ISSN 2464-4617 (print) WSCG 2016 - 24th Conference on Computer Graphics, Visualization and Computer Vision 2016
ISSN 2464-4625 (CD-ROM)

A new approach to turbid water surface identification for autonomous navigation

Mateus Eugênio Colet, Adriana Braun, Isabel H. Manssour
PUCRS, Faculdade de Informática
Porto Alegre, RS, Brazil
mateus.colet@acad.pucrs.br, adriana.braun@gmail.com, isabel.manssour@pucrs.br

ABSTRACT

Navigation of autonomous vehicles in natural environments based on image processing is certainly a complex problem due to the dynamic characteristics of aquatic surfaces, such as brightness and color saturation. This paper presents a new approach to identify turbid water surfaces based on their optical properties, aiming to allow automatic navigation of autonomous vehicles regarding inspection, mitigation and management of aquatic natural disasters. More specifically, computer vision techniques were employed in conjunction to artificial neural networks (ANNs), in order to build a classifier designed to generate a navigation map that is interpreted by a state machine for decision making. To do so, a study on the use of different features based on color and texture of such turbid surfaces was conducted. In order to compress the extracted information, Principal Component Analysis (PCA) was performed and its results were used as inputs to ANN. The whole developed approach was embedded in an aquatic vehicle, and results and assessments were validated in real environments and different scenarios.

Keywords

Computer Vision, Surface Vehicle, Principal Component Analysis, Artificial Neural Network.

1 INTRODUCTION

With recent technological advances, several areas of knowledge have been benefited from techniques of digital image processing and computer vision. The area of robotics, mainly, stands out by the wide use of computer vision, in order to acquire necessary knowledge for agents from the universe around them. In addition to the use of sensors, computer vision can provide more information to increase and analyse the amount of data that can be supplied [IMM09a]. Navigation in natural environments based on image processing is certainly a complex problem. The main difficulties are the dynamic characteristics that aquatic surfaces can present, due to variation of image features such as brightness and color saturation. Physical factors such as light intensity, shadows, reflections, diffraction and refraction effects also influence the identification process for navigation [IMM09a].

International organizations related to risk reduction show statistics stating that the impact of floods affects over 500 million people, with a cost of $ 50 million annually, and it accounts for the highest number of deaths registered in natural disasters [Kro15a]. The effects of these disasters are even more drastic in developing countries, due to lack of early warning systems, flood control and emergency response infrastructure [SKV11a]. Considering this context, we are interested in developing an approach to assist in the navigation of an autonomous vehicle in a post-disaster environment, both for gathering data and identifying its real dimension. Some solutions for the navigation of surface vehicles using computer vision have already been developed. However, there are still some open problems, as the availability of a method to navigate in turbid water surfaces in adverse environments, which could run in a hardware with computational limitations and could be adapted to several autonomous vehicles.

The main goal of this paper is to present an approach for automatic identification of navigable turbid water surfaces, based on computer vision techniques. We focus on the key subproblem of automatically segmenting turbid aquatic surfaces for autonomous navigation. The result of this process is the generation of a navigation map to guide the direction to be taken by the autonomous vehicle. Seeking to improve accuracy, two ANNs were trained: the first one to recognize turbid water regions without reflection and the second one to identify those regions with reflection [GSW07a, ASN11a]. These ANNs, as well as all the algorithms of this approach, run independently in an embedded hardware. A navigation map is built and, then, a finite state machine guides

the direction to be taken by the autonomous vehicle, which can be a boat, a navigable platform or a smaller device.

The main contributions of the presented approach are:

- Proposal of a method to estimate navigable turbid water surfaces from images captured by a monocular camera positioned on an aquatic vehicle;
- Generation of a navigation map that determines the limits of navigable regions and that can be interpreted by other algorithms for navigation;
- Presentation and development of an algorithm for decision making related to autonomous navigation, based on the generated navigation map;
- Embedding the developed approach in an aquatic vehicle.

The remainder of this paper is organized as follows. Section 2 presents some related works. We briefly describe the developed approach in Section 3. In Section 4 we present some experiments and results. Finally, the last section presents closing comments and future works.

2 RELATED WORK

Some solutions of locomotion for surface vehicles using computer vision have already been developed [SMB12a, SMH04a, GSW07a, HS11a, ASN11a, RMB11a]. A detailed description of consequences, influences and variations in variable values, as well as the challenges that effect the ability to detect water surfaces by optical means, is presented by Iqbal et al. [IMM09a]. They focus on difficulties involved in the detection of water bodies, along with state of the art techniques that deal with this topic. Andrew et al. [CAN11a] also emphasize that aquatic environments present several problems, such as the reflection of other objects on the water surface, currents and waves that distort the aspect of water, or the presence of debris or sediment, which changes the color of water or causes movement on its surface.

According to Huntsberger et al. [HAH11a] and Yao et al. [YXL07a], a vehicle equipped with a water detector based on computer vision has a higher probability to navigate safely and efficiently. This is particularly emphasized when the vehicle is in an unknown environment. The image acquisition in this case can be done by a set of cameras [SMH04a] or just a monocular camera [SMB12a, CAN11a, YRC11a].

For example, Santana et.al. [SMB12a] propose a model for water detection with segmentation guided by dynamic texture recognition. From an input video, they defined that the region of water has a signature, based on the measure of entropy over the trajectories obtained from optical flow trackers. In order to classify regions with a higher degree of reliability in surfaces of little ripple, a segmentation method based on appearance is applied. Then, every image is labeled in segments with water if they cover a certain percentage of pixels classified as water by the method based on entropy and texture. It presented a good true positive rate; however, this model does not adapt to mobile cameras due to its constant movement. According to the authors, tracking stabilization techniques would help in reducing the inertial optical flow induced by camera moving.

Rankin and Matthies [RM10a] proposed the detection of water bodies and ponds through the behavior modeling of these surfaces. They used intensity data based on the variation of color spaces RGB and HSB to estimate the contribution of the reflection coefficient, considering the reflection surface and a combination of other factors such as saturation and brightness. According to the authors, the developed method for detecting water bodies in open areas proved to be sensitive to any reflection, both vegetation and objects on the aquatic surface. One way to deal with possible exceptions could make this method more robust and less limited.

Other works [GSW07a, ASN11a, HS11a] use a robust descriptor with the analysis and combination of features to build a classifier with supervised learning. According to the authors, the use of a set of training data allows to build a good classifier to distinguish water surfaces, since natural environments suffer variations, as physical factors.

Gong et al. [GSW07a] present a two-stage algorithm to find the margin between water and land. Images are collected and classified into two types: Reflection-identifiable and Reflection-unidentifiable. After, the images are segmented into smaller regions based on their color and uniformity, which are classified into areas of land or water according to features such as symmetry and brightness. Then, the algorithm traces a border to separate water from land regions by means of a classifier using an adaptive threshold segmentation. Frames with 320 x 240 pixels with reflection-identifiable processing take 2 seconds to be processed, and frames with reflection-unidentifiable take 27 seconds. Besides being a computationally expensive algorithm, it also seems hard to be implemented in autonomous video capture application.

Achar et al. [ASN11a] propose a self-supervised method to segment images into "sky", "river" and "shore" regions. It uses assumptions about river scene structure to learn about appearance models based on features as color, texture and image location, considering the horizon line to automatically specify the correlation among features. It extracts features of color spaces RGB, Lab and HSV individually and in various

combinations to train the classifier. Thus, it allows to label each part of the image with the probability of being water. Each labeled region is used to train a support vector machine (SVM) model generating the output for each image segment. This method presents good results, but each frame of 640 *x* 360 pixels takes around 2.32 seconds to be processed in a high-performance computer. Thus, it is difficult to embed it in medium and small vehicles.

Considering the methods described in several works [RM10a, ASN11a, IMM09a], we have adopted the use of several features (see Section 3.2) for the development of our approach. This is because the aquatic surface not only changes its optical property such as saturation and brightness, but it is also not uniform, causing color variation. Some techniques are robust to distinguish, segment and identify aquatic surfaces based on color analysis, and by using several color spaces, it states that a color descriptor associated with a vector of features becomes robust using statistical measurements to form classifiers [RM10a, ASN11a, HS11a, GSW07a].

3 APPROACH DESCRIPTION

This section presents the proposed approach for the automatic identification of navigable turbid water surfaces and automatic navigation of aquatic vehicles. It starts with an overview of the developed methodology, followed by an explanation about each implemented step.

3.1 Methodology Overview

The developed approach has several steps, as presented in Figure 1. Initially, sequences of images are collected by a monocular camera coupled to the prototype of the autonomous aquatic vehicle shown in Figure 11-(b). Then, the first step consists in the subdivision of each input frame I into blocks of $r \times s$ pixels. The values of r and s should be set to ensure good computational performance and classification granularity. In our experiments, we set $r = s = 10$ pixels, since our input frames have 320×240 pixels. Thereafter, for each block B, a set of 32 colors and texture features is extracted (see Section 3.2). We standardized these features and changed the coordinates of z-scores, by projecting them into the subspace of k principal components obtained through Principal Component Analysis (PCA) for the training phase described in Section 3.3, hence reducing data dimensionality. The values obtained are submitted to the classifiers, modeled as multilayer perceptron Artificial Neural Networks (ANN). As output for the ANNs, each image block B is classified as a "navigable" or "non navigable" region, independently. This procedure allows us to classify each block in different threads, which increases computational performance. Once all image blocks have been classified, we built a navigability map for each frame. This map is then submitted to a Finite State Machine (FSM), that interprets it and defines the actions to be performed by the vehicle. The following sections describe this methodology.

3.2 Extraction of image features

Each image block B is processed individually as follows: Firstly, we convert it to HSV and YUV color spaces, keeping the original RGB block image; afterwards, it is split into 8 color channels (red, green, blue, hue, saturation, value, luminance and chrominance). Then, we compute a series of statistics for each one, as described below.

Initially, we computed the normalized histogram of intensities for each channel c. Hereafter, these histograms will be denoted as H_c, with $c \in$ { *Red, Green, Blue, Hue, Saturation, Value, Y(luminance), U(chrominance)* }. The value of element h_{ci} from the histogram H_c is given by:

$$h_{ci} = \frac{n_i}{n}, \tag{1}$$

where $i \in [0, M]$, n_i is the number of pixels with intensity i in each channel c of a given image block, $n = r \times s$ and M is the maximum intensity value of the color channel, i.e., $M = 255$ considering a color depth of 8 bits per pixel.

Given the eight normalized histograms H_c, the following statistics are computed:

- **Average:**

$$\upsilon_c = \sum_{i=0}^{M-1} i * h_{ci}. \tag{2}$$

- **Entropy:**

$$E_c = -\sum_{i=0}^{M-1} h_{ci} \log_2 h_{ci}. \tag{3}$$

- **Variance:**

$$\sigma_c^2 = \sum_{i=0}^{M-1} (i - \upsilon_c)^2 * h_{ci}, \tag{4}$$

- **Energy:**

$$\varepsilon_c = \sum_{i=0}^{M-1} (h_{ci})^2. \tag{5}$$

After all these statistical measurements have been computed, 32 features per image block were generated (average, entropy, variance and energy of the 8 color channels). Next sections explain how these features are used to train ANNs and, subsequently, as inputs for turbid water recognition.

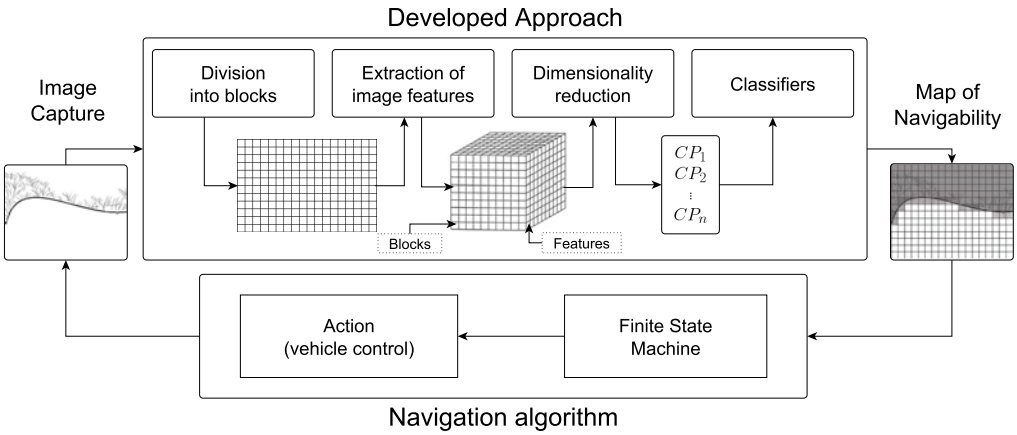


Figure 1: Diagram showing the steps of the proposed methodology.

3.3 Preprocessing and Training

In order to accomplish the proposed goals, we used a supervised approach, i.e., we trained our classifiers using labelled features. Thus, for our experiment, we used a video made in the scenario presented in Figure 10 as a training environment. We selected a set of 15 frames randomly chosen to cover different conditions of luminosity and water turbidity. The acquisition was performed through a monocular camera attached to the prototype vehicle described in Section 4.3. These images were then divided into blocks as previously explained, and the 32 features were extracted.

Next, we performed the manual annotation of image blocks. To this end, an interactive tool was built, in which users were asked to paint in green all navigable regions from the input images, through mouse interactions. Users were supposed to paint disjoint regions, according to the presence of water or not. Figure 2 presents two annotated frames, where blocks marked in red are "not navigable" and the blocks marked in green are "navigable".

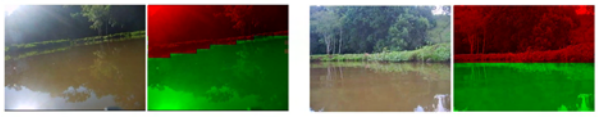


Figure 2: Annotation process of training frames.

A common procedure to avoid data over-fitting and to increase the generality and convergence speed of pattern recognition methods is to employ a dimensionality reduction technique [Bis06a]. We chose to apply Principal Component Analysis of features to accomplish this goal.

First of all, we performed the standardization of the features from the samples. For this, for each image block, the 32 previously described features were computed. We normalized every feature f_j with respect to their range of values, as follows:

$$\hat{f}_j = \frac{f_j - f_{min}}{f_{max} - f_{min}} \ , \tag{6}$$

where $\hat{f}_j$ is the normalized value of each feature, f_j is the original value of the feature, $j = 1, 2, ..., 32$, f_{min} is the minimum value of feature j, and f_{max} is the maximum value of feature j, considering all training blocks. Given the normalized values, we computed, for each feature j, the average μ_j and the standard deviation σ_j, considering all samples. Then, each extracted feature $\hat{f}_j$ was standardized, according to the equation:

$$z_j = \frac{\hat{f}_j - \mu_j}{\sigma_j}. \tag{7}$$

The z-scores of the features computed through Equation 7 of every training block were finally submitted to PCA. The use of PCA as a preprocessing step for a machine learning method can accelerate its convergence, since it allows dimensionality reduction and the correlation among features [YZL06a]. Figure 3 shows the labeled z-scores projected in the sub-space defined by the three principal components, achieved through PCA. We can notice a visible separation of the blocks classified as navigable (red) from the non-navigable ones (blue). We can also observe that this separation is non-linear. Due to this fact, an artificial neural network was employed as a classifier.

The values of μ_j, σ_j and the matrix of the sorted eigenvector from the covariance matrix M obtained in PCA are stored to be used to compute features from images acquired during the experiments conducted in real environments described in Section 4.3. The eigenvectors are sorted according to crescent order of eigenvalues.

We defined the ideal dimension of principal components based on the work by Ian [Jol02a] to minimize the complexity subject to a limit on the fidelity of the problem. According to the author, the set of components

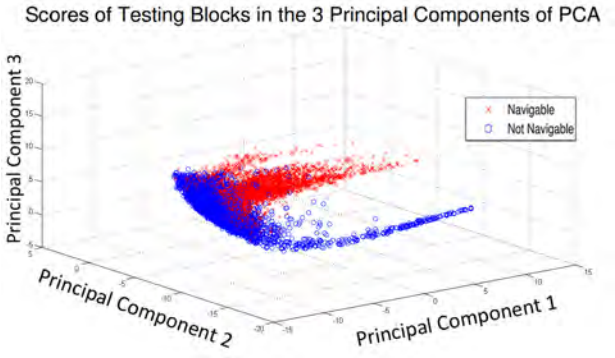


Figure 3: Standardized features from training blocks projected into the subspace defined by the three principal components from PCA.

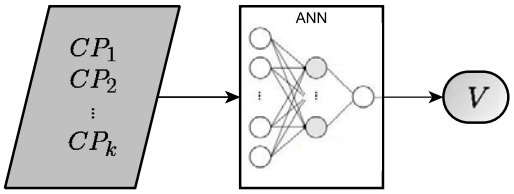
for analysis of values is above a threshold $\hat{r}$ eigenvalues ≥ 1. In our experiments we verified that $81,25\%$ of data variability are incorporated by projecting standardized features into the subspace of the six principal components. Due to this fact, the data coordinates in the subspace of dimension $k = 6$ are then used as input for training the ANN. Thus, the data dimensionality was reduced to $k = 6$. The values μ_j and σ_j are kept and used to standardize the features extracted from new image blocks that must be classified when the vehicle is operational. Next section details the architecture, training and performance of ANNs.

3.4 Classifier

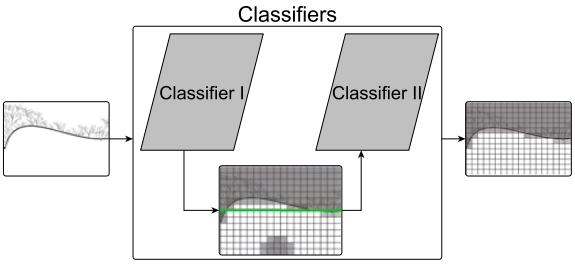
This subsection presents the two ANN classifiers developed for the turbid water surface identification. The purpose of ANN classifiers is to determine if an image block corresponds or not to a navigable surface. The classifier can be defined as follows: B is a block of an image to be classified and $CP = \{ CP_1, CP_2, \cdots CP_k \}$ is the set of coordinates of the extracted features in the subspace of k principal components, computed as explained in the previous section. Thus, the ANN classifier receives the CP's scores as input and returns a value $V \in [0,1]$. The smaller the value of V, less likely it is for a block to correspond to a navigable surface; whereas the greater the value of V, the greater the probability of being a block that corresponds to a navigable surface. We trained two ANNs: the first one described in Section 3.4.1, aims to recognize turbid water surfaces, and the second one aims to recognize regions of reflection on these surfaces, as explained in Section 3.4.2. Figure 4 shows the scheme for the classifiers' modelling (a) and their final architecture (b).

3.4.1 Navigable Surface Identification

In order to solve the problem stated in this approach, we adopted a 3-layer Multilayer Perceptron topology for ANNs [Bha10a]. The input layer has k neurons since



(a) Classifier modeling



(b) Architecture of classifiers

Figure 4: (a) shows how each classifier is built and (b) shows the architecture of our classifiers.

the input data are the set of scores of the training image blocks (in our case, $k = 6$). The intermediate layer has $\frac{k}{2}$ neurons. The output layer has only one neuron since the output is a scalar value $V \in [0,1]$. Figure 4-(a) shows the modeled classifier. If $V < 0.5$, the output means that the image block belongs to a non navigable region; if $V >= 0.5$, that block will be considered as navigable. Intermediate values indicate a low confidence in the classification. Figure 5 shows the values of V for image blocks from the input image on the left, mapped to greyscale images.

We use the resilient propagation algorithm for training our multilayer feedforward network [KNS99a]. According to Svozil et al. [SKP97a], it increases the resolution capability for non-linear problems and ANN becomes very robust, i.e., their performance degrades gracefully in the presence of increasing amounts of noise. In this algorithm, synaptic weights of the network are adjusted according to signal error propagation [Hay98a]. In order to plan an assessment of the convergence of ANNs in the training phase, we used the method developed by Shinzato at al. [SGOW12a]. This method assigns a weight to the classification error for a given ANN, by computing a score S. For the sake of exemplification in this method, a greater weight is assigned to an ANN output with error of 0.1 than an output with error of 0.2. Through this weighted score, there is a tendency to "reward" ANNs with fewer large errors or several small errors and "punish" the other ones. Equation 8 shows how to calculate the score:

$$S = \frac{\frac{1}{N.p(0)} \left(\sum_{i=0}^{hmax} h(i).p(i) \right) + 1}{2.0}, \tag{8}$$

where N is the number of classes ($N = 2$), $h(i)$ is the number of errors ranging from $\frac{i}{hmax}$ to $\frac{i+1}{hmax}$ and $hmax$ is the number of intervals to be considered for discretization in the error counting process. The value of $hmax$ determines the precision for interpretation of the output from the ANN. In other words, e.g., if $hmax = 10$, then the output that has a real value ranging from 0 to 1 is divided into 10 intervals of errors: an error interval for values between 0.0 and 0.1, other error interval for values between 0.1 and 0.2, and so on. After the training phase, the network is executed for each block.

The training of ANNs is repeated until the convergence is reached. In the proposed implementation, the stop criteria adopted is $S = 95\%$ or a limit of 5.000 epochs. Our first ANN reached 95.52% in 1.730 epochs, and the second ANN 95.29% in 460 epochs. After this process, we kept the best ANN, to be used in real time for image blocks' classification.

3.4.2 *Reflection Zone Identification*

In our experiments, we learnt from the classification results of the ANN described in the previous section that a high number of false negatives occur in regions with high reflectance on the water surface. In order to address this problem and enhance performance, a second ANN was trained. The goal of this second ANN is to correctly classify blocks belonging to reflection zones on the water surface as navigable. The topology of this second ANN is the same as described in Section 3.4. The input for the training step consists of features extracted from image blocks manually classified as "reflection zone" (therefore, "navigable") and "non reflection zone". These features are extracted following the same procedures described on Section 3.3.

Due to similarities of reflection zone features and other non navigable zone features, we only applied this second ANN to image blocks below an automatically computed horizon line $L1$ (shown in Figure 5). This line is defined by the upper row of blocks that have at least 25% of classification as "navigable" by the first ANN. Only features extracted from blocks classified as "non navigable" by the first ANN and below the horizon line are submitted to the second ANN. Figure 4-b illustrates this flow. Images in Figure 5 exemplify inputs and outputs from both ANNs. On the left side images, one can note reflection zones on the turbid water surface. The output of the classification computed by the first ANN is shown in the central images, where values of V are mapped to greyscale values (brighter blocks indicate navigable regions). The red ellipses in Figure 5 indicate false negative zones due to reflection and $L1$ indicates the horizon line. Then, the images with blocks classified as "non navigable" by the first ANN that are below $L1$ are submitted to the second ANN.

Figure 5: On the left are the two input images; the images of the middle show the results from the classification of the first ANN, with reflection zones marked by the ellipses; the images on the right side show the result with the combination of both ANN classifiers.

3.4.3 *Classifying New Images*

The model described in previous sections was embedded in an aquatic vehicle, as a prototype. More details on this prototype can be found in Section 4.3. Once the vehicle is on the water surface, new images are acquired by the coupled camera. These images are converted to HSV and YUV space colors, split into 8 color channels, divided into blocks and, for each block, statistics defined in Equations 2 to 5 are computed. This features are then normalised and standardized, according to Equations 6 and 7, which lead us to z-score values.

Given the set of z-scores of new image blocks, we must project them into coordinates of the PCA space. To this end, we used the autovector matrix M for the change of basis of the extracted features:

$$PC = M.Z. \tag{9}$$

where M is the matrix of sorted eigenvectors obtained by PCA, and Z is the vector of normalized and standardized features extracted from each block.

The scores of k's principal components corresponding to each block are then submitted to the first ANN. The horizon line $L1$ is then determined. Blocks with value $V < 0.5$ assigned by the first ANN below the horizon line are submitted to the second ANN. The output of these procedures is a matrix, whose elements correspond to an image block. From now on, this matrix will be addressed as map of navigability. This map will guide the decision-making about the direction the vehicle must follow. Next section explains how the decision-making process was implemented.

3.5 Navigation algorithm

Given the navigation map composed by the output from blocks' classification, a decision making process must be employed to guide the navigation of the aquatic vehicle. This process begins with the subdivision of the navigation map into four regions, as shown in Figure 6-(a). For each navigation map, the regions $SP1$, $SP2$, $SP3$ and $SP4$ are defined by lines $L1$, $L2$ and $L3$. $L1$ is the horizon line that also appears in in Figure 5.

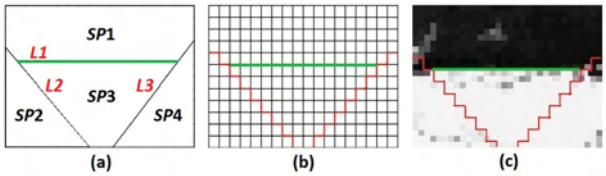


Figure 6: (a) Definition of areas for decision making; (b) search of classifications based on predefined areas; (c) combination of predefined areas with the navigation map.

Lines $L2$ and $L3$ of Figure 6(a) are defined, respectively, according to:

$$\begin{aligned} y_2 &= \tfrac{1}{3}nr + x_2, \\ y_3 &= \tfrac{1}{3}nr + nc - x_3, \end{aligned} \tag{10}$$

where nr is the number of rows of blocks, nc is the number of columns of blocks, y_2 and y_3 are the rows, x_2 and x_3 are the columns of lines $L2$ and $L3$, respectively. The origin is in the upper left corner of the navigation map and y is oriented top down.

Once the areas are delimited, an FSM defines if the vehicle must remain in the same direction, turn left, turn right or stop. First of all, we computed the number of blocks classified as "navigable" in each region. The decision process can be summarised as follows: (1) if the row of the horizon line $L1$ is higher than $\frac{2}{3}.nr$, the vehicle should stop. This occurs mainly when there is few or no navigable blocks ahead of the vehicle; (2)if the number of blocks classified as "navigable" in $SP2$ is higher than in $SP3$ and $SP4$, the vehicle should turn left; (3) if the number of blocks classified as "navigable" in $SP3$ is higher than in $SP2$ and $SP4$, the vehicle should keep forward; (4) if the number of blocks classified as "navigable" in $SP4$ is higher than in $SP2$ and $SP3$, the vehicle should turn right.

Figure 7 illustrates this FSM, with the diagram of actions to be taken according to analysis carried out on the navigation map and the predefined areas.

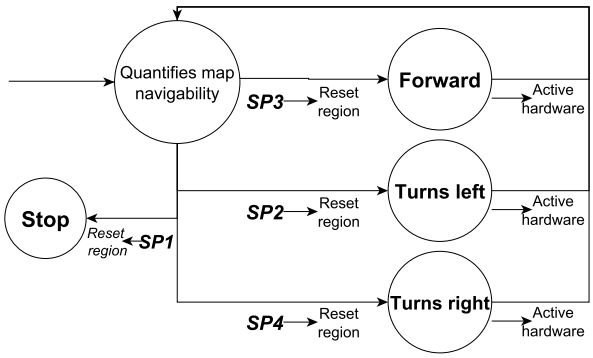


Figure 7: Decision diagram for FSM actions.

The approach to decision making is a proof of concept, developed to ensure fast performance when embedded in the aquatic vehicle. We use FSM because we can easily describe a sequence of states considering different contexts for each input image. Then, it is easy to change from one state to another, defining a specific action to be taken for each state. Next section presents and discusses the results achieved by our approach.

4 RESULTS AND DISCUSSION

This section presents some obtained results, aiming to validate the presented approach. Subsection 4.1 presents the scenarios where the images were collected. The performance metrics evaluated and the results of the tests in real environments are presented in subsection 4.2. In subsection 4.3 we describe our prototype.

4.1 Images and environment

We chose three different environments for extracting the images used to evaluate the developed approach. All images were collected from these environments under different timetables and after a heavy period of rain, in order to achieve the characteristic of turbid water surface. Thus, we tried to approximate as close as possible to the conditions of a real situation where an autonomous vehicle can assist navigation in a post-disaster environment. Figure11-(c) shows our prototype in action, and some of these frames of each evaluated scenario are presented in Figure 8. Each scenario with its peculiarities will be further described.

4.1.1 Scenarios' description

Scenario I corresponds to a rural environment, made up entirely of vegetation, with many trees, rocks and grass on the slope. Figure 8-(a) exemplifies some frames of this scenario. It is possible to notice on these images that the aquatic surface presents large incidence of reflection of the sky, changing the optical property of the turbid water surface.

Scenario II is also a rural environment, but it presents less vegetation and some houses, some of which even working as a form of obstacle to the boat. Figure 8-(b) presents some frames of scenario II. In these images, it is possible to see that there was little incidence of sky reflection on the water surface, showing a subtle reflection of vegetation and houses.

Scenario III corresponds to an urban environment, depicting a real situation of natural disaster. This environment is more complex, since it presents heterogeneous situations. As shown in Figure 8-(c), the images extracted from this scenario can contain, for example, people, cars, animals, and buildings.

4.2 Approach evaluation

Considering the ROC (Receiver Operating Characteristics) analysis [Faw06a], the evaluation was performed

(a) Frames of scenario I

(b) Frames of scenario II

(c) Frames of scenario III

Figure 8: Some examples of frames illustrating each environment used to evaluate the developed approach.

in terms of *accuracy*, *sensitivity* and *precision*, as defined in Equations 11, 12, and 13, respectively.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN}, \tag{11}$$

$$Sensitivity = \frac{TP}{TP+FN}, \tag{12}$$

$$Precision = \frac{TP}{TP+FP}, \tag{13}$$

where TP, TN, FP, and FN refer to True Positive, True Negative, False Positive and False Negative, respectively.

We consider that the proposed approach had a satisfactory accuracy rate in our experiments (Figure 9). It achieved an average accuracy of 95.85% with standard error of 0.924 for scenario I, an average accuracy of 93.35% with standard error of 0.882 for scenario II, and an average accuracy of 91.21% with standard error of 0.980 for scenario III. Figure 10 shows the results for some random frames classified in each scenario.

By analyzing the generated average values for each scenario, it is possible to verify better results of the evaluated metrics for the first scenario. One reason for this may be due to the fact that some images acquired in this scenario were used for training the ANN. Scenario II is quite similar to the first one used for training. Thus,

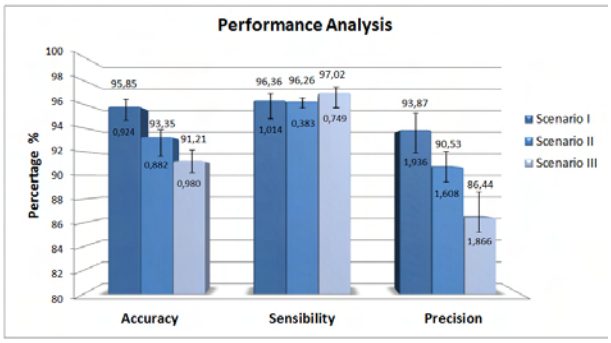


Figure 9: Evaluation results for each scenario.

although it is an unknown scenario, a good result was obtained with an average of 90% among the evaluated metrics. Scenario III corresponds to an adverse environment with a lot of diversity, such as people, houses, cars and objects floating on the water surface. Even so, the approach proved to be efficient, since it has a good sensitivity evaluation, which demonstrates a good performance in identifying the surface with a high rate of true positive values. On the other hand, lower values for precision are due to high rate of false positive values.

4.3 Embedded approach

In order to evaluate the developed approach for autonomous navigation in a real environment, it was embedded in an aquatic vehicle. We build and develop our approach using the programming language C, with support of OpenCV library, OpenMP for multiprocessing programming, and Fast Artificial Neural Network Library (FANN), a free library that implements an ANN multilayer in language C [Nis05a]. The hardware used was a Raspberry Pi board (RPI) model 2 and a Raspberry camera. Figure 11-(b) shows the prototype of the aquatic vehicle with the RPI board and camera connected. Its advantage is the processing totally made on the boat, without the need of having communication or sending commands through an external computer.

Results achieved with the RPI 2 board were: 46% of processor usage, 308.9 MB of memory for execution and 2.5 frames per second (FPS). Figure 11-(a) shows our approach running on the operating system RPI 2. Analyzing the performance of obtained results and considering the usual speed of aquatic vehicles it's possible to say that 2.5 FPS is an acceptable performance.

We used our prototype to evaluate the navigation algorithm, which is based on the generated navigation map. For this evaluation, we collected 48 frames of the described scenarios, with twelve frames for each possible action command defined in our FSM (four for each scenario). Then, we analysed each frame to define the best action or the expected command considering the aquatic surface and its obstacles, and we compared them with the executed command. Table1 presents the

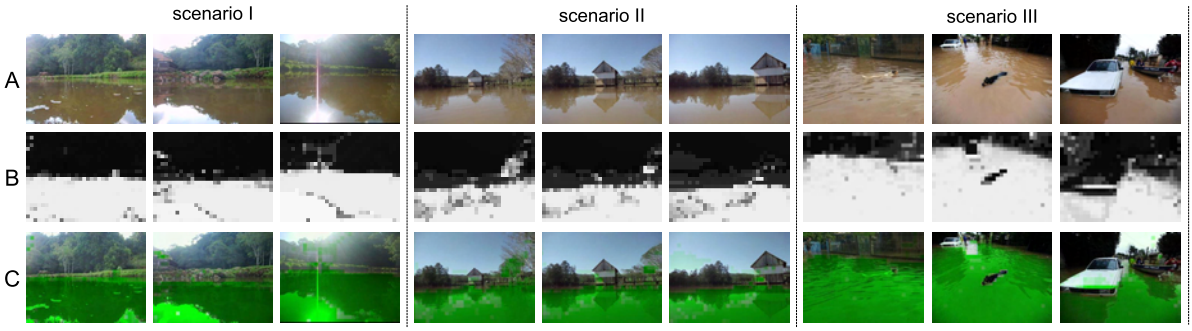
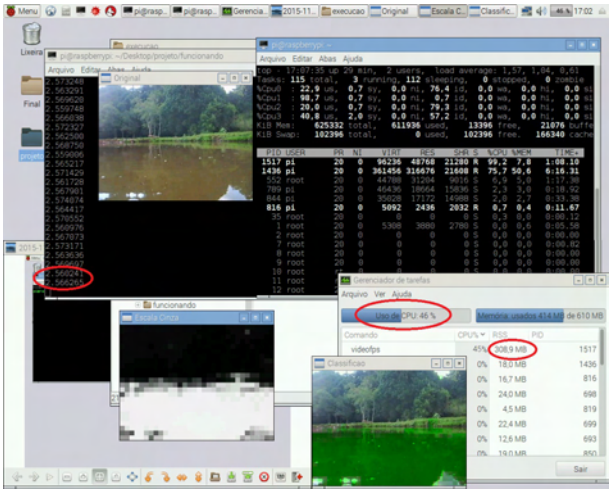


Figure 10: Results obtained for each scenario: (a) input frame; (b) map of generated navigability, and (c) overlay to indicate the navigable region.



(a)

(b) (c)

Figure 11: (a) Performance evaluation of the approach on the RPI 2 board; (b) Prototype built for approach evaluation; (c) Prototype running our approach to collect obtained results in a real environment.

expected commands for these selected frames and the executed commands by our algorithm.

For the obtained average of 66.25%, we considered only the expected values as correct, even though other commands could also be suitable. The low value for the "stop" command is because the vehicle was programmed to stop just when there were few or no navigable blocks ahead of it.

5 CONCLUSION

In this work we proposed an approach for automatic identification of navigable turbid water surfaces, based on computer vision techniques. Artificial neural networks (ANNs) were also used to build a classifier designed to generate a navigation map, and principal component analysis (PCA) was performed to compress the extracted information used as input to ANN.

Table 1: Comparison of expected and executed commands by the developed FSM.

Set of commands defined	Expected Commands	Commands Executed
Forward	12	8
Turn right	12	9
Turn left	12	10
Stop	12	5
Hit average movement	66,25%	

The proposed approach was quantitatively evaluated using a dataset containing images extracted from three different scenarios. Experimental results indicated that the approach effectively identified navigable region achieving between 91.21% and 95.85% of accuracy. For testing and evaluation of our approach, we built a prototype used in three real environments in order to demonstrate the adaptability and viability of our approach to autonomy of aquatic vehicles. Thus, we believe it can be used to assist navigation of an autonomous vehicle in a post-disaster environment.

For future work we intend to use pre and post-processing techniques in the navigability map, mainly to improve false positive results. We would also like to improve our navigation algorithm, in order to develop better search directives on the navigability map and, consequently, execute more precise commands in our FSM. Furthermore, we also want to use other sensors such as laser or distance sensors to increase the capacity of performance in navigation.

6 ACKNOWLEDGMENTS

The authors would like to thank Brazilian agency CAPES for the financial support. This work was also partially supported by PUCRS.

7 REFERENCES

[ASN11a] Achar, Supreeth and Sankaran, B. and Nuske, Stephen. Self-supervised segmentation of river scenes. In Robotics and Automation (ICRA), 2011 IEEE International Conference on, pages 6227-6232. IEEE, 2011.

[Bha10a] R. Bhati. Face recognition system using multi layer feed forward neural networks and principal component analysis with variable learning rate. In Communication Control and Computing Technologies (ICCCCT), IEEE International Conference on, pages 719-724, 2010.

[Bis06a] Christopher M. Bishop. Pattern Recognition and Machine Learning. Springer Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[CAN11a] Andrew Chambers, Supreeth Achar, Stephen Nuske, Jorn Rehder, Bernd Kitt, Lyle Chamberlain, Justin Haines, Sebastian Scherer, and Sanjiv Singh. Perception for a river mapping robot. In Intelligent Robots and Systems (IROS), International Conference on, pages 227-234. IEEE, 2011.

[Faw06a] Tom Fawcett. An introduction to roc analysis. Pattern recognition letters, 27(8):861-874, 2006.

[GSW07a] Xiaojin Gong, Anbumani Subramanian, and Christopher L Wyatt. A two-stage algorithm for shoreline detection. In Applications of Computer Vision. WACV. IEEE, pages 40-40. IEEE, 2007.

[HAH11a] Terry Huntsberger, Hrand Aghazarian and Andrew Howard. Stereo vision-based navigation for autonomous surface vessels. Journal of Field Robotics, 28(1):3-18, 2011.

[Hay98a] Simon Haykin. Neural Networks: A Comprehensive Foundation. Prentice Hall PTR, Upper Saddle River, NJ, USA, 2nd edition, 1998.

[HS11a] Hordur Kristinn Heidarsson and G Sukhatme. Obstacle detection from overhead imagery using self-supervised learning for autonomous surface vehicles. In Intelligent Robots and Systems (IROS), International Conference on, pages 3160-3165. 2011.

[IMM09a] Mohammad Iqbal, Olivier Morel, and Fabrice Meriaudeau. A survey on outdoor water hazard detection. International Conference on Information Communication Technology and Systems, pages 33-39, 2009.

[Jol02a] Ian Jolliffe. Principal component analysis. Wiley Online Library, 2002.

[KNS99a] K. Keeni, K. Nakayama, and H. Shimodaira. A training scheme for pattern classification using multi-layer feed-forward neural networks. In Computational Intelligence and Multimedia Applications. ICCIMA. Proceedings. Third International Conference on, pages 307-311, 1999.

[Kro15a] Wolfgang Kron. Flood disasters a global perspective. Water Policy, 17(S1):6-24, 2015.

[Nis05a] Steffen Nissen. Neural networks made simple. Software 2.0, 2:14-19, 2005.

[RM10a] Arturo Rankin and Larry Matthies. Daytime water detection based on color variation. Intelligent Robots and Systems (IROS), IEEE International Conference on, pages 215-221. IEEE, 2010.

[RMB11a] Arturo L Rankin, Larry H Matthies, and Paolo Bellutta. Daytime water detection based on sky reflections. In Robotics and Automation (ICRA), IEEE International Conference on, pages 5329-5336. IEEE, 2011.

[SGOW12a] P.Y. Shinzato, V. Grassi, F.S. Osorio, and D.F. Wolf. Fast visual road recognition and horizon detection using multiple artificial neural networks. In Intelligent Vehicles Symposium (IV), IEEE, pages 1090-1095. IEEE, 2012.

[SKP97a] Daniel Svozil, Vladimir Kvasnicka, and Jiri Pospichal. Introduction to multi-layer feed-forward neural networks. Chemometrics and Intelligent Laboratory Systems, 39(1):43-62, 1997.

[SKV11a] Paul Scerri, Balajee Kannan, Pras Velagapudi and Kate Macarthur. Flood disaster mitigation: A real-world challenge problem for multi-agent unmanned surface vehicles. In Advanced Agent Technology,pages 252-269. Springer, 2011.

[SMB12a] Pedro Santana, Ricardo Mendonça, and José Barata. Water detection with segmentation guided dynamic texture recognition. In Robotics and Biomimetics (ROBIO), IEEE International Conference on, pages 1836-1841. IEEE, 2012.

[SMH04a] Franklin D Snyder, Daniel D Morris, Paul H Haley, Robert T Collins, and Andrea M Okerholm. Autonomous river navigation. In Optics East, pages 221-232. International Society for Optics and Photonics, 2004.

[YRC11a] Junho Yang, Dushyant Rao, S Chung, and Seth Hutchinson. Monocular vision based navigation in GPS-denied riverine environments. In Proceedings of the AIAA Infotech Aerospace Conference, St. Louis, MO, 2011.

[YXL07a] Tuozhong Yao, Zhiyu Xiang, Jilin Liu, and Dong Xu. Multi-feature fusion based outdoor water hazards detection. Mechatronics and Automation International Conference ICMA, pages 652-656. IEEE, 2007.

[YZL06a] Jun Yan, Benyu Zhang, Ning Liu, and Shuicheng Yan. Effective and efficient dimensionality reduction for large-scale and streaming data preprocessing. IEEE Transactions on Knowledge and Data Engineering, 18(3):320-333, March 2006.