Automatic Texture and Anomaly Mapping in Under-ice Video Datasets

Anthony Spears, Ayanna Howard
Electrical and Computer Engineering Department
Georgia Institute of Technology
Atlanta, GA

Michael West, Thomas Collins
Georgia Tech Research Institute
Atlanta, GA

Abstract— **The exploration of under-ice environments has seen increased interest over the past few years due to advances in technological capabilities, such as autonomous underwater vehicles (AUVs), as well as interest in exploration of polar regions and Jupiter's ice-covered moon Europa. Searching for interesting features under the ice, including animals capable of sustaining life in such harsh environments, is of great interest in both polar (Antarctica) and planetary (Europa) domains. Under-ice environments, such as those encountered beneath the Antarctic ice shelves, are largely devoid of such features and tend to be monochromatic centered on the blues of the ice. Post-processing of under-ice datasets can be very tedious for human analysts. Presented here are algorithms to aid in the post-processing of such large and mostly featureless datasets. Two novel algorithms are presented here which use point-feature detections in video frames to estimate texture (number and spread of features) and anomaly locations (dense groupings of features). Two additional algorithms are proposed which use hue-based methods to estimate the percentage of non-ice pixels present in the video frames and to detect anomalous colored pixel groups corresponding to candidate anomalies against the background of the ice. These algorithms are presented herein along with results from testing with both simulated and real-world under-ice video datasets.**

Keywords—under-ice, underwater, AUV, mapping, texture, video

I. Introduction

The exploration of under-ice environments has seen increased interest over the past few years due to advances in technological capabilities, such as autonomous underwater vehicles (AUVs), as well as interest in exploration of polar regions and Jupiter's ice-covered moon Europa. Searching for interesting features under the ice, including animals capable of sustaining life in such harsh environments, is of great interest in both polar (Antarctica) and planetary (Europa) domains. Underwater environments are known to be largely featureless, even at the seafloor which can many times consist only of monochrome sand or rock. Under-ice environments such as those encountered beneath the Antarctic ice shelves are even more devoid of features and tend to be monochromatic centered on the blues of the ice. With the advent of remotely operated underwater vehicles (ROVs) and autonomous underwater vehicles (AUVs) has come a large number of video datasets taken in these underwater and under-ice environments. Analyzing these videos can be tedious from both the perspective of human operators (ROVs) and scientists performing post-processing (AUVs), as most of the dataset contains no "interesting" information or unique features. However, all frames must be carefully analyzed to find the few frames of interest. We propose here a method for automatically highlighting a relatively feature-rich or uniquely colored frame to bring it to the attention of an analyst. If an estimate of the vehicle location is available, these frames or anomalies of interest can be mapped to create a global view of these anomalies over the vehicle's trajectory. Using point features (already detected by another algorithm in this application), we propose a novel method for estimating the overall texture of the background ice in an image. These same point features are also used in another proposed algorithm to detect any dense groupings of features to mark as an anomaly candidate. A third proposed algorithm presented here uses a histogram of the hues detected in the image to find any groupings of anomalous hues outside of the blues corresponding to the ice background to mark as anomaly candidates. A final algorithm also provides a measurement of the pixels outside the blue hue range as an estimate of non-ice pixels in the image. A combination of these four approaches can be used to map the texture and anomalous colors of the environment over the vehicle's trajectory, and to map anomalies detected in the globally feature- and color-poor under-ice datasets of interest. The proposed methods are evaluated using both simulated data of sub-ice environments and real under-ice data obtained from Lake John, Colorado and McMurdo, Antarctica.

II. Background

The use of point features (pixel locations with large, unique local gradients) such as SURF (Speeded Up Robust Features) [8] features or Shi-Tomasi corners [9] for texture estimation in an image has been explored in some previous work for categorizing images [1]. However, previous methods require extensive processing and have not been applied to underwater or under-ice autonomous systems such as the application under consideration here. Kim and Eustice [2] use point features to estimate saliency (uniqueness) and feature-richness of an image, which requires great processing effort to calculate. We propose an algorithm with very low additional computational cost to get a real time estimate of the texture in an image using point features that have already been extracted for use in another algorithm in the system. While a few previous underwater vision applications use point features in video to perform SLAM [2], none currently use the features detected to estimate if there is an anomaly of interest in the image across a globally feature poor dataset (commonly encountered in underwater and under-ice environments). Such a method for detection of anomalies in a largely featureless under-ice environment is proposed here. Color is commonly used for

978-0-933957-43-5 ©2015 MTS

dissecting an image into foreground and background or segments [4], even in underwater applications [3]. However, such a color-based approach is not commonly used in under-ice applications due to the nature of the environment as largely monochrome and featureless. However, our unique application requires a means for searching the monochrome ice background for any anomalies (specifically animals) that might be located at the ice-water boundary. Therefore, we propose a color-based algorithm using hue histograms and clustering to determine outlying colors corresponding to anomalies in under-ice environments. The under-ice AUV frontier is relatively new in and of itself, and we hope the novel use of the proposed algorithms presented here will help to advance capabilities in the field..

III. ALGORITHMS

The four algorithms presented in this work are detailed in this section.

A. Texture Estimation Algorithm

An estimation of texture in an image can give insight into the content of the image with a single quantitative value. Mapping the texture of video frames over a large vehicle trajectory can indicate areas of interest during post processing, as well as give insight into the type and texture of background ice present in the application considered here. Ice can vary drastically in texture from first year ice that can be only a few inches thick and very smooth with cracks, to multi-year ice that is smooth but varies greatly in topography, to platelet ice that is comprised of a textured mix of small ice fragments at the ice-water boundary, to drastically textured frazil ice comprised of half-formed crystals of ice at the ice-water boundary. In some sub-ice regions, multiple types of ice can be encountered during a single AUV mission. In Antarctica some under-ice locations encounter large currents where any frazil or platelet ice is swept away to leave only the smooth ice beneath, while other locations are sheltered from these currents and platelet ice can accumulate multiple meters in thickness. A mapping of estimated texture throughout under-ice mission video datasets can provide a global view of the ice types encountered over the mission. This mapping can also give scientists an idea of what to expect from a frame during post-analysis.

The algorithm proposed here uses point-features already extracted from a video frame (during visual odometry) to calculate an estimate of ice texture. The model proposed here for a high texture value is the case of a large number of detected point features distributed evenly over the image plane. Given the point features extracted from the image, the algorithm creates a coverage area for each pixel which includes all pixels within a certain radius (20 pixels here) of the point feature. A binary mask with the dimensions of the input image is then created from the summation of each of these individual coverage areas where a pixel inside at least one coverage area takes the value of one and any pixel not covered by any point feature coverage area takes the value of zero. From this binary mask, much information can be gleaned about the distribution of the point feature pixels. In the case of this algorithm, the total number of covered pixels can be easily and quickly determined with a summation over all pixels in the mask image. The ratio of covered pixels over total image pixels gives an estimate of texture which is normalized over image size. The pseudo code for the main loop of this algorithm is shown in Figure 1. Both Shi-Tomasi GFTT (Good Features to Track) [9] features and SURF (Speeded Up Robust Features) [8] features were used during the development of this algorithm. GFTT features and SURF features each performed well on different datasets and seem to complement each other well in this application. A combined system using both GFTT and SURF features is proposed in this algorithm to provide a more robust system.

Pseudo Code (Texture Estimation Loop):
```
Get input frame and initialize masks to all zeroes
Extract SURF and GFTT Feature Locations
Draw coverage circles for each feature (r = 20 pix, value=1)
maskImg = and(SURF mask, GFTT mask)
pointCover = sum(maskImg)
pointCoverPct =
pointCover/(maskImg.width*maskImg.height)
```

Fig. 1. Pseudo code for main loop of texture estimation algorithm

To see how this algorithm obtains an estimate of ice texture, we consider three examples. In the first case, a drastically textured image would have point detections throughout the image plane that are fairly evenly distributed. This would lead to a very small amount of overlap between the coverage zones of the point features, and therefore result in a high value for total coverage and texture. In the case of an anomaly located in the middle of smooth and featureless ice, almost all point features detected would be tightly grouped around the anomaly with little coverage over the rest of the image. This would lead to a large amount of overlap between coverage areas as the point feature pixels are densely grouped, and therefore a small value would be obtained for a coverage and texture estimate. In the third case, mostly smooth ice with no anomalies would only have a few point features which would be fairly evenly distributed. Despite the minimal overlap in coverage areas, the small number of features would lead to a low value for total coverage and texture. A high value for texture can only be obtained when a large number of features are detected and those features are evenly distributed over the image plane. This ensures the algorithm is robust against a large number of densely grouped features, such as in the case of an anomaly in the ice. This algorithm provides the means for quickly calculating an estimate of the background texture in an image based on point features and their distribution over the image plane.

B. Point Feature Anomaly Detection Algorithm

The distribution of detected point features in the image plane can not only give an idea about the texture over the entire image, but can also indicate the presence of an object of interest, or anomaly, in the image. Here, we propose modelling such a small object or anomaly against a mostly featureless background as a tight grouping of point features in a small area. The algorithm developed here to detect such anomalies in a frame of video uses point features already extracted for visual odometry in the system as input. The algorithm first determines groupings for the detected features using a k-means [10] based method which partitions the points (both GFTT and SURF)

into k clusters where each point is part of the group with the nearest mean. The number of clusters (k) varies here between each frame and is calculated as in Equation 1 where *corners* is the number of features detected and CORNERS_PER_CLUSTER is a constant (20 here). Ideally, a closely grouped set of features (such as that corresponding to an anomaly) will form at least one group by itself due to the relative local density of the points. In contrast, a textured image with evenly distributed feature points will yield widely distributed groupings, and a mostly featureless image will have very few points per group where these points will be widely spread out. In order to determine if a group of extracted feature points form an anomaly or object of interest, the number of features within a 30x30 pixel square surrounding the group mean is considered. In order to normalize the threshold of anomaly detection over the total number of features detected in the image, the ratio of feature points within the centroid square over the total number of group features is used as the metric for determining if a group comprises an anomaly in the image. In our algorithm, this threshold is set at 0.8, which requires 80% percent of the total feature points in the group to be within that group's centroid box to be classified as an area of interest (Equation 2 where mppf is the minimum points for a feature, k is the number of clusters, and PCT_PER_CLUSTER is 80% here). In this manner, density of feature points is used to estimate whether an object of interest, or anomaly, is present in each video frame. The pseudo code for the main loop of this algorithm is shown in Figure 2.

$$k = corners/CORNERS\_PER\_CLUSTER \quad (1)$$

$$mppf = k*PCT\_PER\_CLUSTER \quad (2)$$

Pseudo Code (Texture-based Anomaly Detection Loop):

```
Get input frame
Extract SURF/GFTT feature locations
k-means clustering of SURF/GFTT features (k from Eq. 1)
mppf = k*PCT_PER_CLUSTER (Equation 2)
FOR i=0; i<k; i++ {
    pointsBox = SUM (cluster[i] points within centroid box)
    IF pointsBox > mppf
        Anomaly detected }
```

Fig. 2. Pseudo code for main loop of texture-based anomaly algorithm

C. Hue Mapping Algorithm

In addition to the use of point feature distribution as a means for detecting anomalies or areas of interest in a frame, a second method is proposed here which uses hue to find such anomalies. In this case, we propose that the model of hue distribution in an under-ice image consists mainly of blues corresponding to the ice background. Therefore, any pixel groupings outside of the main blue hues of the ice can be considered as anomaly candidates. The proposed algorithm first splits the input video frame into hue, saturation, and value channels, each consisting of a single matrix of 8 bit values the same size as the input image. Hue represents the actual color of the image while value represents brightness and saturation represents color strength. The hue channel is the only one used in this algorithm as it provides information about the color of each pixel, which is used to find anomalous colors in the image. In theory hue values take the form of degrees in a circle and can range from zero back around to 360, both of which correspond to red (Figure 3). However, because 8 bit values (0:255) are used here, this spectrum is halved to fit between 0 and 179 and stay below the 255 maximum value.

Fig. 3. Hue spectrum which typically varies from 0 to 359, but here is divided by two to fit into an 8-bit representation, and so varies from 0 to 179.

We propose here an algorithm to obtain a single quantitative measurement for color in a video frame that provides an indication of how much anomalous color is present in each frame throughout an under-ice video dataset. To obtain this value, each frame is first filtered to eliminate all pixels with a hue corresponding to the blue range (82-151 here). Then all remaining pixels are summed to obtain the total number of non-blue pixels in the frame. This value is normalized over the total number of pixels in the frame to obtain a percentage of pixels that are not within the blue band of hue values. The pseudo code for this algorithm is shown in Figure 4. During analysis, this value can provide indications of frames of interest for post-analysis located at local maximums where an abnormal amount of non-blue color is present in the image.

Pseudo Code (Color Estimation Loop):

```
frame = Get input frame
threshBlueImg1 = THRESHOLD(frame, values above 151)
threshBlueImg1 = THRESHOLD(frame, values below 82)
threshBlueImg = threshBlueImg1 + threshBlueImg2
pixNotBlue = SUM(threshBlueImg)/255
pctNotBlue = pixNotBlue/(frame.width*frame.height)
```

Fig. 4. Pseudo code for main loop of color estimation algorithm

D. Hue-based Anomaly Detection Algorithm

In addition to obtaining a quantitative value for relative colorfulness of a video frame, we propose an algorithm which uses hue to detect anomalies in an under-ice image. In this proposed algorithm, a histogram is created from the input image over all 180 possible values of hue with a bin width of one. The local maximums of the histogram are used as candidates for anomalous colors in the image, as the proposed model assumes a Gaussian-like distribution around at least one hue value for any color grouping in the image. Two normalized thresholds are used to eliminate hues with small and large histogram values, assumed to correspond to noise and background (ice or water) respectively. These thresholds are set at 0.1 and 0.0001 to eliminate from consideration any hue local maximum that represents over 10% of the total pixels or under 0.01% of the total image pixels. In order to reduce the chance that local maximums corresponding to the blues of water or ice are considered to be anomalous, all hue values between 81 and 151 (on a 180 scale) are ignored in a form of band-stop filtering. The values of 0 and 179 (on a 180 scale) are also ignored as they most often correspond to the whites and blacks in the image. While these values also correspond to the red hue, they can be safely ignored as red is very uncommon in underwater imagery. The remaining local maximums in the hue histogram are used to add the

corresponding hue pixels to an output mask along with any pixels of the hue values on either side of the local maximum. The result of this algorithm is a mask of binary values the same size as the input image where a value of one represents a pixel that is of a hue considered to be possibly anomalous and a zero represents background. The pixels of the output mask are then clustered into groups using a connected components method where a connected component is considered to be a continuous group of touching pixels. A normalized threshold is used here to eliminate small (noise) groupings of pixels. This threshold is set at 0.0003 (MIN_PIX in Figure 5) to eliminate any pixel groupings with an area smaller than 0.03% of the image size. Pseudo code for this algorithm is shown in Figure 5. The output result is a number of candidate components in the image that are considered anomalous in hue to the rest of the image, which mostly comprises of the blues of the ice. In this way, we propose to automatically detect any ice anomalies, such as animals, present at the ice-water boundary.

```
Pseudo Code (Local Max Hue-based Anomaly Detection):
frame = Get input frame
hueHistogram[180] = calculateHistogram(frame)
FOR i=0; i<180; i++
        IF (i < 151) AND (i > 81)
                CONTINUE
        ELSE IF (hueHistogram[i] > MAX_PIX)
             OR (hueHistogram[i] < MIN_PIX)
                CONTINUE
        ELSE IF (hueHistogram[i] > hueHistogram[i-1])
            AND (hueHistogram[i] > hueHistogram[i+1])
                Add pixels in hueHistogram[i] to maskImg
dilate(maskImg, 4 times)
erode(maskImg, 4 times)
components[] = find connected components (threshBlueImg)
FOR i=0; i<components.size(); i++ {
        IF component[i].area() > AREA_THRESHOLD
                Anomaly detected }
Clear maskImg to all zeroes
```

Fig. 5. Pseudo code (main loop) for local-max hue-based anomaly detection

IV. Datasets Used For Algorithm Evaluation

In order to evaluate these algorithms, simulated video data of under-ice environments (with inherent ground truth) was used along with real under-ice data collected in Lake John, Colorado and McMurdo, Antarctica. Other real-world, under-ice animal imagery previously collected [11] is also used to further validate the proposed algorithms on real data with anomalies. The simulated data was created using Blender [12] which is a simulation engine that can be used to create visually accurate imagery of an environment.

V. Algorithm Results

The proposed algorithms were first tested on simulated data, as ground truth was ready available and highly accurate. The main dataset used to validate the algorithms was an upward-looking camera view of the simulated ice environment. This ice environment spans a very large area and contains regions of first-year ice, multi-year ice, frazil ice, and platelet ice. A ground truth map of this simulated ice environment is shown in Figure 6. Ice anomalies were introduced across the simulated ice in the form of ice deformations, ice cracks, and various colored animal shapes similar to those encountered under the ice in Antarctica. The camera path was a translational rectangle of surge and sway (no rotational motion) where the vehicle returns to the starting point at the end of the trajectory. Because an estimate (ground truth in this case) of the camera location for each frame was known, maps could be created over the trajectory to show estimated textures as well as the location of ice anomalies using the camera field of view and known altimetry (distance to ice).

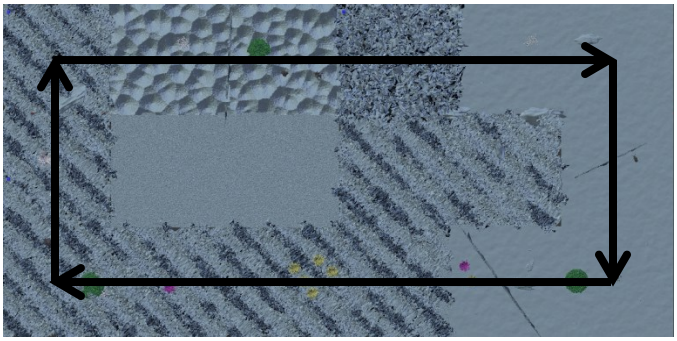


Fig. 6. Ground truth map of simulated under-ice dataset. This view is looking up from beneath the ice at the entire environment with the various ice types and ice anomalies.

A. Texture Estimation Results

The results from evaluation of the texture estimation algorithm on simulated under-ice data can be seen in Figure 7 and Figure 8. Here the left column shows the original image with point feature detections as colored dot groups while the right column shows a visualization of the point cover result mask, used to estimate texture. A point cover image with more white pixels corresponds to a highly textured image while a point cover image with mostly black pixels corresponds to an image with a low texture value. As expected, the first-year ice shown in Figure 7a yields only a few, closely grouped point features, which results in a low value for point cover. At the other end of the spectrum, the frazil ice image (Figure 7d) yields many point-feature detections that are distributed evenly over the image, leading to a point cover image of mostly white pixels which corresponds to a highly textured image - as expected. In between these two extremes are the multi-year ice (Figure 7b) and platelet ice (Figure 7c) images which have point cover values somewhere in the middle. Simulation data (Figure 8a and Figure 8b) with this algorithm yields very similar results to real data obtained from under the ice in Antarctica (Figure 8c and Figure 8d).

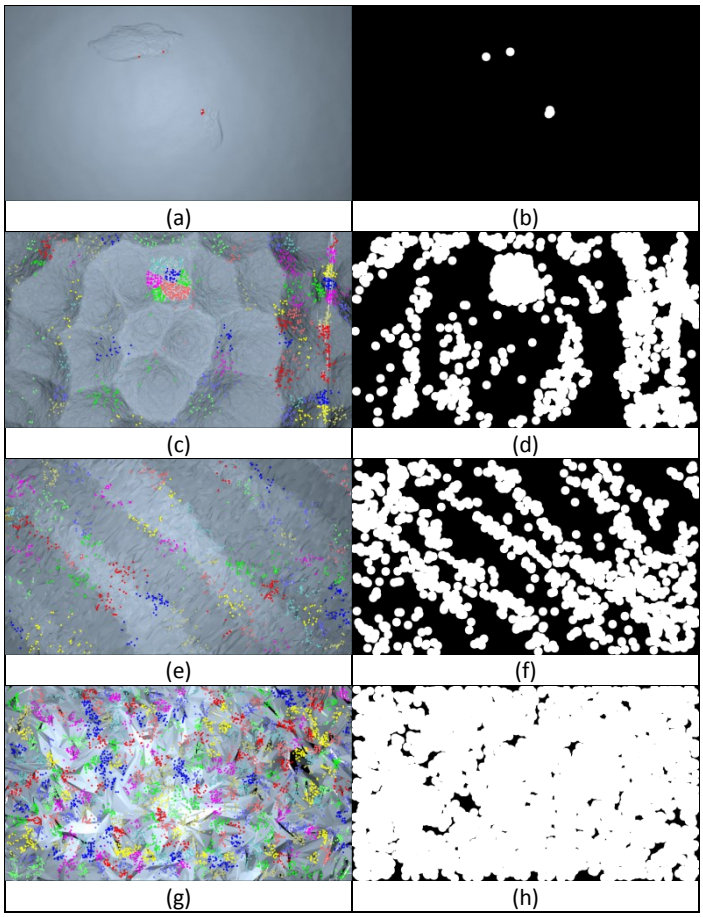


Fig. 7. Results from texture estimation algorithm with the simulated under-ice dataset showing each ice type (a-b: first-year, c-d: multi-year, e-f: platelet, g-h: frazil). On the left are input images with point feature detections as dots with color determined by k-means grouping. On the right are visual representations of the point cover mask used to calculate estimated texture.

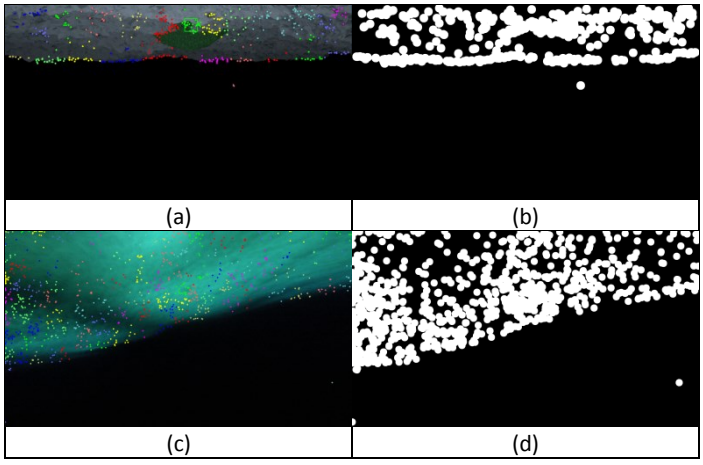


Fig. 8. Results from the texture estimation algorithm with the simulated under-ice dataset (a-b) compared to real-world results (c-d). On the left are the input images with point feature detections as dots with color determined by k-means grouping. On the right is a visual representation of the point cover mask used to calculate estimated texture.

The resultant texture estimation map of the simulated data over the vehicle trajectory is shown in Figure 9 where white represents more textured areas and black represents no texture. The quantitative value for estimated texture can be visualized in the point cover images (the right side of Figure 7) as the percentage of non-zero valued pixels (white pixels here). It can be seen from this output map that the expected texture is obtained with the algorithm for each type of ice: platelet ice is on the left hand side, the top left corner, and the bottom left corner and shows a medium texture value (gray); multi-year ice is on the top to the right of the platelet ice and shows a much lower texture; frazil ice is on the top to the right of the multi-year ice and is the most textured (white); first-year ice is located from the end of the frazil ice clockwise until reaching the platelet ice on the bottom left corner, and shows a very low value for texture (black), as expected. The texture estimation results can also be represented versus time (instead of x, y position) over the dataset duration (Figure 9), which does not require an estimate of the vehicle location. From this representation it can be seen that the global maximum is located around frame 120, which corresponds to the frazil ice, while the frames corresponding to lake ice (frames 150-350) are at the global minimum - as expected. Local maximums (i.e. frames 240, 410) mostly correspond to frames with features (cracks, animals) in the ice. There is a single frame of incorrect estimation (frame 217) in the first-year data resulting from many false feature detections. In areas where there are sharp transitions between ice types (i.e. between frames 120 and 160), the estimated texture value makes a linear translation, as would be expected. The texture estimation algorithm estimates texture values consistent with the corresponding ice types for 93.33% of the upward-all-ice simulated dataset. The bad frame estimates in this case include frames with ice deformations that are not detected due to their smooth transitions (which is acceptable), as well as the single frame of false feature detections in the first-year ice. The algorithm is successfully able to produce higher texture estimates from the more evenly distributed point sets. This can be seen between the image of simulated platelet ice only in Figure 7e (232181 pixel cover / 469 SURF points = 495 pixels / point spread) and the image of simulated first-year ice with features in Figure 11b (42664 pixel cover / 132 SURF points = 323 pixels / point spread). Here the spread of each point-feature is much greater for the evenly distributed points in the platelet ice, as expected, due to the limited overlap between point covers.

Fig. 9. Texture estimation using point features. The vehicle trajectory is plotted along with the estimated texture at each point. Here the texture measurement is represented as a scale from white (most textured) to black (least textured) with a gray scale in between. It can be seen that the most textured points in this case are located at the frazil ice while the least textured points are located at the first-year ice, as expected. Ground truth can be seen in Figure 6.

The proposed texture estimation algorithm was also evaluated using other simulated data with similar results, as well as on real under-ice imagery obtained under the ice near McMurdo, Antarctica. Results from this testing is shown in Table 1. In this case, the algorithm uses a combination of GFTT and SURF point-features for more robust results and better comparison across datasets. It can be seen from this table that the four types of ice simulated (first-year, multi-year, platelet, and frazil) are correctly ranked by texture value as expected. It can also be seen that in the Icefin Dive Jetty dataset, the frame with nothing but blue water has an estimated texture of only 42935 pixels while any image with ice present has a much higher value (at least 53256 pixels). In the image where an ice hole and ice crack feature are visible, the texture estimate is almost double (81014 pixels) that of the empty frame. The GoPro Jetty dataset shows consistent texture estimation across all images except those with Icefin in the frame, as expected. This proposed algorithm shows positive results in attempting to estimate ice texture throughout an under-ice video dataset.

TABLE I. TEXTURE ESTIMATION RESULTS

Platform	Image Contents	Estimated Texture Value	
		Pixels	*Percent*
Simulated	First-year Ice	2572	0.179
	Voronoi Ice	249030	48.038
	Platelet Ice	700515	48.647
	Frazil Ice	1361540	94.551
	First-year ice with features	119972	8.331
	Simulated Front Cam, platelet and green ice	240612	16.709
SCINI	Multi-year ice, Anemones	145782	12.840
GoPro	Jetty Ice, Platelet ice and green ice	516625	24.914
	Jetty Ice, Platelet/green ice, Icefin	47966	2.313
	Platelet ice and green ice	531677	25.640
	Platelet ice	508455	24.996
	Platelet ice and green ice	518316	24.996
	Platelet ice, green ice, Icefin	201945	9.739
Icefin Front	Jetty Ice, Nothing – just blue	42935	12.423
	Jetty Ice, Large ice crack	56476	16.341
	Jetty Ice, Ice crack and ice hole	81014	23.442
	Jetty Ice, Ice up close - textured	53256	15.410
	Jetty Ice, Ice crack	72662	21.025

B. Texture-based Anomaly Detection Results

The texture-based anomaly mapping algorithm using point feature methods was first tested on the simulated all ice dataset described above, given the inherent ground truth benefits. This dataset represents a large simulated under-ice environment comprised of multiple sections of different types of ice (platelet, multi-year, frazil, and first-year) as well as anomalies in the ice (cracks, deformations, and animals of multiple colors seen under the ice in Antarctica). The location of the camera throughout the trajectory is known, so the locations of detected objects can be used to create a map using the camera field of view and the altimetry (distance to the ice). The results from this algorithm over the simulated all ice dataset can be seen in the form of the mapped positions of detected ice anomalies (seen as blue blobs) in Figure 10. The ground truth map can be seen in Figure 6.

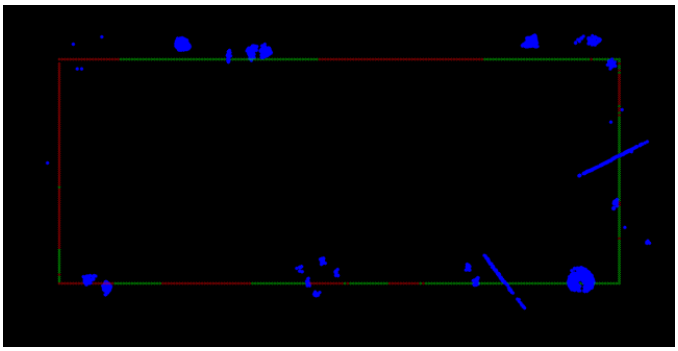


Fig. 10. Resultant map from the point-feature-based under-ice anomaly detection algorithm. Here 18 of 24 anomalies were detected (six misses) with only six false frame detections over the 600 frame dataset.

The five yellow anemones (bottom center), green ice (bottom right, bottom left, top middle), and anemones (top mid-left) were detected successfully as well as cracks (long straight features in lower right quadrant) and ice deformations (right side) that cannot be detected using color-based methods. Overall, 18 of the 24 anomalies were successfully detected (six misses) with only six false frame detections over the 600 frame dataset. An example anemone detection frame can be seen in Figure 11a while Figure 11b shows the detection of both colorful features (yellow and purple anemones) as well as an ice crack feature. Local maximums in the estimated texture plot (Figure 9) (i.e. frame 240 – crack in the ice; frame 410 – five yellow anemones) mostly correspond to frames with anomalies in the ice, and can be used to indicate frames of interest in post-analysis.

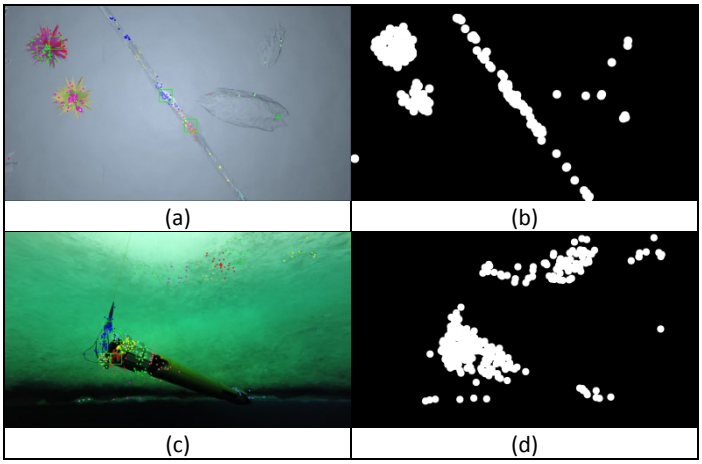


Fig. 11. Results from the texture estimation algorithm (right) and feature-based anomaly detection algorithm (left). It can be seen that in both the simulated (a) and real (c) datasets, anomalies are successfully detected (green boxes).

This algorithm was also tested on other simulation and real-world data with similarly promising results, as summarized in Table 2. Examples of successful detections, successful empty frames, and missed detections are presented in this table across multiple datasets. In the dataset obtained by the Icefin vehicle under the sea ice in Antarctica, there are many false anomaly detections of the vehicle's structure (seen in Figure 12), where there are anomaly detections despite a mostly blue frame. However, these false detections can be easily ignored as the

location of the vehicle structure in the image is constant and can be masked out. True detections using the algorithm can also be seen including specs in front of the camera, ice cracks (Figure 12c) and holes in the ice. In the image from the SCINI [11] vehicle under the McMurdo Ice Shelf in Antarctica, multiple anemones are successfully detected (Figure 15a). In the GoPro dataset taken under the sea ice in Antarctica, the only texture anomaly feature present was the Icefin vehicle itself, which is successfully detected in Figure 15b. Even images from this dataset with only the small diameter tether in view yield clustered point groupings around the tether, indicating an object that is difficult to see with human eyes, but more easily found in post-processed images. It is interesting to point out that while the patches of green ice are considered ice anomalies here, they are not detected with the texture-based algorithm as they are the same texture as the other ice surrounding them. This indicates the benefit of a fusion of color- and texture-based detection methods. The images from this GoPro dataset that do not have anomalies present yield no detections despite the large number of point features, as the features are evenly distributed and not sufficiently dense to indicate an anomaly. This is the desired behavior of the algorithm in such cases, as it is able to ignore highly textured images, which cannot be accomplished with a simple point-feature count for anomaly detection. The detection failures with the simulated dataset result from similarity between textures of the anomalies and the ice background, which presents another motivation for a fused texture- and hue-based detection algorithm. Most of the detection failures encountered in the Icefin camera dataset result from anomalies that are too small to accumulate a sufficient grouping of point features.

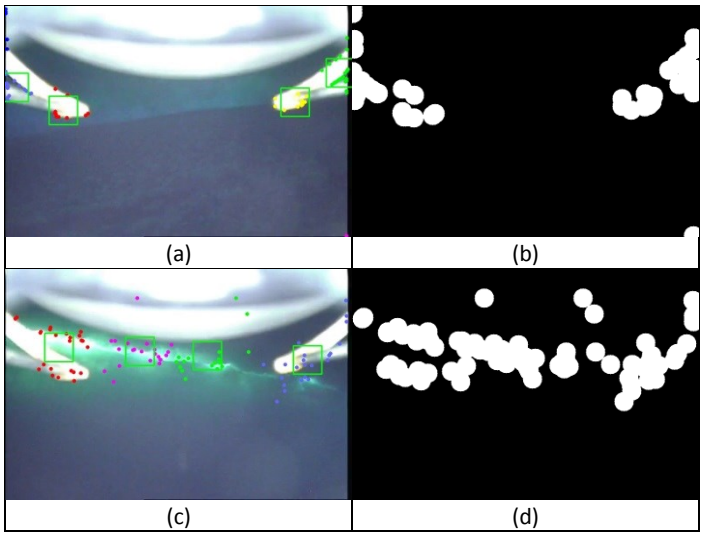


Fig. 12. Results from the texture estimation (right) and feature-based anomaly detection (left) algorithms. It can be seen that only the vehicle body is detected in an empty frame (a) while an ice crack features is detected in (c).

TABLE II. TEXTURE-BASED ANOMALY DETECTION RESULTS

	Image Contents	Qualitative Results
Sim	Platelet ice	Success – no detections
	Platelet ice and green ice	Success –green ice
	Platelet ice and 5 sponges	Fail – misses sponges
	FY ice, anemone group	Success - anemones
	FY ice, ice features, sponges	Success - crack, sponges
	FY ice, green ice, shrimp	Success - shrimp, green ice
	FY ice, ice crack, sea slug	Success - slug, crack
	Frazil ice	Success – no detections
	MY ice, crack, anemones	Fail – misses anemones
SCINI	MY ice and anemones	Success - detects anemones
GoPro in Jetty	Platelet ice, green ice	Success – no detections
	Platelet ice, green ice, Icefin	Success - detects Icefin
	Platelet ice, green ice, tether	Success - no detections
	Platelet ice	Success - no detections
	Icefin, green ice background	Success - detects Icefin
Icefin in Jetty	Blue frame	Success – no detections
	Jetty platelet ice	Success – no detections
	Platelet ice, small ice crack	Fail -misses small ice crack
	Platelet ice, ice crack	Success - detects ice crack
	Platelet ice, large ice hole	Success - detects ice hole
	Platelet ice, spec feature	Success - detects spec
	Platelet ice, small ice hole	Fail – misses small ice hole
	Platelet ice, ice crack	Success - detects ice crack
	Platelet ice, ice crack/hole	Fail - misses ice crack
	Platelet ice, large ice hole	Success - detects ice hole
	Platelet ice, ice hole	Success - detects ice hole

C. *Hue-based Anomaly Detection Results*

The proposed hue-based algorithms were also evaluated using the same simulated and real-world under-ice datasets entailed above for the texture-based algorithms. The simulated all ice under-ice dataset was again used for primary validation of the algorithm, as ground truth was known with the highest accuracy. The output of the blue-thresholding and component-grouping algorithm is shown in Figure 13 and Figure 15. In Figure 13a, the main detection in the image is of the green ice, along with other small false detections at the edge of the ice (which are filtered out based on their small size). In Figure 13c, the yellow anemone is detected completely while the purple anemone has multiple fragmented detections due to the close proximity of the color purple to the blue hue band. However, both anomalies are successfully detected while the ice features of blue color are not - as expected from a hue-based algorithm. This provides an argument for a texture and color fusion method for anomaly detection, as these ice features are detected using the texture-based method. The results of the hue-based anomaly detection algorithm can be represented as an output map with highlighted objects of interest given an estimate of the vehicle position over time as well as the distance to the ice and the camera field of view. The simulation datasets contain ground truth for these values and the estimated output map for the upward-all-ice simulated dataset can be seen in Figure 14 where the trajectory is plotted in green (anomaly detected) and red (nothing detected) and objects of interest are marked with blue blobs. In this case, the green ice and the anemone group anomalies are successfully detected and mapped. It can be seen from comparing the output maps (Figure 14) to the ground truth map (Figure 6) that all 20 non-blue anomalies are detected with zero misses for both local maximum and blue thresholding methods. One false detection was found with the local maximum method while ten false detections (eight of which were in the frazil ice) were found

with the blue thresholding method. Over the 600 frames of the simulated dataset the local maximum method was 88.5 percent accurate in detecting if there was a colorful anomaly in the frame. Specifically, 27 frames had misses and 42 frames had false detections.

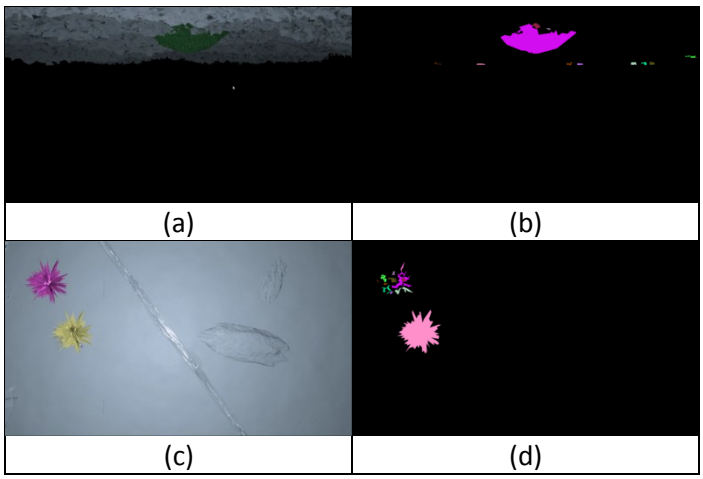
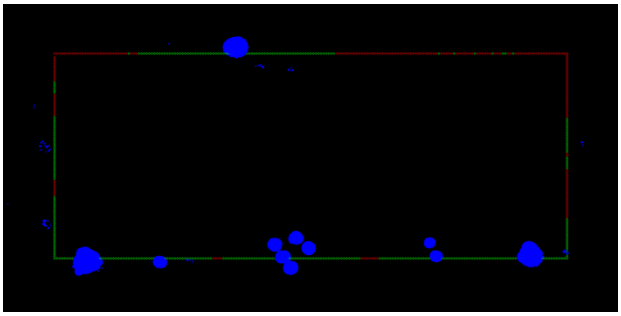
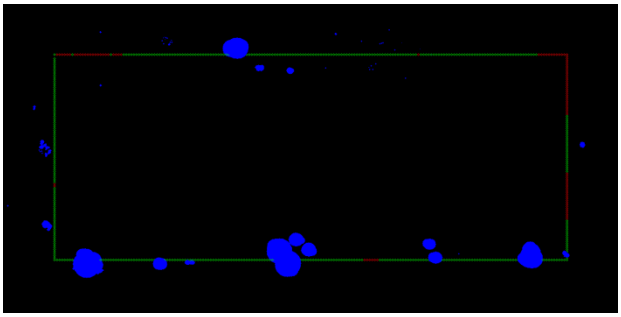


Fig. 13. Results from the blue thresholding hue-based anomaly detection algorithm with simulated under-ice data. Both green ice anomalies (a-b) and yellow/purple sponge anomalies (c-d) are successfully detected.



(a)



(b)

Fig. 14. Output map result from the hue-based anomaly detection algorithm where the blue blobs represent detected anomaly candidates (Left: Local maximum method, Right: Blue threshold method). Using the local maximum method, 100% of the 20 non-ice anomalies were detected with one false detection. Using the blue thresholding method, 100% of the 20 non-ice anomalies were detected with ten false detections (eight of which are in the frazil ice).

While the simulated datasets provided good ground truth for evaluating the accuracy of the algorithm, further insight was gained with the evaluation of the algorithm on real-world data obtained from beneath the ice near McMurdo station. An image from the SCINI dataset [11] of anemones present against the ice shelf resulted in the detection of six of the nine anemones in the foreground (Figure 15a-b). An image containing the Icefin vehicle and green ice as anomalies from the GoPro Jetty dataset shows correct detections of both the Icefin vehicle and the patch of green ice (Figure 15c-d). An image from the Icefin Jetty dataset with an ice hole anomaly is shown in Figure 15e-f, where the hole itself was not detected, but instead the green ice around it. In this image, the false detection of the Icefin vehicle frame can be seen. This can be masked or ignored from processing as the location is known and constant over time.

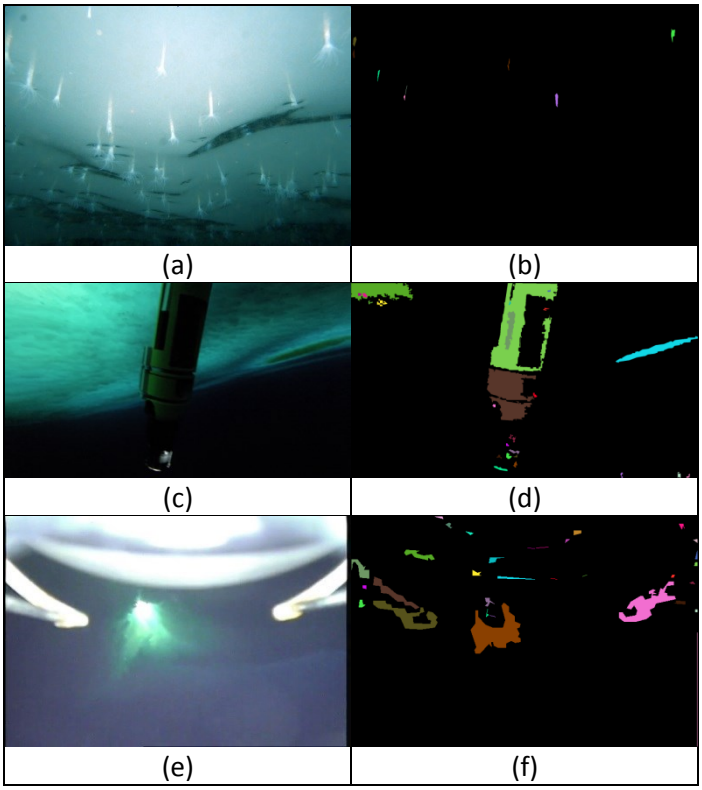


Fig. 15. Results from the blue thresholding hue-based anomaly detection algorithm with real-world under-ice datasets. Anemones (a-b), the Icefin vehicle (c-d), green ice (c-d), and a hole in the ice (e-f) are all successfully detected. False detections of the Icefin vehicle body can also be seen (f), but can be filtered out in practice as the locations are known and constant.

D. Hue-based color estimation results

In addition to the hue-based filtering and clustering method for anomaly detection in a frame, a quantitative value for percent "interesting" color (i.e. non-ice) is obtained for each frame. When these values are mapped over the vehicle trajectory, local maximums can be used during post processing to indicate candidate frames of interest to examine for anomalies. This value of non-blue hue percentage can give insight into the color contents of the video frames, and therefore indicate areas of interest over the vehicle's trajectory. This proposed algorithm was evaluated using the all ice simulated dataset (with ground truth), and a plot of the resultant hue values are shown in Figure 16. The maximum between frames 50 and 100 corresponds to the green ice in the

top-middle of Figure 6. The maximum between frames 280 and 330 corresponds to the green ice in the bottom right of Figure 6. The smaller maximum between frames 330 and 360 corresponds to the yellow and purple sponges found in the bottom middle-right of Figure 6. The maximum between frames 400 and 420 corresponds to the group of five yellow sponges in the bottom middle of Figure 6. The lower maximum between frames 440 and 460 corresponds to a single purple sponge. The maximum between frames 460-520 represents the green ice at the bottom left of the map in Figure 6. Most of the frames with color values around zero percent represent those with only ice in the frame across the various ice types (Figure 7). Using an estimate of vehicle location, a map of the vehicle trajectory along with estimated color can be obtained (Figure 16), where brighter values represent more non-ice color and darker values represent a smaller percentage of non-ice color present in a frame. It can be seen from this map that the algorithm outputs high color values for areas of large non-blue groupings, such as the green ice and five yellow sponges, while it has low values for ice-only areas or areas of ice with only white and black features. The use of a quantitative non-blue percentage value for each frame in a dataset yields an accurate estimate for 95% of the simulated all ice dataset, where almost all of the 30 bad frames occur from small false detections the frazil ice.

Fig. 16. Output map from the hue-based color estimation algorithm. The vehicle trajectory is plotted along with the estimated amount of non-ice color at each point. Here the color measurement is represented as a scale from white (most color) to black (least color) with a gray scale in between. It can be seen that the highest valued locations in this case are located around the large green ice patches and groups of colorful features while the lowest valued points are located in locations with no colorful features (black, white, and blue), as expected. Ground truth can be seen in Figure 6.

In addition to testing with simulated data, this hue-based color estimation algorithm was tested with real-world under-ice datasets obtained in Antarctica. A summary of the results are shown in Table 3. In the simulation dataset it can be seen that images of just platelet ice yield a value of zero (as expected) while images with green ice and other colorful anomalies yield values between seven and nine percent (corresponding to the local maximums in Figure 16). The simulated forward-looking camera image has much lower relative value (less than two percent) because over half of the image is black, corresponding to the open water taking up most of the frame (which is ignored). In the Icefin Jetty dataset, a much smaller variation is observed in the values with a baseline of around three percent representing the amount of vehicle frame always present in the images. However, the platelet ice with an ice crack shows a local maximum at 0.4% above the baseline with only water in the image. One image in this dataset shows a box at the seafloor, which results in a color value of 82%. Such a high value can be expected as there is no ice in the frame and the box is not of a blue hue. The anemones in the image taken from the SCINI [11] vehicle in Antarctica represent 0.4% of the image. In the GoPro Jetty dataset, no vehicle frame is present and the baseline for the values in this dataset is around 0.375%. Various amounts of green ice across the images cause this value to increase to 0.4-4.1%. When the Icefin vehicle is present in the image in addition to the patch of green ice, this output value increases to 8.2%. In the frame containing the Icefin vehicle against a patch of green ice, this value increases drastically to 83%. This value representing percent of pixels outside of the blue hue range provides a good relative indication within a dataset of the likelihood that a frame has a color anomaly present.

TABLE III. HUE-ESTIMATION RESULTS

Platform	Image Contents	% Estimate of Non-ice Pixels
Simulation	Platelet Ice	0.000
	MY ice with green ice and features	7.028
	Platelet ice with yellow sponges	9.000
	Platelet/green ice and colorful features	6.949
	Platelet ice and green ice	1.933
Icefin in Jetty	Nothing – just blue	3.264
	Seafloor, Box feature	82.45
	Platelet ice	3.562
	Platelet ice and ice crack	3.682
SCINI	Multi-year ice and Anemones	0.410
GoPro in Jetty	Platelet ice and green ice	0.710
	Platelet ice, green ice and Icefin	8.192
	Platelet ice and green ice	0.844
	Platelet ice and small amount of green ice	0.411
	Platelet ice and green ice	2.367
	Platelet ice only	0.375
	Platelet ice and large amount of green ice	4.103
	Platelet ice, green ice, Icefin	83.426

After evaluation of both proposed anomaly detection methods with the all ice simulated dataset, results were gathered for evaluation of algorithm detection performance. The ground truth used for this evaluation was human annotation of each frame in the dataset. The results are shown Table 4 for both the hue-based and feature-based detection algorithms. As expected, the flat first-year and multi-year ice show the best results as they contain a smoother ice background. There are some missed detections and false detections of anomalies in the ice, but most of these scenarios occur at the abrupt boundaries between ice types (not common in real-world datasets). However, these "false" detections at sharp transitions between ice types are acceptable as they would represent areas of interest for post-analysis. The hue-based methods did not perform well with the frazil ice due to false positives stemming from small blobs of artificial colors introduced in the simulated data with such a jagged ice structure. These small false detections would be eliminated in a real-world dataset by raising the threshold for component size due to the incredibly high texture of frazil ice. The low detection score for the feature-based method with frazil ice is caused by the high texture of the ice which masks the texture

of the anomalies (mostly misses here). In the case of highly textured platelet and frazil ice, the hue-based algorithm should always perform with better results. In practice, the algorithm could automatically favor the hue-based method in a situation of high background texture detection using the point cover texture estimation detailed above. The main problem encountered with the hue-based algorithm was missed detections with the white and black features due to the fact that these hues are filtered out during pre-processing. The main issue with the feature-based algorithm was both false detections and misses due to similarity in textures between anomalies and background ice. This leads to the desire for a fusion algorithm that uses both hue and texture to more robustly find anomalies, which is out of the scope of this work.

TABLE IV. ANOMALY DETECTION RESULTS (SIMULATED DATASET)

Ice Type	**Correct Detection Percentage**		
	Feature-based (SURF)	***Hue-based***	***Hue-based (blue-thresh)***
First-year	93.89	98.33	93.89
Multi-year	95.83	100	100
Platelet	57.21	80.18	81.53
Frazil	100	0	0

VI. CONCLUSION

Post processing analysis of under-ice video datasets can be incredibly tedious. The four algorithms presented here aid in this process by providing automatic estimates of the texture, non-ice color percentage, and anomaly candidates in each video frame. The texture estimates proposed here provide a quantitative estimate for the number and spread of the detected point features in a frame, corresponding to the texture of the background ice. Any dense groupings of these extracted features are proposed to indicate anomaly candidates for post-processing by a human analyst. Hue, or color, is also used to estimate the percentage of pixels in the image that do not correspond to ice background – yielding a good relative estimate of how "interesting" the colors in a frame are. Any pixel groupings outside of the blue hues corresponding to the ice that are found from this hue analysis provide candidates for ice anomalies in post processing. Each of these algorithms show good results across both simulated and real under-ice datasets and provide useful tools in the post-processing of these large datasets. Suggested future work in this area includes a fusion of these anomaly detection algorithms as well as further analysis of these algorithms on additional datasets.

ACKNOWLEDGMENT

The authors would like to acknowledge NASA and NSF for providing the means for Antarctic data collection as well as the Georgia Institute of Technology for providing the means for algorithm development and evaluation.

REFERENCES

[1] J. Zhang, M. Marszałek, S. Lazebnik, and C. Schmid. "Local features and kernels for classification of texture and object categories: A comprehensive study." International journal of computer vision 73, no. 2, pp. 213-238, 2007.

[2] A. Kim, Ayoung and R. Eustice. "Real-time visual SLAM for autonomous underwater hull inspection using visual saliency." Robotics, IEEE Transactions on 29, no. 3, pp. 719-733, 2013.

[3] J. Sattar and G. Dudek. "On the performance of color tracking algorithms for underwater robots under varying lighting and visibility." IEEE ICRA, pp. 3550-3555, 2006.

[4] W. Skarbek, A. Koschan, T. Bericht, and Z. Veroffentlichung. "Colour image segmentation-a survey," (1994).

[5] A. Spears, M. Meister, M. West, B. Schmidt, T. Collins, A. Howard, "Design and devlopment of an under-ice AUV for use in polar regions," IEEE OCEANS, 2014.

[6] A. Spears, M. West, T. Collins, and A. Howard, "Evaluation of Sonar and Video Data Collection Efforts in an Under-Ice Environment Using an Unmanned Underwater Vehicle," in IEEE SMC 2014, pp. 1–6, 2014.

[7] VideoRay LLC, "Videoray Pro 4 Manual," 2015

[8] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in Computer Vision–ECCV, pp. 404–417, 2006.

[9] J. Shi and C. Tomasi, "Good features to track," IEEE CVPR, pp. 593–600, 1994.

[10] Arthur, David, and Sergei Vassilvitskii. "k-means++: The advantages of careful seeding." ACM-SIAM symposium on Discrete algorithms, pp. 1027-1035. Society for Industrial and Applied Mathematics, 2007.

[11] F. Cazenave, R. Zook, D. Carroll, M. Flagg, and S. Kim, "Development of the ROVSCINI and deployment in McMurdo sound, Antarctica.," Journal of Ocean Technology,vol. 6, no. 3, 2011.

[12] Blender Foundation, http://www.blender.org/