three fish demonstrating the relationship between the mean pixel value data and the computed moving average and process estimate.

In Figure 2b, the Fish 2 data is comprised of approximately 70 images with 7 fish images. The initial mean pixel value obtained from the difference image is about 10, and by the

10th frame the moving average is plotted. The process estimate begins at the 19th frame within the inclusion zone. Before the 30th frame the fish has entered the image and the mean pixel value signal rises to 25, drops back to 8, and then rises again to 15. This movement results in a moving average which is relatively small, which means that detection is dependent upon the threshold value. If the threshold was set higher, these fish images would not be detected. While the moving average is greater than that allowed by the threshold the process estimate is not updated. Once the moving average is within the inclusion region no-motion process estimation begins again.

The Fish 2 data illustrates an important caveat which needs to be discussed. It is clear that once the moving average is within the inclusion region the process estimate is again updated, but the process estimate is delayed. As discussed earlier, this is because the current moving average is dependent upon the ten most recent mean pixel values while the process estimate is dependent upon the previous ten moving averages. Thus, the process estimate is based upon the previous 19 mean pixel values, where the mean pixel from ten images back is the most dominant. The danger is that the moving average is not forced outside of the inclusion region until several fish images have already past, and remains outside after the fish has left the scene.

The Fish 3 data in Figure 2c comprises approximately 270 difference images. In this series of images the fish enters after the 100th image and exits near the 120th image. The fish then re-enters for approximately 50 images. On the VHS tape it can be seen that the steelhead passes completely through the tube tail first and then reappears, this time keeping its body in the viewing range. The mean pixel value measurements of each difference image are used to calculate the moving average, which closely follows the data. The moving average does not begin until the 10th frame because it is dependent upon the current mean pixel value as well as the previous nine values. The process estimate begins at the 20th frame and adapts to the change which is occurring in the original data. When the fish enters the image after the 100th frame the process estimate is not updated since the moving average is greater than that allowed by the decision threshold. The process estimate maintains its previous estimate without updating itself until the moving average data falls within the decision threshold again. This is again seen near frame 140, the process estimate is not updated until frame 190 since the moving average is too great. Once the fish has exited the scene the mean pixel value of the differences images returns to the same values that were occurring before the fish occurrence. The fact that the no-motion process estimate is able to follow the system even after a fish occurrence lends credibility to the rigor of the algorithm. Further, comparisons between the three fish occurrence graphs yield some insight into the robustness of our algorithm. The process estimate of each of the three figures range in mean pixel values of 9 to 15, thus demonstrating that our algorithm is able to adapt to varying ambient conditions.

Another important aspect of data is the signal strength that is detected when the fish enters the image. Figure 3 shows some typical fish images which correspond to the data in Figure 2. Fish 1 obtains a mean pixel value greater than 50 at one point in its crossing.

Since this measurement is so large, the moving average is large as well. Thus, the moving average is considerably greater than the decision threshold region around the process estimate. A similar phenomenon is produced by Fish 3, where the maximum mean pixel value of the fish passing through the tube is greater than 50. This results in a moving average which is significantly greater than the decision threshold around the process estimate. Contrast these occurrences with the Fish 2 data where the maximum mean pixel value for fish crossing is only 25 due to turbidity. This results in a moving average which is not much greater than the process estimate. In fact, this is an example of how the sensitivity of the system depends upon the decision threshold. Fish 3 conversely is captured on tape with a large amount of contrast, resulting in a signal which is much greater than the surrounding no-motion images (Fig. 3). It therefore becomes imperative to optimally choose a decision threshold.

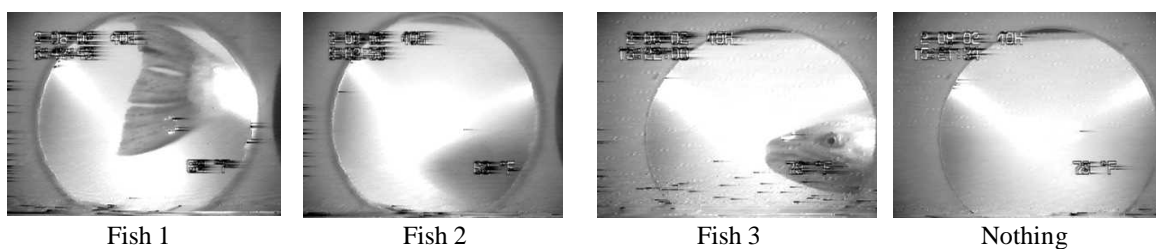


Figure 3. Four images taken from the images used in the Matlab analysis.

The decision threshold is an important parameter of the system which can have a great affect on the number of images which are recorded. The ideal decision threshold is large enough to update the process estimate with all of the no-motion images and yet be small enough to save all fish images. Thus, after classification there are four image types: fish images that are correctly classified as fish images, fish images that are incorrectly classified as no-motion images, no-motion images that are incorrectly classified as fish images, and no-motion images that are correctly classified as no-motion images. A decision threshold should be chosen which has a high probability of correctly detecting fish as well as a low probability of false alarm. The Detection and Error Probabilities Graph, shown in Figure 4, displays probabilities for the four different ways that an image can be classified. If type I and type II errors are equally weighted then the optimal decision threshold value is approximately 3.5. This threshold maximizes the fish detection probability while minimizing the false alarm probability (Fig. 4)

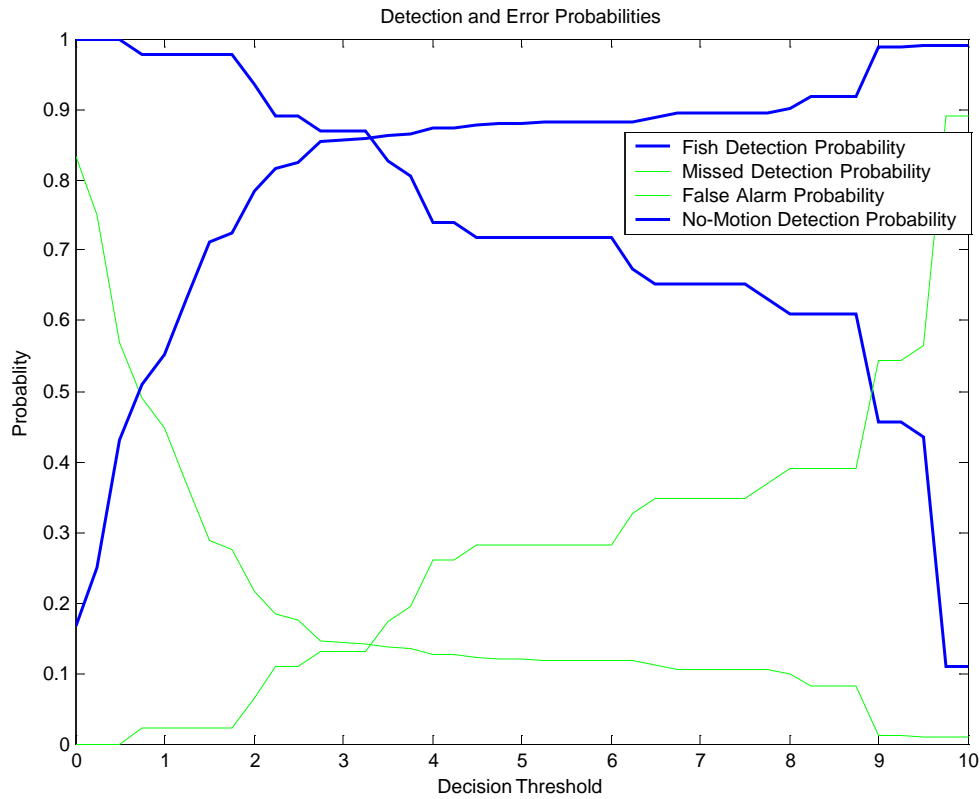


Figure 4. The resulting detection and error rates obtained from using various decision threshold values.

II) Tape Results

To test our real-time algorithm on actual videotape, we manually watched 23 minutes and 57 seconds of videotape which was recorded at 6 frames per second for a total of 8,622 images watched. The two fish shown in Figure 5 appeared in this video segment, 17 images for the first fish and 9 for the second, for a total of 26 fish images. We ran the algorithm with a decision threshold of 4, and it detected 4 of the first 17 fish images and 3 of the second 9 fish images, giving a detection probability of 7/26 or 27%. This number is much lower than the 70% theoretical value from Figure 4. The reason for this is the effect the lag has on the small sample of 26 images. This lag caused our algorithm to both miss the early fish images and to continue recording after the fish had exited the scene. Not only does this lower the detection rate but it also increases the false alarm rate. Of the 8,596 non-fish images on the tape, 22 were misclassified as fish, due in large part to the lag mentioned above, giving a false alarm rate of 0.25%. This number is much lower than the Matlab value of 15%. We believe that Matlab false alarm rates are in error for the same reason the detection probability is in error; namely, a small sample size of only several hundred images (compared with over 8,000) and lag induced effects.

These numbers can also be viewed in terms of the ability of our algorithm to detect no-motion images. The algorithm viewed 8,596 images without fish. Of these, 8,574 were correctly labeled as no-motion images giving a 99.7% detection rate. Of the 22 fish

images, 19 were misclassified as no-motion images, giving a no-motion false alarm rate of $19/26 = 73\%$. Obviously we have conservatively constructed our algorithm to favor the misclassification of no-motion images as fish rather than miss detecting actual fish.

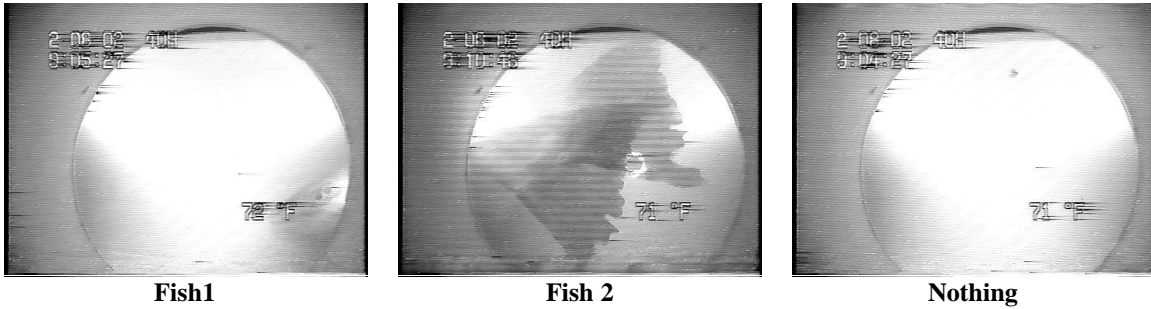


Figure 5. Sample Images Captured in Real-Time

What is important is not these per/fish detection rates but the fish crossing detection rate. Only one image per fish crossing need be recorded for DFG to produce a population estimate. To estimate the crossing detection rate we compute the probability of not missing all of the fish images during a given crossing. These probabilities can be estimated using a binomial probability with a detection probability of 27% (as estimated above) and a non-detection probability of 73%

Table 1.

Fish Images per Crossing	Probability of detecting at least one fish per crossing	Fish Crossing Detection Rate
1	$1-(0.73)^1$	0.27
2	$1-(0.73)^2$	0.47
3	$1-(0.73)^3$	0.61
4	$1-(0.73)^4$	0.72
5	$1-(0.73)^5$	0.79
6	$1-(0.73)^6$	0.85
7	$1-(0.73)^7$	0.89
8	$1-(0.73)^8$	0.92
9	$1-(0.73)^9$	0.94
10	$1-(0.73)^{10}$	0.96

Our observations of the data have show that fish can pass through the video housing tube in as little as 3 frames, however other fish have spent as many as 50 frames in front of the camera. A typical fish crossing lasts 8 frames which would imply the crossing detection rate is greater than 90% (Table 1).

Conclusions

This paper introduces an adaptive real-time motion detection algorithm to be used to help DFG biologists obtain an estimate of the population of steelhead trout in Freshwater Creek. This algorithm has been applied to still images and videotape of migrating steelhead trout and estimates have been given for the detection and false alarm probabilities. The false alarm probability is quite low, but the current estimate of the per image detection rate is only 27%. However, this estimate was computed from only 26 images and larger test data sets are required to improve its accuracy. What is most important is the fish crossing detection probability, which we have estimated to be greater than 90%. A more accurate estimate will require computation of the fish-images-per-crossing frequency and larger test data sets. Further, the analysis of both Matlab simulations and videotape indicate that the percentage of recorded fish per crossing can be increased to 100% for detected fish if an image buffer is kept of the most recent images. From our experiments it is also clear that we must study the effect of changing the process-estimate-history window on the responsiveness of our algorithm.

The work presented here is only the first step towards solving this videotape image classification problem. We are in the process of generalizing the adaptive mean pixel estimator to an adaptive image histogram estimator. The authors believe that illumination independent/histogram normalized class representations of other objects seen on the tapes such as air bubbles can be computed and used to classify such objects in real time.

Acknowledgements

We would like to thank the North-Coast Steelhead Monitoring Program in Arcata California for providing the steelhead trout videotape and the Video for Linux community for sharing their Linux video programming knowledge.

References

- [1] S. Y. Foo, "A machine vision approach to detect and categorize fires in aircraft dry bays and engine compartments," in *Conf. Rec. IEEE -IAS 30th Annu. Meeting*, Lake Buena Vista, FL, Oct. 1995, pp. 1556-1564.
- [2] "Machine vision fire detector for millisecond response," *Photonics Spectra*, pp. 18-20, Oct. 1993.
- [3] R. Duda, P. Hart, *Pattern Classification and Scene Analysis*. New York: Wiley-Interscience, 1973.
- [4] A. Webb, *Statistical Pattern Recognition*. New York: Oxford University Press, 1999.
- [5] A. Knight, *Basics of Matlab and Beyond*. New York: Chapman & Hall/CRC, 2000.
- [6] M. Haley, "Automated steelhead identification using statistical pattern recognition techniques", M.S. Thesis, Humboldt State University, Arcata, Ca, 2003.
- [7] G. Glass, K. Ables, *Unix for Programmers and Users*. Upper Saddle River, NJ: Prentice-Hall, 1999.
- [8] B. Kernighan, D. Ritchie, *The C Programming Language*, 2nd Ed., Englewood Cliffs, NJ: Prentice-Hall, 1988.