PROJECT REPORT

Vision Analysis of Pack Ice for Potential Use in a Hazard Warning and Avoidance System

by

Oscar De Silva,

Harsh Environment Technology Center (HETC),

Memorial University of Newfoundland,

Canada.

Date: 25th May, 2016

Project Supervisors: Claude Daley

John Dolny

George Mann

Ray Gosine

Dennis Peters

Table of Contents

Introduction

Ships travelling through pack ice are exposed to collisions that can result in structural damage to the hull. The GEM project at Memorial University has developed ice-ship interaction simulation software that allows study of the impact forces applied on a ship when it maneuvers through pack ice [1]. Such capability is useful in order to predict the collisions that would potentially affect the structural integrity and operational performance of ships and floating offshore structures. GEM is capable of simulating transit through complex pack ice formations at a rate much faster than real time [1]. If hyper-real time simulation were available in a real operational setting, with actual ice, it would permit a variety of benefits, including general operational planning. If the near field ice information were sufficiently accurate, GEM could also be used in a "feed forward" near-field hazard warning and avoidance system (HWAS).

Currently the GEM software generates a random field of pack ice with specific size information for the analysis. However, in real-world scenarios the actual size, position, and distribution information about a pack ice field would be needed. At the moment it is unsure where such accurate information would come from. To bridge this gap and implement a hazard warning system, a ship requires a system to detect and model the near-field pack ice. Such a sensory system should be able to detect and model the ice field adequately so that the HWAS has sufficient time horizon to execute a safe manoeuvre through a given pack ice field.

This project investigates the feasibility of a pack ice HWAS based on a vision system. I.e. a camera and a machine vision computer attached onboard the ship. The vision system should be able to estimate a pack ice field using the information received from a camera and it should be able to perform this task at a sufficient speed to meet the update rates required for safe maneuvering through a pack ice field.

Project Objective

Develop a computer vision system in a laboratory setting which can estimate size, relative position, and distribution of a set of ice polygons by taking the camera images within a given time duration and to communicate this information to the GEM software.

Technical Background

Object detection and 3D reconstruction using camera images are two areas studied under machine vision research [2-3]. This project requires an object detection method in order to extract the polygonal shape of each ice floe seen in an image. Then this polygonal shape should be reconstructed in metric coordinates using calibration parameters related to the camera.

Image based object detection can be performed using two main methods: 1) image segmentation and blob analysis, 2) feature based object detection [2]. Feature based approaches are preferred when the object carries local uniquely identifiable textures. Since ice floes are uniform and have a planar appearance, the applicability of feature based matching (using SURF or SIFT features) will be of limited use.

Image segmentation and blob analysis methods attempt to identify regions of an image (identified as blobs). Then parameters of each blob (size, color, axis) are analysed in order to identify an object. This approach would easily allow to segment an image to produce regions related to pack ice and identify polygons related to each ice floe. Popular image segmentation methods include Otsu's thresholding [1], color threshold methods [1], watershed algorithm [4], graph cut algorithm [5], snake algorithm [6], video segmentation algorithms [7] etc. However, only a subset of these algorithms allow modest speeds required for real time application, while others are intended for post processing (offline) applications.

3D reconstruction of objects using images is a well defined geometric problem [3]. The reconstruction process uses a pin hole camera model to mathematically describe a camera. The geometric constraints related to a problem and the mathematical model of the camera can be exploited to solve for the 3D points corresponding to a set of pixel values recorded on a 2D image. Additionally, the reconstruction can be improved by incorporating sequence of images [4] and the dynamics of the moving platform (in this case the ship).

Many of the popular machine vision methods discussed here are already available as MatLab [8] and C libraries [9] for rapid development and evaluation of algorithms. The project evaluates different approaches related to object detection and 3D reconstruction in order to design an optimized solution which meets the performance and speed requirements of a pack ice HWAS. Additionally, the project proceeds to validate these methods using images from laboratory experiments, real images taken onboard ships and simulated video streams from the GEM simulation software [1].

Project Methodology

The project is carried out in 5 distinct steps. The first three steps develop the vision system using MatLab for ease of algorithm development. Once finalized the design is deployed as a C++ application to achieve real time performance. The five different activities carried out in this project are as follows:

1. **Develop a 3D reconstruction method:** A numerical analysis will be performed to design and evaluate a 3D reconstruction algorithm. The 3D reconstruction method should be able to predict the shape and size of each floe using shapes registered on an image.
2. **Develop experimental test-bed:** Next, a real dataset will be generated for experimental validation of the algorithm. The data would allow to asses the polygon reconstruction accuracy and scaling properties of the developed algorithm.
3. **Develop image processing pipeline**: Next, the object detection method will be improved such that it is capable of handling actual images captured onboard ships at a modest frequency.
4. **Implementation for real time use:** Under Activity 4 the MatLab based vision system will be deployed using OpenCV libraries to achieve real-time execution speeds.
5. **Establish communication with GEM:** The developed Qt application establishes a link with the GEM software in order to communicate between the two processes. This link allows to send polygon data to and from GEM, and also to acquire ship ice interaction force predictions from GEM.

Project Timeline

	Activity	**Sept**	**Oct**	**Nov**	**Dec**	**Jan**	**Feb**	**Mar**	**Apr**
1	Develop 3D reconstruction method								
2	Develop experimental testbed								
3	Develop image processing pipeline								
4	Implementation for real time use								
5	Establish communication with GEM								
	Project Documentation								

Work completed Work in progress Future work

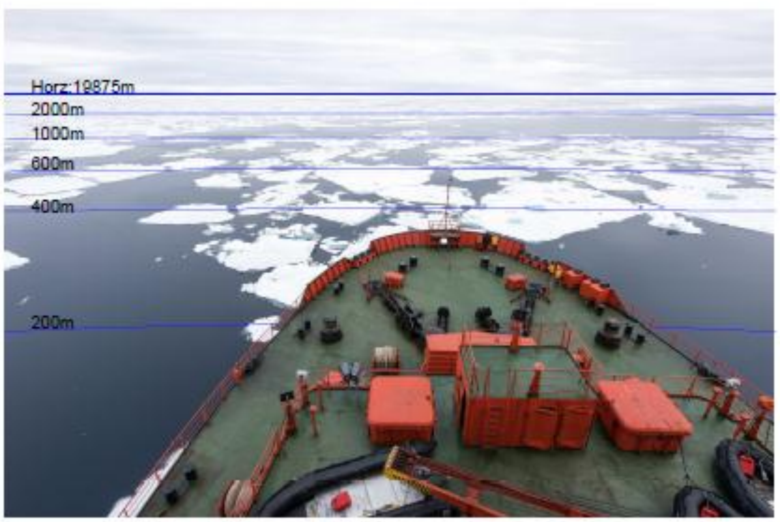
Project schedule

Completed work

Activity 1: Develop a 3D reconstruction method

As part of Activity 1, first, a realistic mathematical model for a vision system was identified. Second, a 3D reconstruction procedure was developed. Third, the error statistics and sensitivity of the reconstruction procedure were evaluated.

a) Vision system model: The test image shown in Figure 1 was used to identify a mathematical model for the vision system with realistic parameters. All the necessary image information, camera calibration information, and GPS tags related to the image were available embedded in the EXIF data of the image. The horizon line on the image was used to identify the orientation (pitch) of the camera and the GPS tags provided the height information of the camera's location.



Author	Christopher Prentiss Michel
Resolution	7360 x 4912
Calibration Data	Available from EXIF data of image
Location Data	Available from GPS tags

Figure 1- Test image used for the mathematical model

b) 3D reconstruction procedure: The 3D reconstruction process is performed using a ray tracing method where a ray of light is projected on to the sea surface through each vertex of a polygon seen on an image. The reconstruction method requires to know the height of the camera above sea level and the camera parameters (focal length, lens distortion). These can be established using a camera calibration procedure or can be extracted from images (E.g. Figure 1).

Next a numerical analysis of the reconstruction method was performed using a synthetic polygon field. A uniform synthetic field was generated which consisted an identical ice floe every 100m on the sea surface (shown in Figure 2(a)). These polygons are projected to an image plane (shown on Figure 2(b)) modelling the pixel noise and discretization effects which occur during image capture. The image is then used to 3D reconstruct the polygons. As seen in Figure 2(c), the 3D reconstruction capability deteriorates as the distance approaches 2km. I.e., the gap between polygons on the image is close to a pixel when the distance approaches 2km.

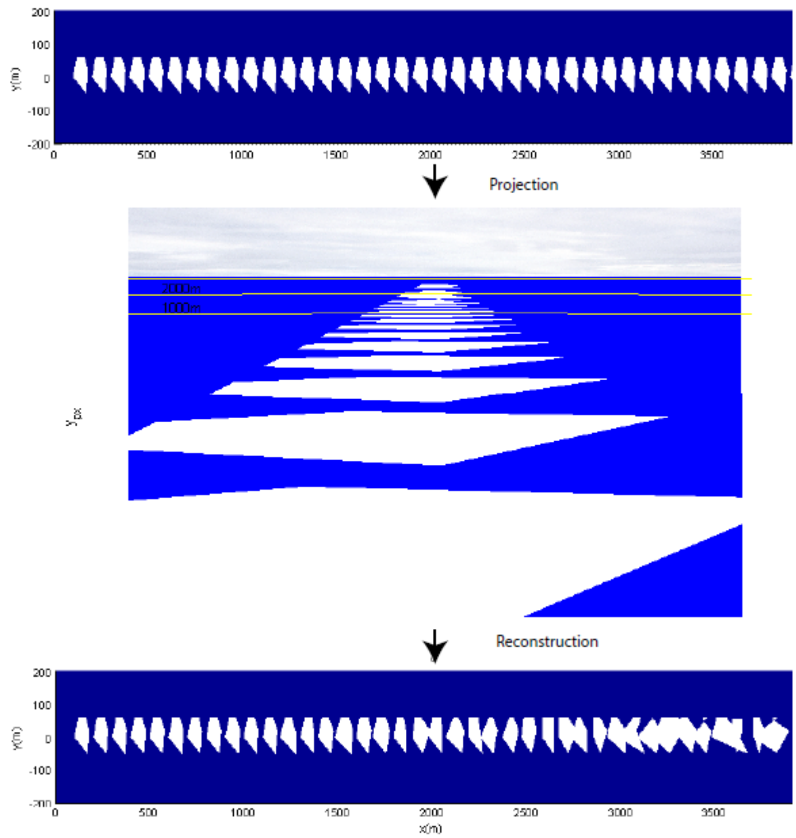


Figure 2- (a) Top: Polygon field, (b) Middle: Projection, (c) Bottom: Reconstruction using the image

The error analysis indicates an exponential degrade of per pixel distance resolution as an object moves further away from the ship. Figure 3 illustrates that a range of 2km would deteriorate the distance resolution (distance increment along sea surface per pixel) as much as 10m/px. This analysis additionally considered the earth curvature effects when generating the distance resolution results. However, it was found that errors due to earth curvature effects are negligible for a near field reconstruction.

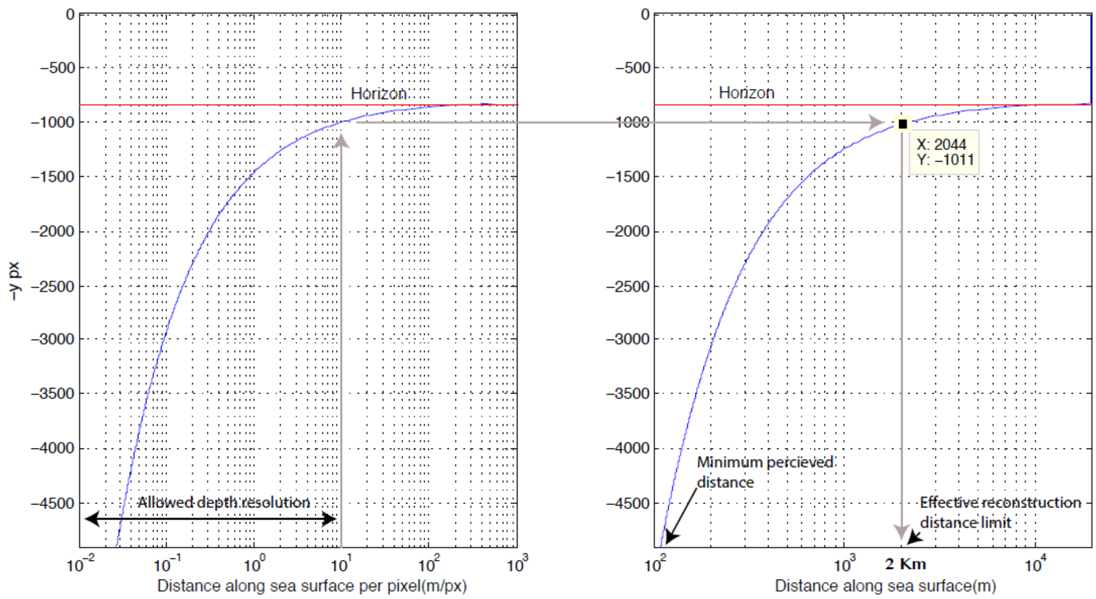


Figure 3 – Distance per pixel variation along y axis of image

Error analysis – reconstruction accuracy: The polygon field was used to generate statistical results related to vertex position reconstruction errors, polygon size reconstruction errors, and angle reconstruction errors. Figure 4 illustrates the reconstruction error statistics related vertex position and area for different distances along the sea surface. The study also considered the sensitivity of these results for the following factors: freeboard height, camera height, and camera orientation.

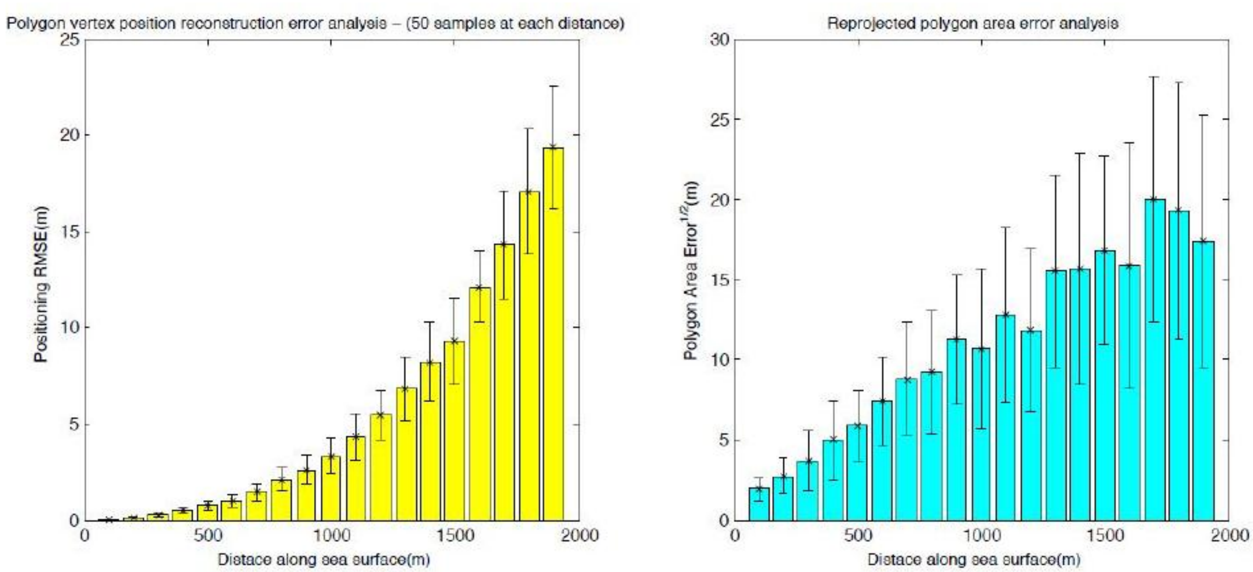


Figure 4- (a) Left: Vertex position reconstruction error (b) Right: Area reconstruction error

Completion of Activity 1 allowed to establish a mathematical model for the vision system and allowed to develop a 3D reconstruction method. It additionally identified the main factors affecting the accuracy of the reconstruction process. Furthermore, the analysis identified a set of performance indicators (position, area, and angle reconstruction accuracy) to use for validation purposes throughout the rest of the project.

Activity 2: Develop an experimental test bed

The above analysis is primarily a numerical study. In order to perform the reconstruction error analysis under an actual setting, a polygon field with known sizes and locations should be captured. Hence a scaled down laboratory setup was developed to facilitate the vision study.

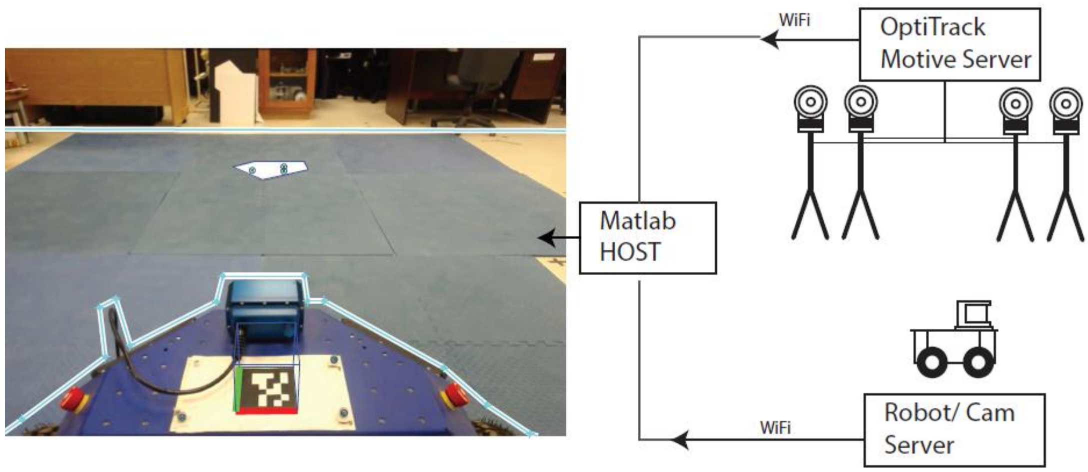


Figure 5- The experimental testbed

The test bed: Figure 5 illustrates the laboratory scale test bed developed for experimental purposes of this project. The test bed includes a mobile robot equipped with a high resolution camera facing a synthetic polygon field. The sizes of the polygon field are known and an external motion capture system (Opti-Track) allows to find the actual location of the polygons with respect to the camera. The procedure allows experimental validation of the reconstruction method using the know size data and location data related to the synthetic pack ice field. All marker data from the Opti-track system, camera images, and control commands of the robot are communicated to and from a remote computer using a Wifi link.

System calibration: Calibration of the test bed includes the following tasks which were performed using the listed methods:

Table 1 Approach used to perform different calibration tasks

Calibration task	**Method of calibration**
Identify camera parameters	Matlab camera calibration toolbox
Identify location and orientation of camera w.r.t. the ship (robot)	A checker board calibration pattern attached to the robot. (This should be modified to a method which uses known feature locations of a ship)
Identify camera to motion capture system transformation	Opti-Track marker alignment function (An optimization approach solves for the best alignment)

Scaling: A scaling analysis was performed to identify how the errors relate between the model scale and actual ship scale. The analysis suggests that, when scaling from experiment scale to ship scale by linearly increasing the camera altitude and distance of the polygon away from the camera by some scale factor ‘x’, the error associated with the polygon area and position shall also be scaled by the same factor. The error associated with polygon angles however, need not to be scaled. Figure 6 shows the agreement between the polygon angle errors at experiment scale and polygon angle errors at ship scale. This analysis was performed using the mathematical model of the vision system developed under Activity 1.

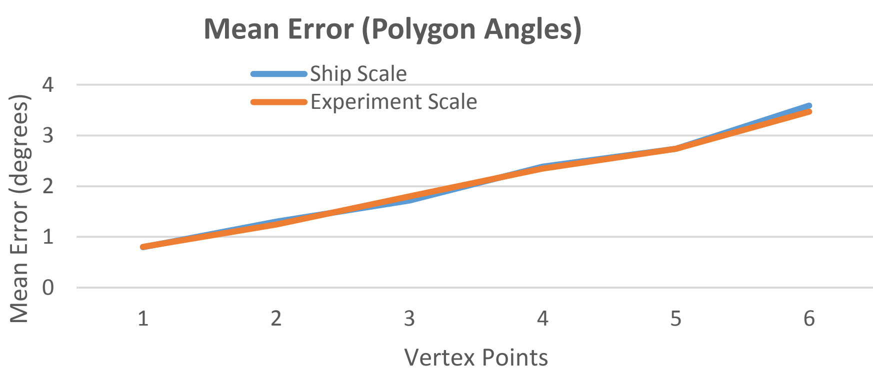


Figure 6- Model scale verification

Activity 3: Develop the image processing pipeline

In order to perform the reconstruction methods shown under Activity 1, it is necessary to identify a set of polygons related to the pack ice field seen in an image, and to generate vertices related to these polygons. Activity 3 identifies the necessary steps that one should follow to generate the vertex list of polygons seen on an image. The sequence of operations are illustrated in Figure 7 and the methods used under each step are detailed in Table 2.

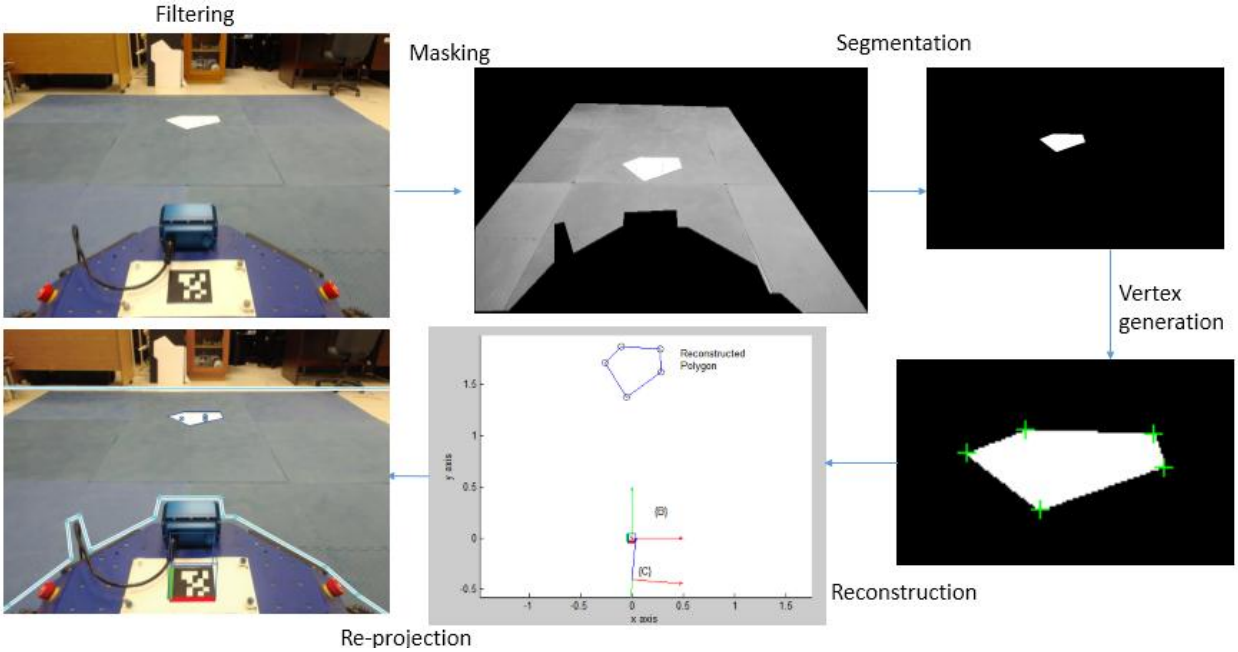


Figure 7- Image processing pipeline

Image processing operations:

Table 2-Steps used to identify polygons in an image

1	**Image intensity corrections**	This process adjusts the image to account for different lightning conditions of the capture. Method: Histogram equalization
2	**Masking**	Masks out the ship and the sky of the image. A standalone application was created which can be used to create the masking image using the watershed algorithm.
3	**Segmentation**	This process divides the image in to multiple discrete areas considering the intensity of pixels. Method: Otsu`s thresholding algorithm +de-noising, followed by a morphological erosion operation. The resulting blobs are used as labels to apply the watershed algorithm to segment the image.
4	**Vertex generation**	This process determines the best fit polygon for a given segment. Method: Douglous-Pueker algorithm.

5	**3D reconstruction**	Using a ray trace method, which applies a closed form solution to the geometric optimization problem (Activity 1).
6	**Re-projection**	Using mathematical model of the camera(Activity 1).

Activity 4: Implementation for real time use

All of the above algorithms and methods were developed in Matlab. As part of activity 4, these algorithms were programmed in C++ to achieve fast execution and to test the expected performance/speed during final deployment. A QT based application (Pack ice Vision App) was developed to facilitate further tuning and visualization of results. Image processing operations were performed using open source computer vision (OpenCV) libraries [9] for fast and hardware accelerated execution. Figure 8 illustrates the application developed for testing the performance of the vision system.

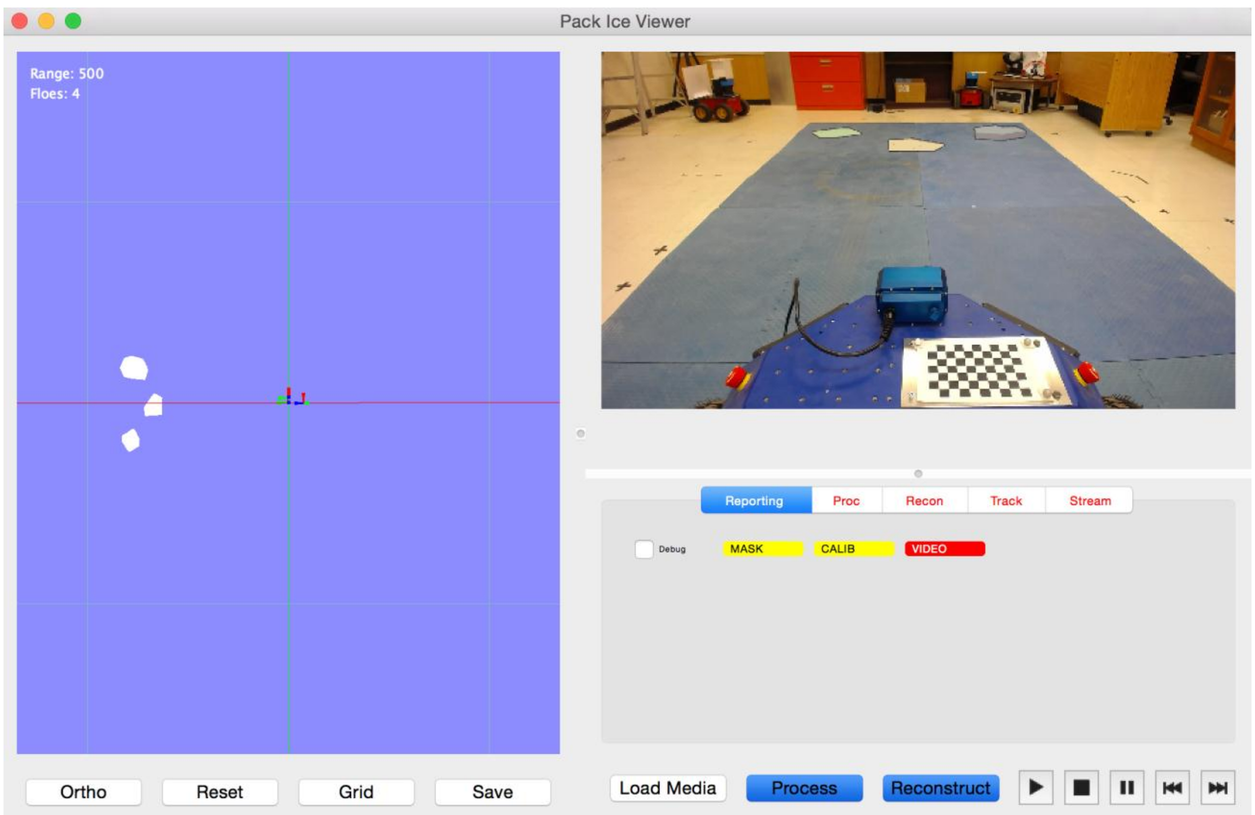


Figure 8- Qt Application developed for the project

Performance of the system was validated using 3 different data sources. 1) Images captured from a deck of a ship. 2) Experimental dataset (images and video) captured as part of Activity 2, 3) Simulated video streamed from the GEM software. The performance of the vision system for these different images/videos are presented under the results section of the report.

Activity 5: Establish communication with the GEM software

As part of Activity 5, the "Pack ice Vision Application" was modified to communicate with the GEM ship-ice interaction simulation software. This link was established using a server client architecture using the Qt Socket module available with Qt libraries. The GEM software was modified such that it opens a server during application start-up. The pack ice vision system application developed under this project initiates a client which connects with GEM at start-up.

The server and the client are programmed with following functionality to support this project.

Table 3 Server client custom commands

	Functionality	Client request	Server response
1	**Request camera parameters from GEM**	CALIB REQUEST	Sever sends a data structure having camera parameters
2	**Request a single image of current simulation from GEM**	FRAME REQUEST	Sever sends a JPG compressed image of the scene
3	**Request video stream of current simulation from GEM**	Client loops the FRAME REQUEST	Sever sends a JPG compressed image at each iteration
4	**Request polygon field from GEM**	POLYGON PULL REQUEST	Server updates the polygon field of the current simulation tab
5	**Send polygon field to GEM for simulation**	POLYGON PUSH REQUEST	Server sends a data structure of the polygon field seen in current simulation tab
6	**Request to initiate new simulation**	SIM REQUEST	Server initiates a new simulation tab using the polygons and executes simulation. Once complete the server sends the expected collision forces to client

The first four functionalities outlined in Table 4 were used during this project for validation of the pack ice vision system. The video streaming capability was used to allow the pack ice vision application to use the image stream coming from GEM as its input image source for ice floe detection and reconstruction. The server requests outlined in Table 4 can be controlled using the UI panel available under the stream tab of the application as shown in Figure 9. The results of the system generated using this server client connection are outlined under the results section of the report.

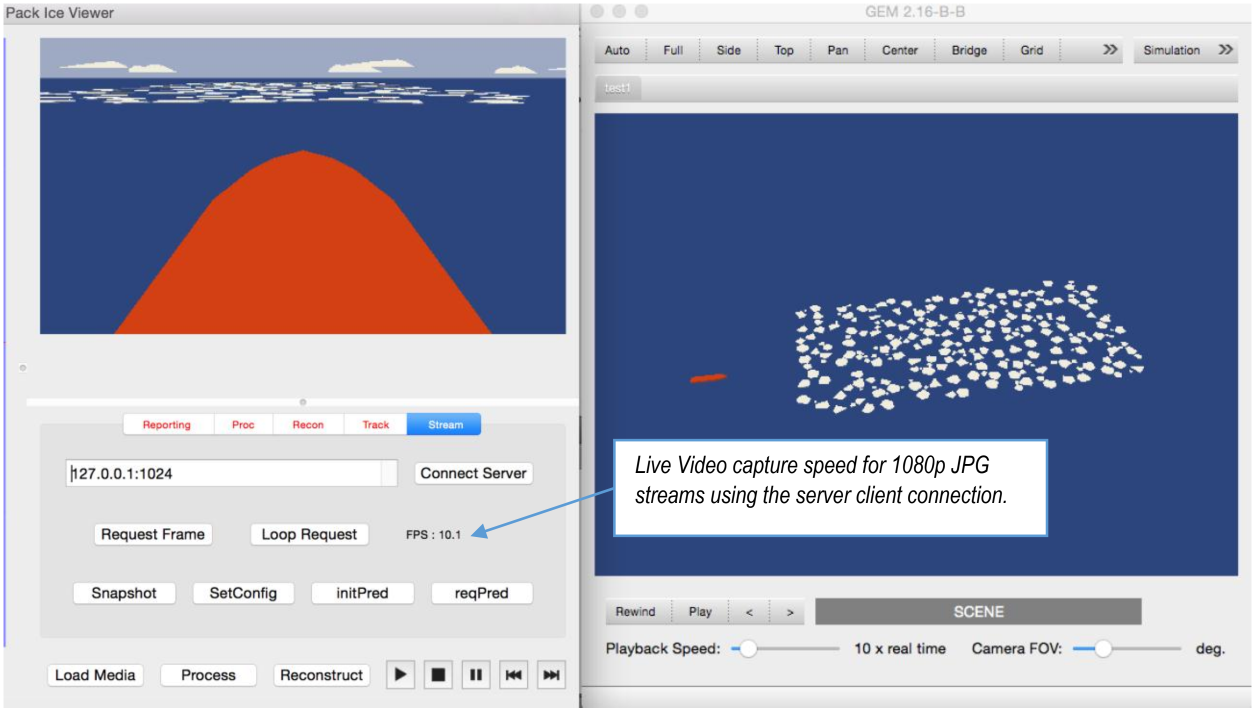


Figure 9- GEM sending a JPG video stream to the "Pack Ice Vision Application" at a rate of 10 FPS

The final 3 functionalities outlined in Table 4 were not tested during the scope of this project. However, these server requests are essential for the next phase of the project which will attempt to use GEM based prediction to maneuver the ship through a pack ice field with minimal ship ice interaction.

Results

This section summarizes the performance of the "Pack ice vision application" developed as part of the project.

Figures 10-15 illustrate few of the evaluated sample images along with the speed of execution for each case. The first set of images (Figures 10-13) shown below illustrate the performance for segmentation only. This is because the images did not have any calibration data to perform 3D reconstruction.

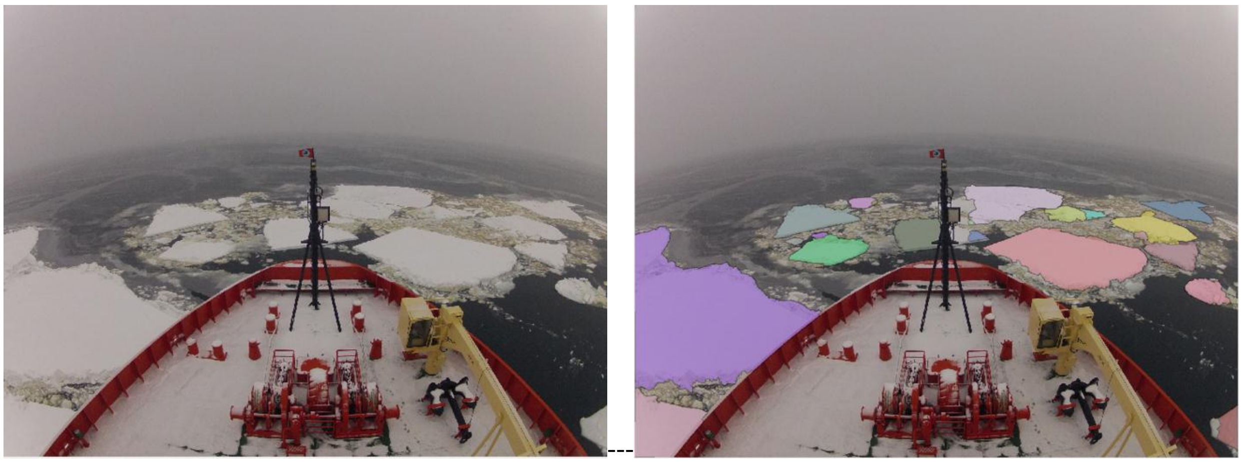


Figure 10 – Segmentation Image 1 (1920x1440 Segmentation time: 221 ms)

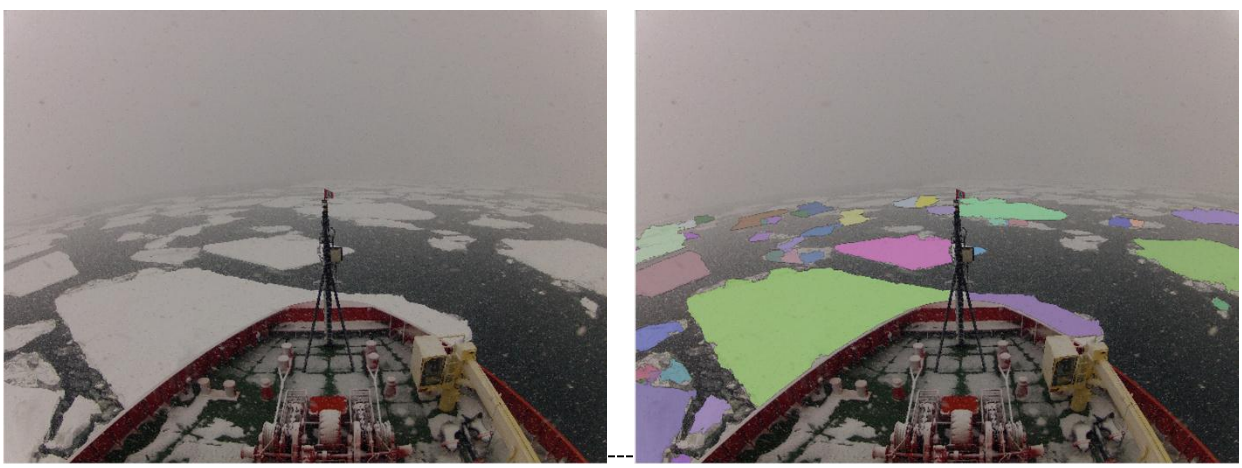


Figure 11 – Segmentation Image 2 (1920x1440 Segmentation time: 972 ms)

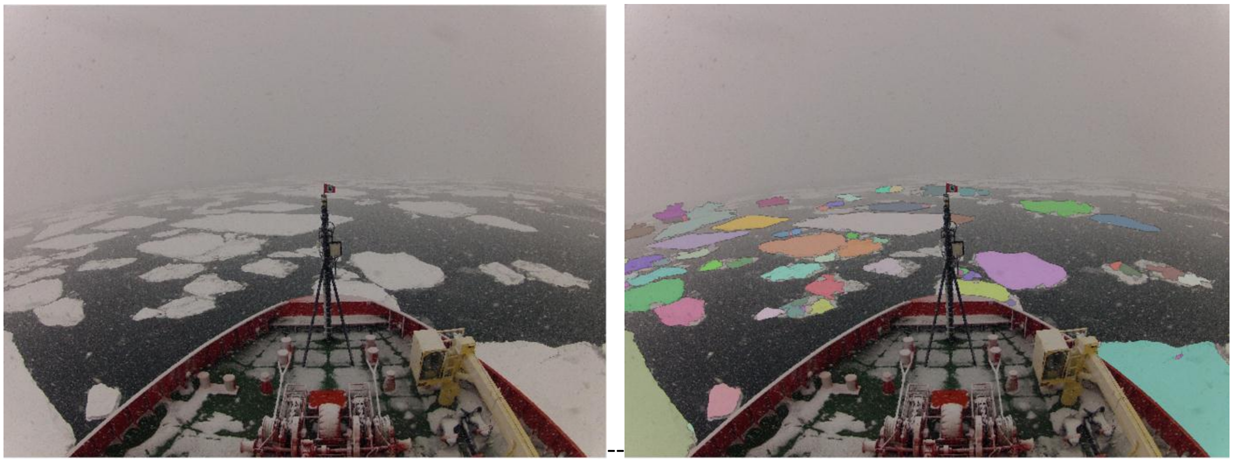


Figure 12 – Segmentation Image 3 (1920x1440 Segmentation time: 920 ms)

The next set of figures shown below illustrate the performance for both segmentation and reconstruction.

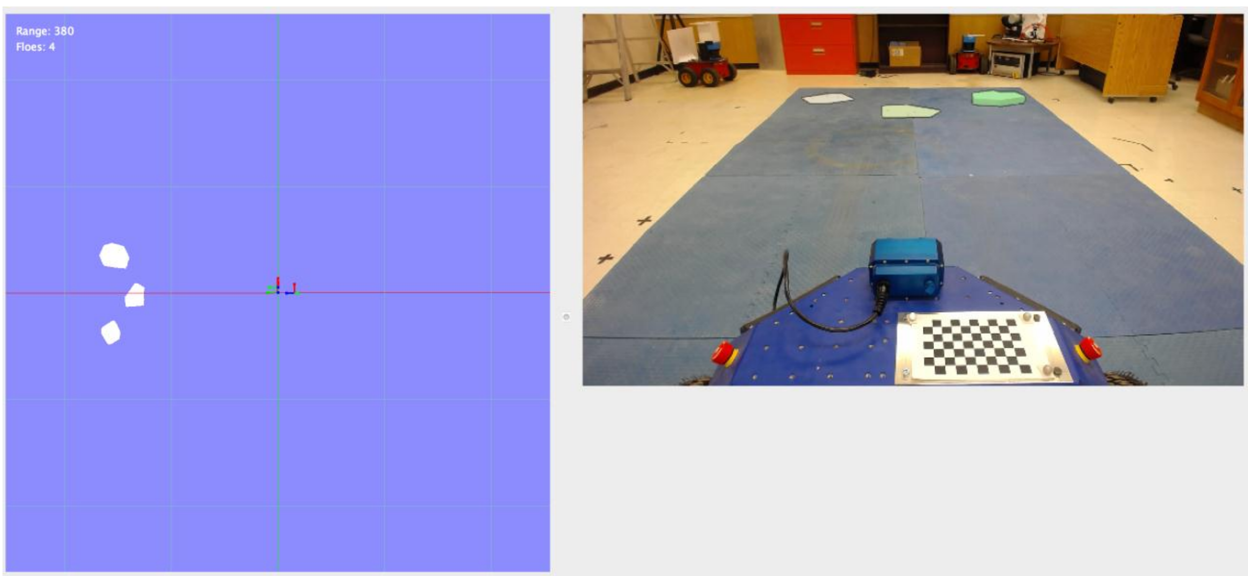


Figure 13 – Segmentation and reconstruction Image 1 (1920 × 1080 Segmentation time: 118 ms , Reconstruction time: 2 ms)

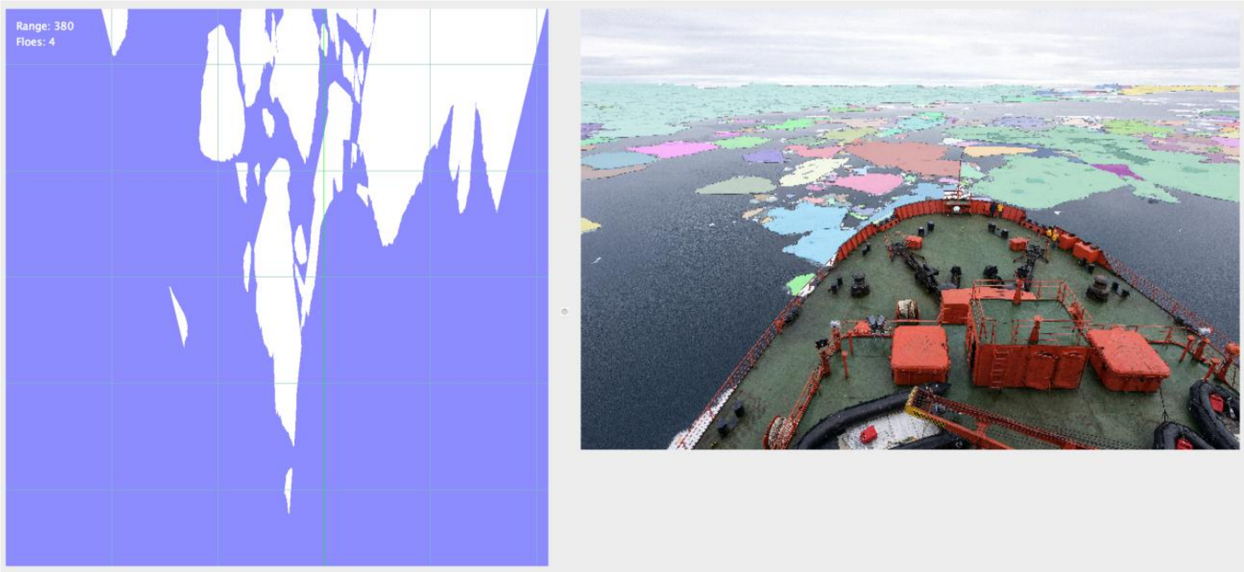


Figure 14 – Segmentation and reconstruction Image 2 (7360 × 4912 Segmentation time: 4585 ms , Reconstruction time: 228 ms)

These results validate the performance of the image processing pipe line for actual field data. As the resolution and the number of ice floes increase, the time taken to perform the detection and reconstruction steps increase proportionally. The software is capable of performing the image analysis and reconstruction task at a rate of 3 frames per second (FPS) for 1080p video streams from GEM (Figures 15) and at a rate of 1 FPS for actual images of pack ice fields (Figure 11, 12). For a generic case with 1080p video it is observed that the worst case processing time expected from the system is 1 second. While this can be further improved through optimizing the code, 1 second update rate for the system is deemed sufficient considering the operational speeds of ocean vessels.

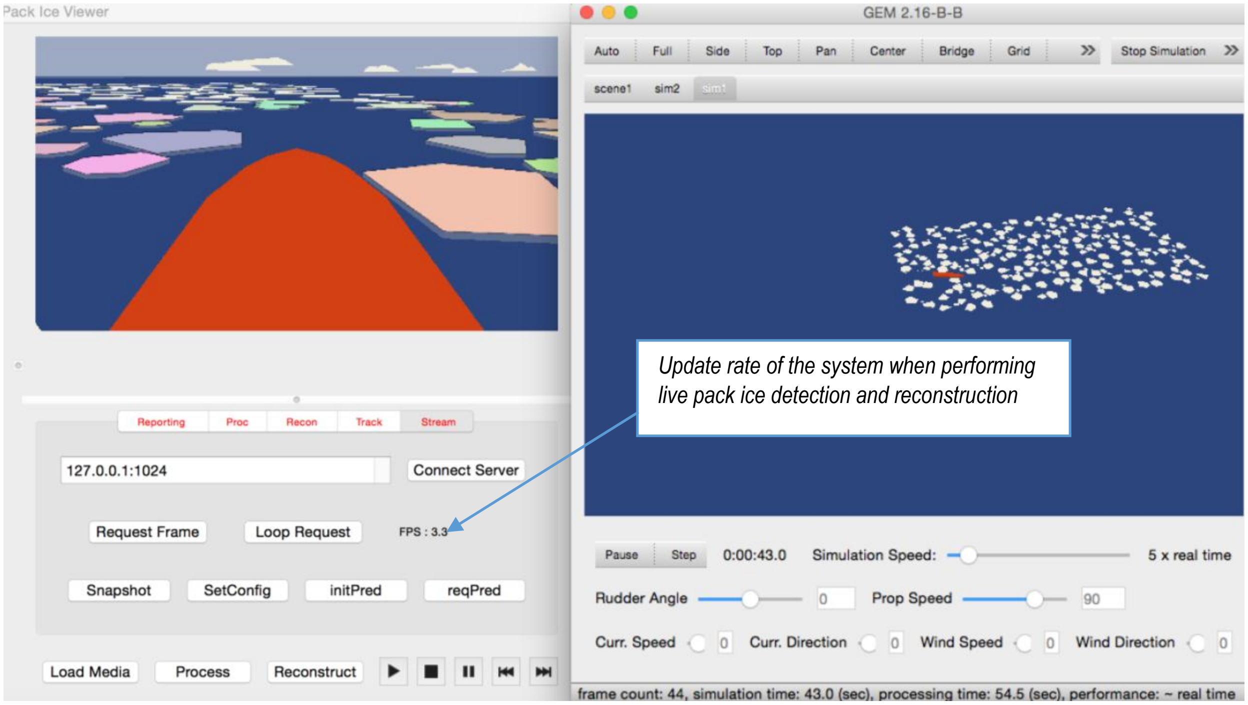


Figure 15 – Segmentation and reconstruction of GEM video stream (1920x1080). The average frame rate that the vision system can execute is 3 FPS.

The pack ice detection system is capable of effectively segmenting images to identify ice floes and reconstruct the field at close proximity. Testing of the system for different datasets revealed three aspects of the current system that should be further studied to realize a more robust system.

- A learning system which can identify and discard incorrect identifications of ice floes.
- A tracking system which can keep track of the shape, size and a unique identifier for each ice floe in a video stream.
- Code and algorithm optimizations for faster performance of the system.

These are expected to be pursued in the second phase of the pack ice vision study project.

Conclusion

This project developed the key system components necessary to implement a vision based hazard avoidance system for ships travelling through pack ice. First the project developed an object detection method to detect ice floes in an image, then a 3D reconstruction method was developed which can reconstruct an ice floe field in front of a ship. The project deployed the vision system using C++ code to test the speed of execution and performance using actual images captured from a deck, and using experimental data captured in the lab. The project concluded by establishing a software connection with the GEM ship ice interaction simulation software. The current version of the developed system is capable of processing images or a video stream of (1920x1080) resolution with a worst case performance of 1 FPS. The software is also capable of sending the reconstructed ice field to GEM to predict ship ice interaction forces expected during transit.

This work developed a version of the vision based hazard avoidance system for initial validation purposes. From this validation work it is evident that a vision based system is capable of implementing at a modest speed applicable for ocean going vessels. Both the performance and features of the vision system is expected to be further improved during the second phase of this project.

Future work

As future work the following system components are required to be improved for robust performance of the pack ice vision application.

- Improvements to the image processing pipeline for faster performance.
- Improvements to the image processing pipeline for better detection of floes.
- Improvements to the reconstruction method by incorporating a tracking algorithm.
- Experiments for further validation of the developed system specifically to study the accuracy of force prediction of the vision system aided GEM prediction.

References

[1] Alawneh, S., Dragt, R., Peters, D., Member, S., Daley, C., & Bruneau, S. (2015). Hyper-Real-Time Ice Simulation and Modeling Using GPGPU. *IEEE Transactions On Computing*, *64*(12), 3475–3487.

[2] Gonzalez, R. C., & Woods, R. E. (2006). *Digital Image Processing* (3rd ed.). Prentice-Hall.

[3] Hartley, R., & Zisserman, A. (2003). *Multiple View Geometry in Computer Vision*. Cambridge University Press.

[4] Yang, X., Li, H., & Zhou, X. (2006). Nuclei Segmentation Using Marker-Controlled Watershed , Tracking Using Mean-Shift , and Kalman Filter in Time-Lapse Microscopy. *IEEE Transactions on Circuits and Systems*, *53*(11), 2405–2414.

[5] Freedman, D., & Zhang, T. (2005). Interactive Graph Cut Based Segmentation With Shape Priors. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*.

[6] Dagher, I., & Tom, K. El. (2008). WaterBalloons : A hybrid watershed Balloon Snake segmentation. *Image and Vision Computing*, *26*, 905–912.

[7] Grundmann, M., Han, M., & Essa, I. (2010). Efficient Hierarchical Graph-Based Video Segmentation. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on* (pp. 2141–2148).

[8] Mathworks. (2016). Image Processing Toolbox: User Guide (R2016a). Retrieved May 26, 2016, from http://www.mathworks.com/help/images/index.html

[9] OpenCV dev team. (2014). The OpenCV Reference Manual. Retrieved May 26, 2016, from http://docs.opencv.org/2.4.13/doc/user_guide/user_guide.html