F. Westling, C. Sun, D. Wang, A Modular Learning Approach for Fish Counting and Measurement Using Stereo Baited Remote Underwater Video, In Proceedings of Digital Image Computing: Techniques and Applications, Wollongong, Australia, 25-27 November 2014, pp.263-269.

A modular learning approach for fish counting and measurement using stereo baited remote underwater video

Fredrik Westling*
School of Computer Science and Engineering
University of New South Wales
fredrik.westling@gmail.com

Changming Sun
Computational Informatics
CSIRO
changming.sun@csiro.au

Dadong Wang
Computational Informatics
CSIRO
dadong.wang@csiro.au

***Abstract*—An approach is suggested for automating fish identification and measurement using stereo Baited Remote Underwater Video footage. Simple methods for identifying fish are not sufficient for measurement, since the snout and tail points must be found, and the stereo data should be incorporated to find a true measurement. We present a modular framework that ties together various approaches in order to develop a generalised system for automated fish detection. A method is also suggested for using machine learning to improve identification. Experimental results indicate the suitability of our approach.**

I. Introduction

When studying fish populations, either for marine research or industrial fishery purposes, it is crucial to obtain accurate information on size and shape of fish populations [1]. Traditionally, this has been carried out using methods including extracting fish from the ocean by casting nets, and human underwater observation [2]. This poses several issues since these methods are intrusive upon the ecosystem: casting nets kills fish and interferes with unrelated wildlife, and human observation disturbs the marine life. To this end, various systems using underwater cameras have been suggested and implemented in recent years, including using a method called Baited Remote Underwater Video Systems (BRUVS) [3], [4]. However, current systems require manual analysis by trained experts which requires considerable time and effort. Spampinato et al. [2] suggest it could take as much as 15 minutes for a marine biologist to work through a minute of footage, classifying and annotating. Automating this process is clearly of critical importance to the success of these systems.

II. Background

Many different approaches have been taken to automating fish identification in stereo video, although many authors use different circumstances to develop their work on. Several authors, for instance, restrict their studies to constrained environments, e.g. fish tanks [5]. Han et al. used a simple background subtraction method to extract and measure fish from simple low-quality stereo images [6]. Some researchers have developed sophisticated methods based on simple, staged photographs of single fish [7]. These methods are not immediately practical for natural environments like those seen in BRUVS footage, which has many different (potentially unseen) fish that move and cluster. However, they offer some useful techniques.

Other works have focused on developing successful methods which compensate for changing fish shapes caused by fish deformation while swimming, using point distribution models [8] or deformable template matching [9]. These results are more practical, since fish in natural environments can be seen from any angle and deform in many ways.

More recent research has also explored machine learning techniques such as Support Vector Machines and decision trees [10]. The basic problem of incorporating machine learning techniques for this task is deciding upon features to use that can be learned. Larsen et al. were able to find a set of features that provided a 76% resubstitution rate using linear discriminant analysis [11]. Shortis et al. [12] suggest using a Nearest Neighbour Classifier to validate the candidate regions as ‘fish’ or ‘not fish of interest’.

In this paper, we propose a new method for structuring such systems to incorporate improvable ‘modules’ and a supervised learning approach.

III. Method

A. Modules

The problem of counting and measuring fish can be broken into three major steps: Identification, Tracking and Measuring. We can define an implementation of this framework by defining a particular set of interfaces. Once these interfaces are defined, the implementations of each step can be changed, switched out and redefined without affecting the others. This breakdown was based on Shortis et al.’s proposed approach to fish recognition [12], although we vary some of their suggestions.

Identification of fish in any given frame is the natural first step of the process, since nothing can be done unless fish are identified. We took this identification as acting within a single frame, figuring out which objects in the image are of interest. This step, by itself, is inadequate to analyse BRUVS footage, as individual frames fail to capture all required information.

Tracking of fish across N frames is necessary in order to obtain an accurate count, as it prevents the system from double counting individuals. It can also be used to refine the

*Most of this work was carried out when the first author was at CSIRO.

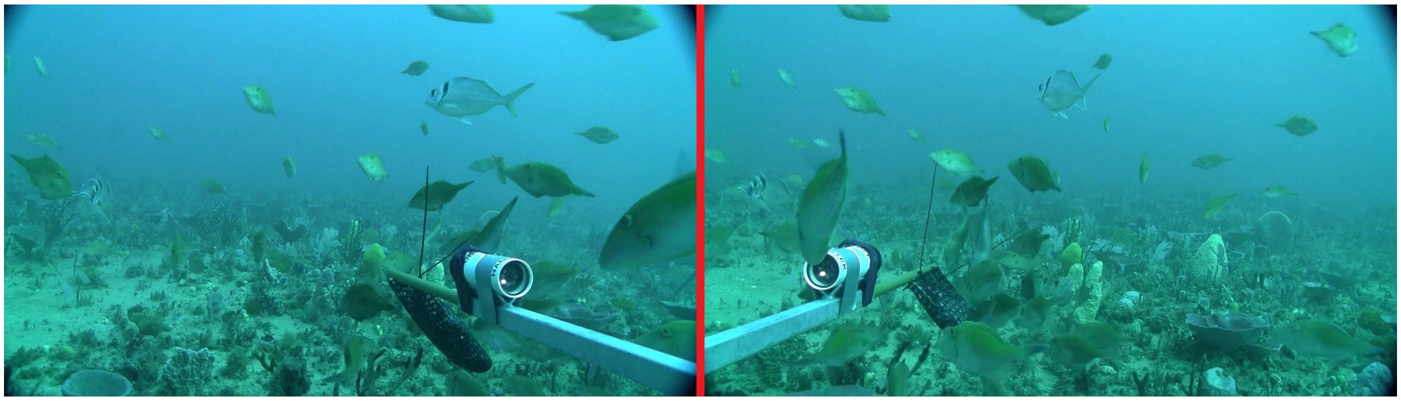

Fig. 1. Raw frames of a BRUVS stereo pair.

identifications made on individual frames. For small *N*, simple characteristics can be taken into account to invalidate noise, under the assumption that fish are consistently visible. By taking this into consideration, it also becomes possible to find fish that are no longer visible (for instance, hidden behind another). As *N* grows, techniques like trajectory analysis can be used to validate identification further [13].

Measuring is applied to matching stereo frames to find the size of fish that have been identified and tracked. Typically, snout to tail (or fork) measurements are used to define the length of a fish [12], although alternative systems have to be employed when these two points are not available [14]. Stereo video techniques need to be applied to obtain an accurate measurement between the two points. We use sparse disparity techniques as they lent themselves well to our implementation.

These three modules are the main parts of our suggested framework, but we also added two steps which are only required due to insufficient data being available. Having access to the image background allows the use of many identification and tracking methods made popular in pedestrian tracking research [15] which depend on background subtraction. When a background image is not available, background extraction can be used to generate a background using a sequence of video. This background can then be improved as dynamic parts of the image are identified and tracked. The other added step is the generation of a training set of fish images, required for supervised learning methods. We present a suggested approach for generating this set during the operation of the system, in a semi-supervised manner. These two modules are not required if the associated data are available.

B. Implementation

A crucial aspect of implementation, in order to make the system modular, is defining a common interface for the three steps. As long as each individual module uses a well-defined interface, the processing techniques implemented can be independently changed. The system was implemented in C++ using the Open Source Computer Vision Library (OpenCV) [16], using BRUVS stereo video footage of a naturally noisy underwater environment. An example stereo pair can be seen in Figure 1.

1) Background extraction: We wanted to use background subtraction techniques to identify dynamic foreground objects in the footage, but did not have access to a single frame with no fish in view. In order to get around this, we extracted the background from the video footage we had. In order to generate a sharp, clear background image (unlike those produced by frame-averaging techniques), we set each pixel to its most likely value for each colour channel by generating a histogram for the pixel channels across a uniformly distributed subset of frames. The results can be seen in Figure 2.

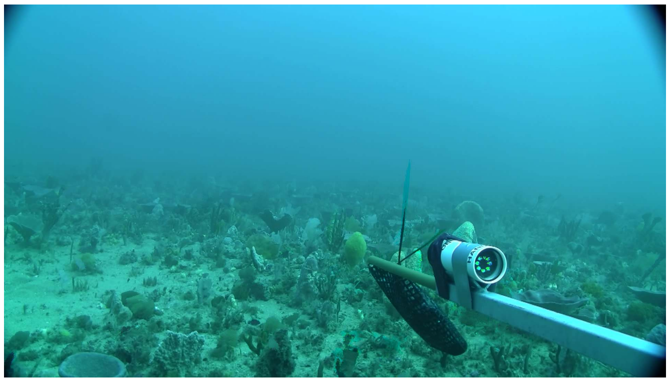

Fig. 2. The extracted background using 1 minute of footage from the left camera.

2) Identification: With a focus on running time, fairly simple methods were chosen to identify fish in the video footage. We perform a simple absolute background subtraction on each frame to isolate dynamic (foreground) elements. Then, we apply a median filter to the image to reduce noise and flatten similar sections into a contour-like map of the image. The result can be seen in Figure 3. A threshold value is generated by analysing the shape of the resulting histogram, and this value allows a thresholding that generates a binary image with dynamic objects represented by white shapes. This approach allows very simple contour extraction, by simply taking the boundary of the extracted foreground region. These contours are then used to describe the shape of the identified fish in later processes; However, this approach makes no distinction between fish that overlap, which may cause deformation of the contour. This common situation is instead handled in later sections.

3) Tracking: Tracking fish across frames allows the system to distinguish between fish and noise, which is typically static,

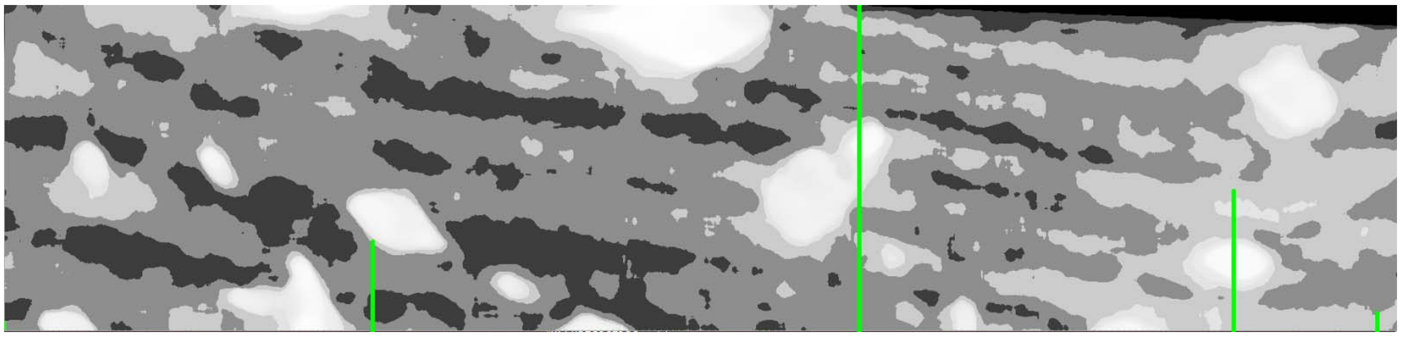

Fig. 3. Frame prior to thresholding. Light shades indicate foreground. The image histogram is overlaid in green.

and may 'flicker' between frames. It can also be used as a way to provide information that cannot be found by looking at individual static images. We demonstrate a number of techniques in this module that are used to analyse the footage.

Most importantly, the tracking process involves 'tagging' identified fish by comparing them to previously seen individuals. This can be calculated by comparing all newly identified fish with those seen in the previous frame. First we compare by location, since fish have a limited speed and a single individual is unlikely to move by more than its length across two frames. For any conflicts that arise by fish being physically close, fish can be matched by comparing physical characteristics including size, direction of motion and mean pixel intensities. When each fish has been tagged with a unique identifier, the system can keep track of each fish, including when they leave the frame. These identifiers are maintained across all frames, allowing a fish count to be estimated by simply counting the number of identifiers allocated.

Once this basic tracking step has been done, further information can be drawn from the N previous frames in order to improve fish detection rate. In particular, this approach was targeted at counteracting the effects of overlapping or obstructed fish. If any tracked and tagged fish (the obstructed) is not found in the latest frame and another fish (the obstructor) occupies the location where the obstructed was last seen, we assume the fish identification has merged the two, or the obstructed fish is completely within the contour of the obstructor. This causes two problems with tracking - the obstructed fish is lost from the frame (and hence is lost from the tracker), and the contour of the obstructor becomes deformed by the addition of the obstructed.

We detect the obstruction of fish by comparing the locations of any fish lost from the tracking with the locations of fish which were identified in the current frame. Once the obstruction has been detected, we add the lost fish to the set of fish seen in the frame, maintaining the last-seen contour and velocity. This process is repeated for each frame in which the fish remains lost. When new fish are identified in frame, we compare their physical properties to the fish that was lost, if they have been identified close to the outline of the obstructed. Using this technique, the obstructed fish is picked up after it re-emerges from the obstructor. An example of this process can be seen in Figure 4, which shows a pair of fish after obstruction, immediately prior to re-emergence, and following reidentification.

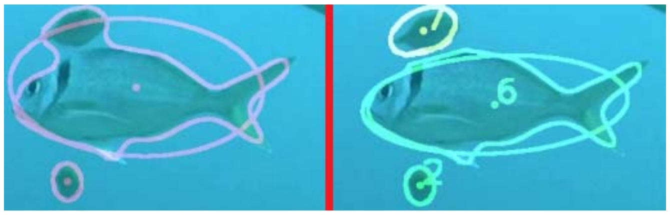

Fig. 5. Merged fish outline, before and after tagging and contour adjustment.

To fix the contour deformation, we compare the new contour of the obstructor with the old contour of the obstructed. Any contour points of the obstructor that fall within the last-known contour of the obstructed are discarded. Meanwhile, the contour of the obstructed fish is maintained as its last known contour, as we assume (lacking evidence to the contrary) that the fish does not deform while we cannot see it. This provides a reasonable estimate for the correct outline of both fish, given a prior contour existed for each. Figure 5 shows a pair of fish which have been identified as merged. The purple contour is clearly not a correct fish outline, and the best-fitting ellipse (used to estimate fish size and angle) is poorly fitted. After the tracking process has been applied, the last-known contour has been applied to the obstructed (fish 7) and shows some

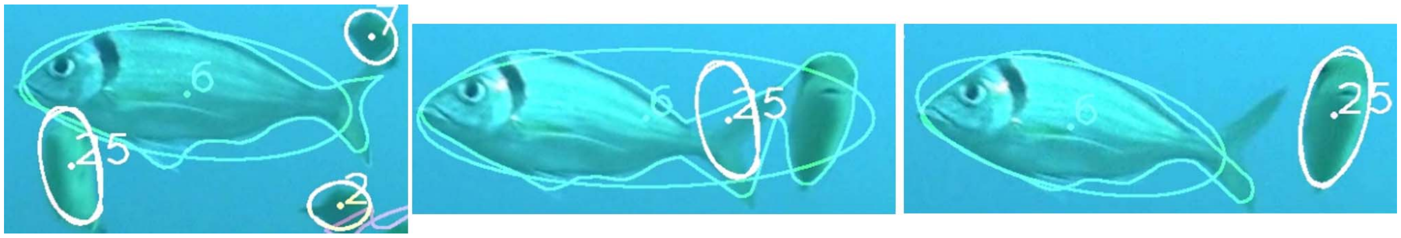

Fig. 4. Fish before obstruction, immediately before clearing, and after clearing the obstruction.

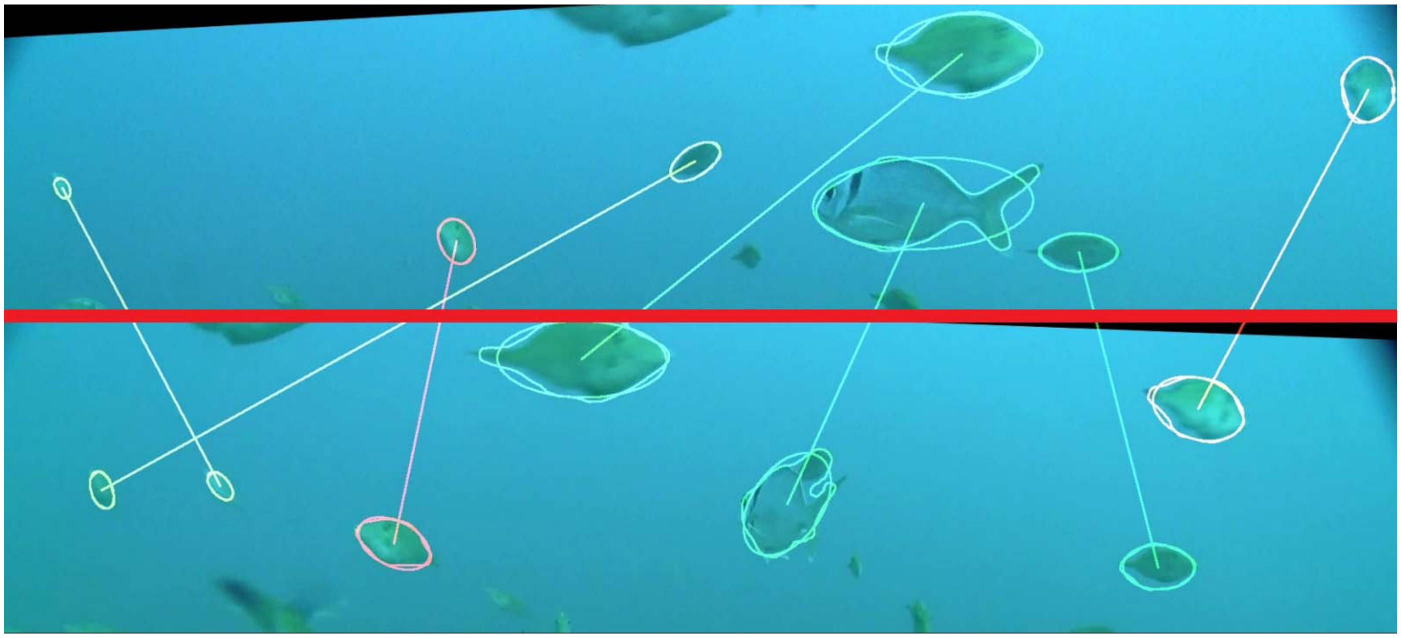
Fig. 6. Example of matched fish. Left frame is on top, right frame is on bottom.

displacement from the actual outline. The contour for the obstructor (fish 6) has been adjusted to remove the merge, leaving only a slight ridge and an ellipse that is a far better estimate for fish size.

4) Measurement: The implementation of this module depends on the data available, but involves three main parts: Calculating disparity between left and right fish, finding the snout and tail points, and performing the calculations (the last step being covered in other works [17], we will avoid discussing it here).

There are a number of ways to calculate disparity between left and right images, either with a dense approach (using all points to create a depth map) or a sparse one (using only points of interest) [18], [19], [20]. This process can be simplified by rectifying the images using the intrinsic and extrinsic camera parameters [21], or using fundamental matrices [22]. We applied rectification to the images before applying any other processes. We used sparse disparity to calculate real-world coordinates. This required identifying points in the left image that correspond with points in the right. We approached this by comparing the known locations of tracked fish in the left and right frames. This was feasible due to the principles of epipolar geometry – since the images were rectified, matching fish had to be on the same horizontal line [23], [21]. Despite this, there were typically a large set of potential matches, including fish that could only be seen in one image.

In order to sort through the potential stereo matches, we analyse the colour intensities of the image. We generate and compare the histograms for the hue and saturation channel for each pixel within the contours of the fish. This provides a reasonable mapping from left to right fish, and the result can be seen in Figure 6.

Once we know which fish correspond we know that, by extension, the snouts and tails correspond as well. Rather than finding the true snout and tail points, we define an ellipse of best fit around the contour of the fish. We can then estimate the snout and tail points as the major axis endpoints of this ellipse. We then refine these approximations by finding nearby contour points which match the corresponding fish's contour by sitting on the same epipolar line. This was found to provide a reasonable approximation to the snout and tail points, as can be seen in Figure 7.

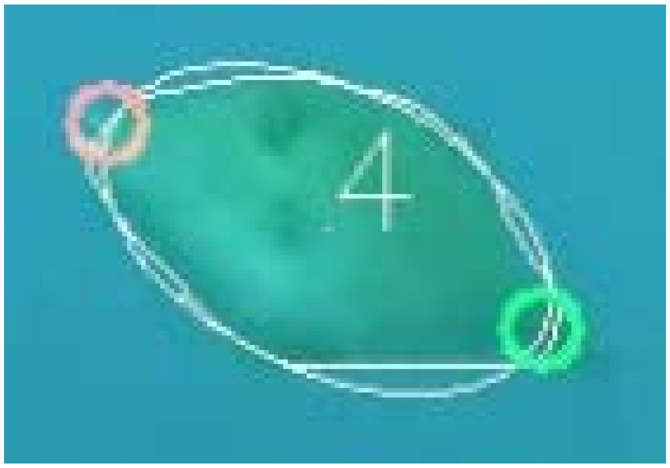

Fig. 7. A fish with estimated snout and tail points marked.

C. Learning

Learning algorithms, if properly trained, can be used to classify candidate regions as fish or 'not-fish' [12]. They can also be used as multi-class classifier, either attempting to label the fish type [10] or identifying a candidate as a correct fish, partially correct fish, or noise.

To use supervised learning algorithms, a set of labelled training data is required. Existing libraries of fish images are not necessarily sufficient for this, due to the specific effects of lighting, background, depth and water colour. Instead, we generated training data by storing non-validated candidate regions identified during a typical run of the system. Then, we labelled these images according to the desired learning classes and grouped them with a set of numerical feature descriptors developed for use as inputs to learning algorithms.

In order to achieve this, we needed to define a set of features with which we can classify fish. If the wrong features are selected, a classifier will never converge to a successful classification, so this is often a major step for machine learning research [10]. The features we use are based on colour and shape of individual fish, rather than involving the relationships between fish. To describe the colour of the fish, we use the mean and variance for the RGB colour channels of the pixels within the fish contour. For the shape features, we use the contour area and the average convexity defect. The convexity defect for each section of the contour is the maximum deflection from the convex hull, where the section is defined by all contour points bounded by consecutive points in the hull. We also use Hu's 7 invariant moments [24] as numerical shape descriptors. These features are trivial to compute from the fish images, and can be used to improve the accuracy of the fish identification by discarding candidate regions that are not classified as fish.

IV. Experimental Results

A. Raw System

In Figure 8, an example of annotated final output can be seen, with identified contours, assigned IDs and measurements overlaying the original images. However, this frame shows a reduced view of the frame, with the noisy part of image (including the seafloor) excluded to simplify development. When the noise is included, the simple methods used for identification achieved a raw accuracy of 43% (56.57% of candidates were 'junk').

Since a focus of automating this identification is to improve time costs, the timing of the system is analysed also. An output from running 100 frames, one-by-one, is provided in Figure 9, and shows that the major time costs are incurred at the identification stage (background subtraction and contour extraction).

```
Average time required per section:
Rectification:         0.1572 s
BG Subtraction:        0.2176 s
Contour Extraction:    0.1717 s
Tracking:              0.1619 s
Matching:              0.0269 s
Measuring:             0.0344 s
Drawing:               0.0217 s

Time per frame: 0.7914 s
Time per second of footage: 19.785 s
```

Fig. 9. Timing output.

B. Learners

To test the attributes selected for learning, the WEKA machine-learning toolkit [25] was used with manually labelled candidate fish. The training data was tested under multiple conditions. We trained binary classifiers using the dataset, first with a subset of attributes and then with all fifteen. We also trained classifiers that labelled fish as 'fish', 'junk', or 'fish with incorrect contour', on both the subset and full set of features. Finally, we tested the learners on a second set of labelled data, from BRUVS footage which identified far more noise than correct candidates.

The datasets were tested using variants of multilayer perceptrons (MLPs) and decision trees using a single layer and multiple layers. The ZeroR algorithm (which predicts the mode

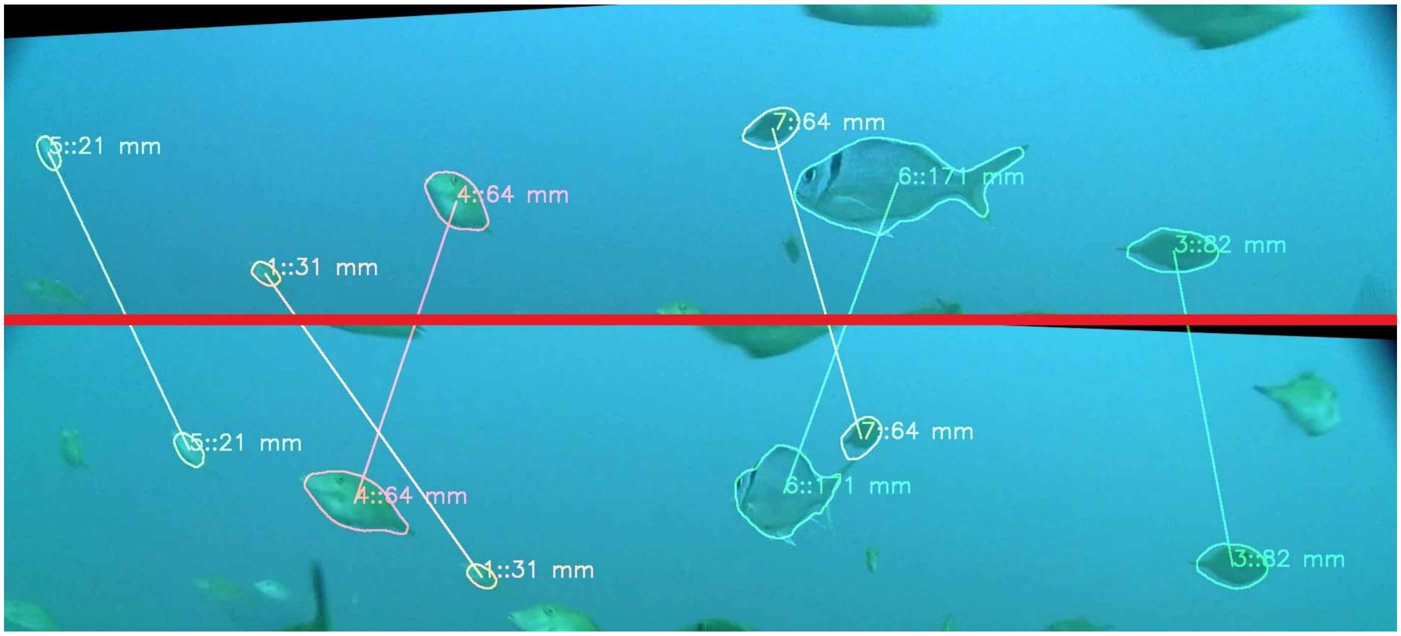

Fig. 8. Example of annotated final output.

of a discrete class as the class for all instances) was used as a baseline against which to compare the others to. Table I shows the results from these learners. The numbers indicate the percentage of correctly classified candidates, and results that were significantly better than the baseline are marked with an asterisk. It can be seen that when all attributes are used, all tested learning algorithms are able to improve on the raw performance of the identifier, from less than 60% to over 80% in the case of binary classification. This indicates that any of these learning algorithms could be successfully integrated into a counting and measurement system using the numeric attributes described.

Conversely, the results also show that none of the learners improved on ZeroR when the raw system had a high failure rate. This indicates that learning with this method cannot replace a well-developed identification algorithm, merely augment an imperfect one.

TABLE I. WEKA TESTING OUTPUT. '*' REPRESENTS A STATISTICALLY SIGNIFICANT IMPROVEMENT ON ZERoR. 'ZERoR' IS A ZERO-RULE CLASSIFIER. 'STUMP' IS THE DECISIONSTUMP ALGORITHM (A 1-LAYER DECISION TREE). 'J48 TREE' IS A DECISION TREE USING THE J48 ALGORITHM. '0L MLP' IS A SINGLE-LAYER PERCEPTRON ALGORITHM. 'MLP' IS A MULTI-LAYER PERCEPTRON ALGORITHM.

Dataset	ZeroR	Stump	J48 Tree	0L MLP	MLP
Set 1 Binary Attribs-	58.57	64.57*	72.29*	62.36*	63.07
Set 1 Binary Attribs+	56.57	84.56*	83.90*	84.12*	82.47*
Set 1 Tertiary Attribs-	39.29	42.50	62.50*	63.50*	68.07*
Set 1 Tertiary Attribs+	56.57	66.86*	70.81*	74.65*	74.16*
Set 2	85.67	84.67	82.00	82.00	80.93

V. CONCLUSIONS AND FUTURE WORK

A framework was suggested for a generalised automated fish counting and measurement system, along with a particular approach for implementing each part of this framework. Also, a feature set for machine learning was presented which can improve performance of a simple implementation of the identification portion of the framework significantly.

Further work on the implementation of the framework could incorporate techniques such as Gaussian Mixture Models [26] or Optical Flow [27] to improve identification and noise handling. The machine learning algorithms conceptually proven using WEKA should also be incorporated into the implementation. It should be feasible to provide a training set during the operation of the system rather than prior to it, by using a scoring learner which outputs a confidence value representing the probability that the candidate is a fish. This learner can then request human input if the confidence is not high enough, providing a system which learns to not need human intervention, but can automatically adapt to different environments, lighting conditions, and fish species.

ACKNOWLEDGEMENTS

We acknowledge Mr Justin Hulls, at the University of Tasmania, for providing the stereo BRUVS footage used for development and testing.

REFERENCES

[1] C. Costa, A. Loy, S. Cataudella, D. Davis, and M. Scardi, "Extracting fish size using dual underwater cameras," *Aquacultural Engineering*, vol. 35, no. 3, pp. 218–227, 2006.

[2] C. Spampinato, Y.-H. Chen-Burger, G. Nadarajan, and R. Fisher, "Detecting, tracking and counting fish in low quality unconstrained underwater videos," in *Proceedings from the 3rd International Conference on Computer Vision Theory and Applications (VISAPP)*, vol. 2, 2008, pp. 514–519.

[3] C. Johansson, M. Stowar, and M. Cappo, "The use of stereo BRUVS for measuring fish size; Report to the Marine and Tropical Sciences Research Facility," Reef and Rainforest Research Centre and Australian Institute of Marine Science, report, January 2008.

[4] A. Marouchos, M. Sherlock, B. Barker, and A. Williams, "Development of a stereo deepwater Baited Remote Underwater Video System (DeepBRUVS)," in *OCEANS, Spain*. IEEE, June 2011, pp. 1–5.

[5] D.-J. Lee, J. Archibald, R. Schoenberger, A. Dennis, and D. Shiozawa, "Contour Matching for Fish Species Recognition and Migration Monitoring," in *Applications of Computational Intelligence in Biology*, ser. Studies in Computational Intelligence, T. Smolinski, M. Milanova, and A.-E. Hassanien, Eds. Springer Berlin Heidelberg, 2008, vol. 122, pp. 183–207. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-78534-7_8

[6] J. Han, A. Asada, H. Takahashi, and K. Sawada, "Automated three-dimensional measurement method of in situ fish with a stereo camera," in *OCEANS 2010 IEEE - Sydney*, May 2010, pp. 1–5.

[7] N. Abdullah, M. Shafry, M. Rahim, and I. Amin, "Measuring fish length from digital images (filedi)," in *Proceedings of the 2nd International Conference on Interaction Sciences: Information Technology, Culture and Human*, ser. ICIS '09. New York, NY, USA: ACM, 2009, pp. 38–43. [Online]. Available: http://doi.acm.org/10.1145/1655925.1655932

[8] R. Tillett, N. McFarlane, and J. Lines, "Estimating Dimensions of Free-Swimming Fish Using 3D Point Distribution Models," *Computer Vision and Image Understanding*, vol. 79, no. 1, pp. 123–141, 2000.

[9] A. Rova, G. Mori, and L. Dill, "One fish, two fish, butterfish, trumpeter: Recognizing fish in underwater video," in *IAPR Conference on Machine Vision Applications*, 2007.

[10] P. Huang, B. Boom, and R. Fisher, "Underwater Live Fish Recognition Using a Balance-Guaranteed Optimized Tree," in *Computer Vision – ACCV 2012*, ser. Lecture Notes in Computer Science, K.-M. Lee, Y. Matsushita, J. Rehg, and Z. Hu, Eds. Springer Berlin Heidelberg, 2013, vol. 7724, pp. 422–433.

[11] R. Larsen, H. Olafsdottir, and B. Ersbøll, "Shape and Texture Based Classification of Fish Species," in *Image Analysis*, ser. Lecture Notes in Computer Science, A.-B. Salberg, J. Hardeberg, and R. Jenssen, Eds. Springer Berlin Heidelberg, 2009, vol. 5575, pp. 745–749. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-02230-2_76

[12] M. Shortis, M. Ravanbakskh, F. Shaifat, E. Harvey, A. Mian, J. Seager, P. Culverhouse, D. Cline, and D. Edgington, "A review of techniques for the identification and measurement of fish in underwater stereo-video image sequences," in *Proceedings of SPIE, Videometrics, Range Imaging, and Applications XII; and Automated Visual Inspection*, vol. 8791, 2013, pp. 87 910G–87 910G–10. [Online]. Available: http://dx.doi.org/10.1117/12.2020941

[13] C. Spampinato, D. Giordano, R. Di Salvo, Y.-H. Chen-Burger, R. B. Fisher, and G. Nadarajan, "Automatic fish classification for underwater species behavior understanding," in *Proceedings of the First ACM International Workshop on Analysis and Retrieval of Tracked Events and Motion in Imagery Streams*, ser. ARTEMIS '10. New York, NY, USA: ACM, 2010, pp. 45–50. [Online]. Available: http://doi.acm.org/10.1145/1877868.1877881

[14] S. Rosen, T. Jörgensen, D. Hammersland-White, and J. Holst, "Deepvision: a stereo camera system provides highly accurate counts and lengths of fish passing inside a trawl," *Canadian Journal of Fisheries and Aquatic Sciences*, vol. 70, no. 10, pp. 1456–1467, 2013. [Online]. Available: http://dx.doi.org/10.1139/cjfas-2013-0124

[15] N. Buch, S. Velastin, and J. Orwell, "A Review of Computer Vision Techniques for the Analysis of Urban Traffic," *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 3, pp. 920–939, Sept 2011.

[16] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[17] Y. Mustafah, R. Noor, H. Hasbi, and A. Azma, "Stereo vision images processing for real-time object distance and size measurements," in *In-*

ternational Conference on Computer and Communication Engineering (ICCCE), July 2012, pp. 659–663.

[18] K. Mühlmann, D. Maier, J. Hesser, and R. Männer, "Calculating dense disparity maps from color stereo images, an efficient implementation," *International Journal of Computer Vision*, vol. 47, no. 1-3, pp. 79–88, 2002.

[19] S. Hawe, M. Kleinsteuber, and K. Diepold, "Dense disparity maps from sparse disparity measurements," in *IEEE International Conference on Computer Vision (ICCV)*, Nov 2011, pp. 2126–2133.

[20] C. Sun, "Fast Stereo Matching Using Rectangular Subregioning and 3D Maximum-Surface Techniques," *International Journal of Computer Vision*, vol. 47, no. 1/2/3, pp. 99–117, May 2002.

[21] A. Fusiello, E. Trucco, and A. Verri, "A compact algorithm for rectification of stereo pairs," *Machine Vision and Applications*, vol. 12, no. 1, pp. 16–22, 2000.

[22] C. Sun, "Closed-form Stereo Image Rectification," in *Image and Vision Computing New Zealand*. Dunedin, New Zealand: ACM, 26-28 November 2012, pp. 132–137.

[23] G. Xu and Z. Zhang, *Epipolar geometry in stereo, motion and object recognition: a unified approach*. Springer, 1996, vol. 6.

[24] M.-K. Hu, "Visual pattern recognition by moment invariants," *IRE Transactions on Information Theory*, vol. 8, no. 2, pp. 179–187, February 1962.

[25] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. Witten, "The WEKA Data Mining Software: An Update," *SIGKDD Explor. Newsl.*, vol. 11, no. 1, pp. 10–18, Nov. 2009. [Online]. Available: http://doi.acm.org/10.1145/1656274.1656278

[26] Z. Zivkovic, "Improved adaptive Gaussian mixture model for background subtraction," in *Proceedings of the 17th International Conference on Pattern Recognition (ICPR)*, vol. 2, Aug 2004, pp. 28–31.

[27] J. S. Pérez, E. Meinhardt-Llopis, and G. Facciolo, "TV-L1 Optical Flow Estimation," *Image Processing On Line*, vol. 3, pp. 137–150, 2013.