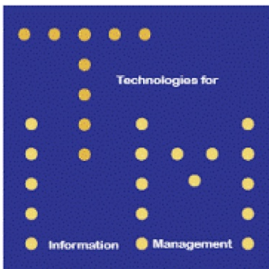
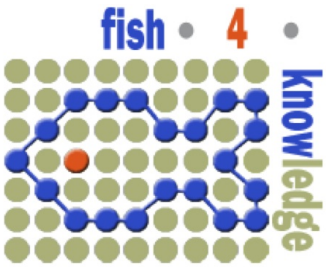




Fish4Knowledge Deliverable D1.2

Fish and Environment Property Description

Principal Author: C. Spampinato, R. Di Salvo, D. Giordano, A. Faro, X. Huang, C. Beyan
Contributors: UCAT, UEDI
Dissemination: PU

Abstract: The Fish and Environment Property Description deliverable (D1.2) of the Fish4Knowledge project describes the list of image cues used for describing fish and environment in order to support higher level components for fish detection, tracking, classification, behavior analysis and data visualization. The adopted descriptors have been developed in order to deal both with the peculiarities of fish appearance and motion in their natural habitat and the scene under analysis. To this end, we have chosen fish descriptors which are as much generic as possible in order to make the above task insensitive to variations in the target's position, size, appearance, orientation, scale and trajectory's length with respect to the camera.
At the same time global and local image and video features have been employed to describe underwater environments for understanding fish-background interactions, supporting data uncertainty management and monitoring the system maintenance needs.

Deliverable due: Month 18

Contents

1 Introduction

Fish and environment description are key tasks in the F4K project because they provide the basic information for supporting the fish identification task (from detection to recognition) and for studying fish interaction with the surrounding environment.
Fish description techniques, in detail, have been investigated and devised for i) discriminating background objects from fish, ii) fish species classification and iii) fish behaviour classification and anomaly detections (Section 2). For the first task, specific real-world fish features, such as object proximity, continuity, symmetry and objectness (see Sect. 2.1), have been employed and applied to a post-processing level to the fish detection algorithms (introduced in Deliverable 1.1) has proved their ability to recover effectively errors occurring in the detection task, especially by reducing false positives. Since fish are generally characterized by erratic and fast movements which lead to frequent changes in size and position, a pre-processing layer before fish classification has been employed aiming at making the descriptors independent from the object's position and orientation. After this pre-processing setting, a combination of boundary, color and texture features is used.
The fish behaviour analysis process (Section 2.3), instead, relies on representing and modeling behaviours through trajectory analysis. More specifically, fish behaviour analysis has been carried out by Hidden Markov Models, since they intrinsically encode spatio-temporal sequences of data avoiding the normalization of different-length trajectories derived from a point-sequence representation of fish movements. Beyond the behaviour classification approach based on HMM, we have also devised methods to identify anomalous behaviours.
Finally, descriptors of the observed underwater scenes have been extracted for the tasks of video, image segmentation and scene classification (Section 3), which are crucial to support our strive towards fully understanding marine ecosystems and displaying the interpretation of the achieved results to marine biologists.

2 Fish Description

Fish description in the Fish4Knowledge project is necessary to support fish detection, recognition and behaviour understanding. In detail, different features have been used/conceived to meet the following three goals:

- to discriminate fish (as coming out from the foreground identification process) from other background objects in order to reduce the number of false positives due to errors during the detection process.
- to identify the species each fish instance belongs to by analysing fish texture, shape and colors;
- to model and recognise fish behaviour and its interaction with the surrounding context.

In the next subsections the features employed to represent a fish, its appearance and its behaviour are described.

2.1 Fish - Background Objects Classification

As described in Deliverable 1.1 several object detection algorithms have been developed for background modeling and foreground identification. However, none of them demonstrated to be generally superior to the other ones in terms of false positives. This led us to add a complementary processing level to fish detectors in order to detect and recover from failures that eventually happen in the detection process. In detail, to discriminate fish from other background objects we adopted a set of specific features of real-world objects. The set of considered features exploit two main concepts: 1) the "human perceptual organization model" to discriminate blobs that are most likely produced by the motion of a biological object from blobs that may arise due to changes in the background (i.e. luminosity), 2) the "motion objectness" to compute the probability that a change detected by the above algorithms is due to fish movement instead of background movement (e.g. corals or algae).

2.1.1 Perceptual Organization Model Features

The ability of humans to identify objects, and more in general structures, without a priori knowledge of their contents is known as perceptual organization, which is governed by the four Gestalt laws that identify some basic principles of whole objects such as proximity, similarity, continuity, symmetry and convexity [1], i.e. real-world objects tend to have convex shape, tend to be symmetric with respect to a reference axis, etc. To measure quantitatively the Gestalt laws in real-word applications, we have adopted the method proposed in [2] which encodes such laws into a boundary energy function:

$$E\left[\partial R\right] = \frac{-\int\int_R f\left(x,y\right) dxdy}{L\left(\partial R\right)} \tag{1}$$

where ∂R is the object's contour, $L\left(\partial R\right)$ the contour's length, and $f\left(x,y\right)$ is a weight function for each point belonging to the object. The first step for the evaluation of $f\left(x,y\right)$ is a superpixel segmentation [3] of the object's region into homogeneous patches. For each pixel (x,y), belonging to patch i, the corresponding weight is computed as:

$$f\left(x,y\right) = e^{-\theta \bullet \eta\left(S_i - S_a\right)} \tag{2}$$

In this formula, S_i is a two-component vector $[B_i \;\; C_i]$, where B_i and C_i represent, respectively, the boundary complexity of patch i and its cohesiveness with the other patches which make up the object. S_a is a reference vector computed on the largest patch of the object [2]. θ is a weight vector and η is vector element-by-element absolute value.

2.1.2 Motion Objectness

To distinguish between moving blobs produced by fish movement and blobs due to background object movements, we conceived the concept of "Motion Objectness" taking inspiration from the work of Alexe *et al.* in [4] who used the "Objectness" to identify generic class objects on still images. The intuition behind this concept is that any fish is characterized by specific intraframe and interframe properties which make its movement different from background objects.

In the following we use the term "blob" to indicate a particular area closed by a contour as provided by the fish detection processes.

- **Intraframe Properties**. To describe the peculiarities of a fish within an image, we took under consideration the following characteristics:

  – *Closed boundary in space*. This property aims at evaluating how the blob's contour matches the object's boundary. To measure it, we assess the density of edges included in a blob. Let δ_b and b^θ_{in} be, respectively, the contour of the considered blob b and the inner blob obtained by shrinking b of a factor θ. The edge density is given by:

  $$ED = \frac{\sum_{p \in b^\theta_{in}} M_{ED}(p)}{A_{b \setminus \delta_b}} \tag{3}$$

  where $M_{ED}(p)$ is a binary edgemap and indicates if the pixel p is classified as edge by a Canny edge detector. $A_{b\setminus\delta_b}$ is the area of the blob b minus its contour δ_b ($\setminus$ is the set difference symbol).
  Moreover, we also used the percentage of superpixels intersecting the blob's contour, computed as follows:

  $$SI = 1 - \frac{\sum_{s \in \mathcal{S}} min(|s \setminus b|, |s \bigcap b|)}{|b|} \tag{4}$$

  where $\mathcal{S}$ is the set of superpixels computed as in [3] and $|b|$ the blob's area.

  – *Appearance difference from surrounding areas*. The dissimilarity of an object to its surrounding area is estimated by analysing color contrast along the object's boundary. Let b^θ_{out} and b^θ_{in} be the outer and the inner blob obtained, respectively, by dilating and shrinking the original blob b of a factor θ (empirically set to 2 in our implementation), the color contrast along the boundary of a blob b is computed as the Chi-square distance between the LAB histograms of the two rings (outer and inner) surrounding the object's boundary:

  $$CC = \chi^2(h(b \setminus b^\theta_{in}), h(b^\theta_{out} \setminus b)) \tag{5}$$

  – *Internal homogeneity of color and texture*. Most fish appear to have a limited number of colors and a uniform texture (due to the low resolution of the video), especially when compared with complex background objects (e.g. algae, corals, rocks, etc.). The internal homogeneity of a blob has been assessed by computing the average color value and the average texture of all the superpixels in a blob. The more similar these average results are, the more likely the detected blob is actually a fish. The average color homogeneity is given by the following formula:

  $$HC = 1 - \frac{\sum_{s \in \mathcal{S}} \left|\left|C_s - \overline{C}\right|\right|}{Dim(\mathcal{S})} \tag{6}$$

  where $\mathcal{S}$ is the same set of superpixels described above, C_s is the average color within each superpixel s, $\overline{C}$ is the average color within the whole blob and $Dim(\mathcal{S})$

is the number of superpixels. Analogously, the measurement of the internal texture homogeneity is performed by averaging the outputs of a bank of Gabor filters applied to each superpixel in the detected blob.

- *Preferred positions*. The spatial coordinates of the blob's centroid are also used to measure "objectness", since we assume that some positions are more likely than others to contain objects.

- **Interframe Properties**. Any fish holds the *motion coherence* property that allows us to distinguish it from the rest of the scene. To measure this property, we propose two cues based on the object's motion vector (calculated according to [5]):

  - *Difference of motion vectors at object boundary*: Let $M_{b,in}$ and $M_{b,out}$ be the average motion vectors computed, respectively, in the ring just inside and the ring just outside the blob's boundary, the motion difference at the boundary Δ_{MV} is assessed as the Chi-square distance between the motion histograms assessed in the two rings (whose size was set empirically set to 3 in our implementation):

$$\Delta_{MV} = \chi^2(h(M_{b,out}), h(M_{b,in})) \tag{7}$$

  - *Internal motion homogeneity*. This cue is based on the assumption that the internal motion vectors of a correctly-detected fish are more uniform than the ones of a false positive. The detected blob is split into a set of superpixels (as above) and the average of motion vector's magnitude in each superpixel is compared with the global one. The internal motion homogeneity MH is computed as follows:

$$MH = 1 - \frac{1}{Dim(\mathcal{S})} \sum_{s \in \mathcal{S}} \frac{1}{Dim(\mathcal{V}_s)} \left| \sum_{p \in \mathcal{R}_s \cap \mathcal{V}_s} |M_p| - \overline{M} \right|^2 \tag{8}$$

    where $\mathcal{S}$ is the set of superpixels in the analysed blob and $Dim(\mathcal{S})$ its dimension, $\mathcal{R}_s$ is the union between the superpixel's current bounding box $R_s(t)$ and the bounding box of its last appearance $R_s(t-1)$, $\mathcal{V}_s$ is the set of valid points in $\mathcal{R}_s$, i.e. the points whose displacement project them inside $\mathcal{R}_s$ and $Dim(\mathcal{V}_s)$ is the number of valid points in $\mathcal{V}_s$. Finally, M_p is the motion vector (two components: x and y) describing the displacement of pixel p in two consecutive appearances and $\overline{M}$ is the average motion vector between all superpixels.

The feature vector, containing the above objectness's measures and the perceptual organization energy value of each detected blob, is then given as input to a naive Bayes classifier with two classes: *"object of interest" (OI)* and *"false positive" (FP)*, which computes the probability that the considered blob is a fish or not. A detailed evaluation performance of the proposed approach can be found in [6]. Fig. 1 shows an example of estimated probabilities of some detected blobs to be fish. By filtering out all the blobs with estimated probability lower than a threshold, we were able to reduce the number of false positives to about 10% as shown in [6].

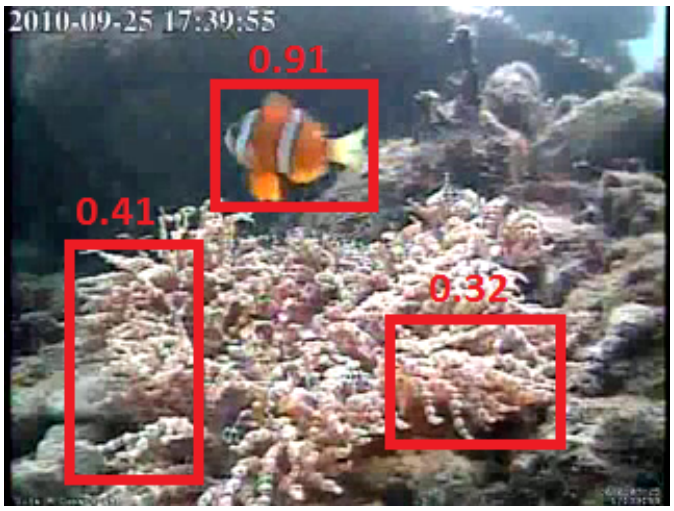


Figure 1: Example of estimated probability of the blobs (for simplicity only the bounding boxes are drawn) to be a fish.

2.2 Fish Species Classification

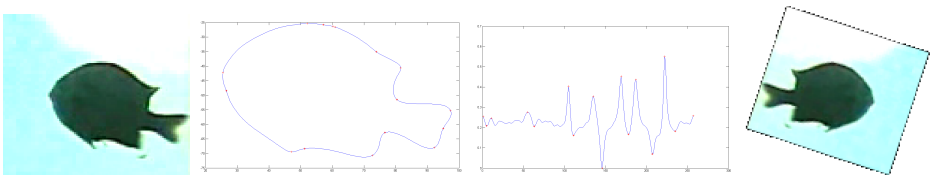
Fish descriptions for species classification have to cope with the peculiarities of fish and their movement; in fact, fish show erratic and fast movements (in three dimensions, which look even faster when processing low fps videos) that lead to frequent changes in size and appearance. This implies that the adopted features must be invariant to affine transformation, object's position, orientation, scale and slant. In order to overcome this limitation we devised a strategy that works on fish contours on a pre-processing module that produces standard fish orientation. In detail, we propose a streamline hypothesis, which uses the assumption that the tail has an abrupt shape because fish need a more frictional tail (caudal fin) to swim and help them keep balance. In order to find the tail's side, we smooth the fish boundary with a Gaussian filter to eliminate some noise, and then calculate the curvature of each boundary pixel as following:

$$\kappa(u,\sigma) = \frac{X_u(u,\sigma)Y_{uu}(u,\sigma) - X_{uu}(u,\sigma)Y_u(u,\sigma)}{(X_u(u,\sigma)^2 + Y_u(u,\sigma)^2))^{\frac{3}{2}}} \tag{9}$$

where $X_u(u,\sigma)$/$X_{uu}(u,\sigma)$ and $Y_u(u,\sigma)$/$Y_{uu}(u,\sigma)$ are the first and the second derivative of $X(u,\sigma)$ and $Y(u,\sigma)$, respectively; $X(u,\sigma)$ and $Y(u,\sigma)$ are the convolution result of 1-D Gaussian kernel function $g(u,\sigma)$ with fish boundary coordinates $x(u)$ and $y(u)$. However, the pixel curvature is sensitive to local corners and so, we normalize it using the logarithm function:

$$\kappa_{normalize} = \begin{cases} log(\kappa) & \text{if } \kappa \geq 1 \\ -log(2-\kappa) & \text{if } \kappa < 1 \end{cases} \tag{10}$$

The fish boundary coordinates are weighted by their local curvature and the vector starting from the center of the mask and ending to the weighted curvature's center estimates the tail' orientation. A typical fish orientation detection procedure is illustrated in Figure 2. Finally, every fish image is divided into four parts (head/tail/top/bottom) according to their relative positions from the fish center.



(a) . (b) . (c) . (d) .

Figure 2: Fish orientation demonstration: (a) original fish image; (b) fish boundary after applying the gaussian filter; (c) curvature along fish boundary; (d) oriented fish image.

This method achieved a stable accuracy (95%) in finding the tail side in 1000 hand-labeled images. This curvature orientation method selects the relative curvature center which is invariant to the contour scale change. After this, 66 types of features are extracted. These features are a combination of color, shape and texture properties in different parts of the fish such as tail/head/top/bottom, as well as the whole fish. We use normalized color histogram in the Red&Green channel and the Hue component in HSV color space. These color features are normalized to minimize the influence of illumination changes. We recompute the range of every bin according to the average distribution over all samples and map them into an 11-bin histogram to take full advantage of all bins, as shown below:

$$\widetilde{B}_i = \sum_{j=a_i}^{a_{i+1}} B_j \qquad s.t. \qquad a_i = \min\{X \in \mathbb{N}^+ \| \Sigma_{j=1}^{X} \overline{B}_j \geq \frac{i}{11}\} \tag{11}$$

where $B_j, j \in \{1, ..., 50\}$ is the original color histogram bin, $\overline{B}_j, j \in \{1, ..., 50\}$ is the averaged histogram over all samples and $\widetilde{B}_i, i \in \{1, ..., 11\}$ is the recomputed bin.

In order to describe the fish texture, we calculate the co-occurrence matrix, Fourier descriptor and Gabor filter. The grey level co-occurrence matrices describe the co-occurrence frequency of two grey scale pixels at a given distance d [7]:

$$C_{\Delta u, \Delta v}(i, j) = \sum_{p=1}^{n} \sum_{q=1}^{m} \begin{cases} 1 & \text{if } I(p, q) = i \text{ and } I(p + \Delta u, q + \Delta v) = j \\ 0, & \text{otherwise} \end{cases} \tag{12}$$

The frequency is calculated for several orientations λ. Moreover we compute contrast, correlation, energy, entropy, homogeneity, variance, inverse difference moment, cluster shade, cluster prominence, max probability, auto correlation and dissimilarity. The histogram of the oriented gradients and moment invariants, as well as affine moment invariants, are employed as the shape's features. Furthermore, some specific features like tail/head area ratio, tail/body area ratio, *etc.* are also included. All features are normalized by subtracting the mean and divided by the standard deviation (z-score normalized).

The fish recognition module was tested on 3179 fish images with a 6-fold cross validation procedure, sequential forward feature selection procedure and hierarchical tree as classifier achieving an average performance of over 90% on 10 fish species.

2.3 Fish Behaviour Classification

Fish behavior understanding is of key importance for marine biologists since changes in behavior patterns (e.g. finding abnormalities and/or detecting distinctive behaviours of different

species) might be correlated to environmental effects such as pollution and climate change.
Whilst human behaviour understanding is one of the most exciting and explored topics of recent research in computer vision and multimedia, relatively little has been done to understand animal behaviour.
Most of the human-centered applications exploit as features for behaviour modeling and learning, information derived from a direct representation of the scene (e.g. visual concepts), since trajectories, silhouettes and, more in general, object part positions may be very sensitive to viewpoint changes [8]. On the contrary, these features seem to work better on the animal domain due to the structure of the animals' bodies (e.g. fish body is less structured than human body) [9] and to the fact the cameras are rather static (e.g. it is not easy and handy to change viewpoints of cameras in the underwater domain). According to this, we have employed trajectory features to model and describe fish behaviour.
However, one of the main problems in the analysis of trajectories is finding an appropriate way of representing them. The typical point-sequence representation, although it contains all the information describing the movement of an object, is often difficult to work with, since comparing different-length trajectories implies a normalization of the number of points, with the consequent risk of over- or under-sampling; moreover, it is not practical to represent a generic motion pattern as a sequence of points. Histograms of position, speed, orientation, etc. may also be employed to describe trajectories, but they lose all temporal information, which is an essential part of the pattern recognition process. On the contrary, HMMs are often used in the description of trajectories, since they encode intrinsically spatio-temporal sequences of data and also provide intuitive algorithms to generate sample trajectories and to check whether an input trajectory matches the pattern learned by the HMM.
Aiming the scope of F4K, trajectory analysis has been adopted for prominent activity recognition and abnormal behavior detection. In detail, HMMs were adopted both for fish species behaviour (such as solitary behaviour, pairing, etc..) recognition and also for identifying uncommon trajectories. Beyond the HMM-based approach, anomaly detection was carried out also by a rule-based trajectory filtering mechanism, which aims at extracting normal fish trajectories as much as possible (ideally all) while rejecting abnormal trajectories.

2.3.1 HMM Modeling of Fish Trajectories

A Hidden Markov Model (HMM) is a stochastic model describing a Markovian process where the states are not directly observable, differently from a regular Markov chain. The estimation of the current state is then performed by analysing the system's output variables, which depend on the current state: assuming discrete output variables, each state has a probability distribution over the values these variables can assume, hence by analysing the output sequences it is possible to obtain the information necessary for the estimation of the state sequence. HMMs can be trained from output sequences, making them especially appropriate for temporal pattern recognition [10]. The parameters of an n-state HMM with m discrete output variables are:

- Prior distribution π: probability for the initialization of the HMM's first state.
- State transition probabilities A: an $n \times n$ matrix whose $a_{i,j}$ element is the probability of going from state i to state j.
- Emission distributions B: an $n \times m$ matrix whose $b_{i,j}$ element is the probability that, in state i, the output token will be j.

ID	Fish Species	Behaviour	Trajectories
DR_S	Dascyllus Reticulatus	Solitary	104
CM_S	Chromis Margaritifer	Solitary	106
PD_S	Plectrogly-Phidodon dickii	Solitary	95
PM_S	Pomacentrus Moluccenis	Solitary	60
CT_S	Chaetodon Trifascialis	Solitary	57
SB_S	Scolopsis Bilineate	Solitary	237
AC_S	Amphiprion Clarkii	Solitary	63
SF_S	Siganus Fuscescens	Solitary	51
DR_P	Dascyllus Reticulatus	Pairing	104
CM_P	Chromis Margaritifer	Pairing	144
PD_P	Plectrogly-Phidodon dickii	Pairing	138
CT_S	Chaetodon Trifascialis	Pairing	90
SB_P	Scolopsis Bilineate	Pairing	104

Table 1: Ground Truth Trajectories for Fish Species

The set of the three model matrices is typically referred to as λ. Of course, the structure and dimensions of these matrices can vary if there are multiple output variables or if the distribution is continuous, as is in the fish case. A detailed description of continuous-output HMMs using mixtures of Gaussians is out of the scope of this document and can be found in [10].
In this context we extend each fish trajectory $T = \{(x_0, y_0), (x_1, y_1), ..., (x_n, y_n)\}$ (i.e. the sequence of centroid coordinates provided by the tracking algorithm) with an HMM, whose output variables are position coordinates, speed and direction of the fish, modeled by mixtures of Gaussians. Differently from the traditional approach, we do not force the states of the model to match real world locations, instead we let the HMM learn its own internal configuration by applying the Baum-Welch algorithm and feeding a trajectory or a set of trajectories as input. Moreover, we do not apply the state transition probability as in [11] because it does not hold for 3D unconstrained motion such as fish movement. All states have the same initial probability.
This HMM based fish description has been applied both for learning fish-species behaviour (solitary, pairing, etc.) and for detecting uncommon trajectories which may be either actual anomalous fish behaviour to be investigated by marine biologists or errors of the tracker.
In the former case, we have employed a species related behaviour recognition. For each fish species and for each behaviour type shown in Table 1, we trained a Hidden Markov Model specialised in the recognition of the corresponding trajectory patterns. Each HMM was trained using the Baum-Welch algorithm, and the number of states and output mixtures were both set to 4.

Table 1 also shows the number of ground truth-trajectories labeled for each of the considered combinations. For each HMM, 70% of the corresponding events were used for training and the rest 30% for testing, meaning that the trajectories classification module was trained on 947 trajectories and tested on the remaining 406 trajectories. Table 2 shows the classification performance of each single HMM, in terms of detection rate (DR) and false alarm rate (FAR) given in percentage. The classes being detected are the ones of Table 1.

For anomaly detection, after modeling trajectories through HMM, trajectory clusters representing common paths are built and then the trajectories that do not match the common

ID	DR	FAR
DR_S	70.9%	35.7%
CM_S	71.8%	39.1%
PD_S	72.4%	33.3%
PM_S	100.0%	33.3%
CT_S	100.0%	33.3%
SB_S	77.4%	41.5%
AC_S	73.6%	0.0%
SF_S	100.0%	0.0%
DR_P	75.0%	27.7%
CM_P	73.9%	30.7%
PD_P	75.0%	14.2%
CT_S	71.4%	25.0%
SB_P	73.3%	33.3%
Average	**81.9%**	**24.11%**

Table 2: Trajectory classification performance by species and event type.

behaviour (described by means of the above clusters) are considered "unusual". In order to represent the trajectories in a format more appropriate to clustering than HMMs, Multi-Dimensional Scaling (MDS) [12] is applied, which projects HMMs to a relatively short vector space, while maintaining the distances in the original HMM space. Since MDS exploits a distance matrix among input data to reduce the original space, we have introduced a probabilistic metric to compare HMMs describing trajectories. If λ_1 and λ_2 are the HMM parameters which model trajectories O_1 and O_2, we adopt Juang and Rabiner's approach [13] in defining the (asymmetric) distance $D(\lambda_1, \lambda_2)$ as:

$$D(\lambda_1, \lambda_2) = [log\ L(O_1|\lambda_1) - log\ L(O_1|\lambda_2)] \tag{13}$$

where $L(O_x|\lambda_y)$ is the probability that trajectory O_x is modeled by λ_y. Since equation (13) is not symmetric, an averaged distance is computed as:

$$D_{ave}(\lambda_1, \lambda_2) = \frac{1}{2}(D(\lambda_1, \lambda_2) + D(\lambda_2, \lambda_1)) \tag{14}$$

In order to avoid errors with short data sequences, the HMM parameters λ_k for a single trajectory are interpolated (linear interpolation between the π, A, and B matrices) with λ_{all}, representing the HMM parameters obtained by training a model with all trajectories in the training set:

$$\begin{aligned} \pi_k &\leftarrow \beta\pi_k + (1-\beta)\,\pi_{all} \\ A_k &\leftarrow \beta A_k + (1-\beta)\,A_{all} \\ B_k &\leftarrow \beta B_k + (1-\beta)\,B_{all} \end{aligned} \tag{15}$$

The MDS algorithm takes as input a distance matrix D, where each element $d_{i,j}$ is equal to $D_{ave}(\lambda_i, \lambda_j)$. Starting from D, a B matrix (which projects the original points into points

whose barycentric coordinates are the origin) is computed, whose eigenvalues and eigenvectors are then employed to compute the projected trajectory vectors (details are described in [12]). These vectors are then clustered using unsupervised K-means algorithm [14]. We perform two clustering cycles: one to filter out outliers from the training dataset and the second one to build the clusters of common paths which are then used to detect anomalous trajectories. To recap, the employed steps for anomalous trajectory detection are:

- First, trajectory modeling through HMM
- Clustering of the HMMs modeling of the input trajectories (training data set) into k clusters.
- Train k HMMs with the trajectories in each cluster.
- For each trajectory, detect it as anomalous if the maximum likelihood between it and the k HMMs is lower than a threshold.
- Re-clustering of the input trajectories training data set into k clusters, leaving out the trajectories identified as anomalous by the previous step. This step makes the final clusters more accurate and with less variance around their centroids.

The resulting k HMMs are then used to evaluate the likelihood that a test trajectory belongs to one of the common path clusters, and if the maximum likelihood is smaller than a threshold, the trajectory is labeled as anomalous.
We performed several tests in order to investigate the best configuration of the HMM/K-means system to identify incorrect trajectories. The training set T_{train} was made up of 300 correct fish trajectories with lengths between 5 and 30 points, manually selected from the tracking results on 10 underwater videos (320×240 resolution at 5 fps) from the F4K repository. The test set used to evaluate the system consisted of 3700 trajectories, with uniformly distributed lengths between 3 and 50, equally divided into correct and erroneous trajectories.
Since the output results depend on the HMMs' parameters, we performed a parametric analysis of the performance in order to find the best HMM configuration for the target we are dealing with. The first HMM parameter we analyzed was the number of iterations for the Baum-Welch algorithm used to train the models. We found out that, for HMMs built from a reasonable number of trajectories (for example, those representing whole clusters), 45 iterations was a good compromise between computation time and the maximization of the probability that trajectories in the training set matched that HMM. However, for HMM representing single trajectories, this value resulted to be too high, because states transition probabilities would "flatten", practically making states equiprobable. Moreover, whichever smaller number of iterations we applied, the input data sequence was too short to make the resulting HMM effectively learn the trajectory pattern. The solution we found to this problem was to train the single-trajectory HMM with 15 iterations, and interpolating it with the HMM parameters obtained by learning all trajectories in the training set.
After setting the number of training iterations, the next parameter we analyzed was the number of clusters. We ran a few simulations, with varying HMM parameters (the number of states and output mixtures), and applied the method described in [14] to estimate the optimal number of clusters. According to most of these simulations, 3 clusters were enough to represent the

HMM configuration			Results	
S	*M*	*T*	*DR*	*FAR*
5	4	-100	55.6%	36.7%
5	4	-120	21.0%	3.1%
5	16	-100	57.3%	32.7%
5	16	-120	24.3%	7.1%
15	4	-100	55.0%	28.61%
15	4	-120	23.9%	2.0%
25	4	-100	54.2%	29.6%
25	4	-120	26.1%	5.1%
10	4	-100	18.9%	1.0%
10	4	-120	22.4%	0.0%
15	16	-100	59.6%	29.5%
15	16	-110	39.2%	13.3%

Table 3: Performance of the system with different HMM configurations.

variability of the training set (and no simulations gave a result smaller than 2 or larger than 4), so we set the number of clusters for the following tests to 3.

Our first test session consisted in varying HMM parameters to gather some preliminary information on the most promising configuration. Table 3 shows the best performance of the system we achieved by varying HMM parameters: in the left group of columns, *S* is the number of HMM states, *M* is the number of output mixtures and *T* is the minimum log-likelihood threshold used to decide whether a given trajectory matches at least one cluster; on the right columns, for each configuration we show the corresponding detection rate (*DR*, the percentage of correctly identified anomalous trajectories) and the false alarm rate (*FAR*, the percentage of correct trajectories identified as anomalous).

The results we obtained with these first tests showed the difficulty in choosing a set of HMM parameters which provided satisfactory percentages of both true positives and false negatives. The best HMM configuration seemed to be the ones with 15 states and 16 mixtures of Gaussians, and the most discriminating variable was clearly the probability threshold.

In order to understand if the HMM parameters were somehow related to the trajectory length, we carried out a second run of experiments which consisted in building several test sets, each containing the test set trajectories having lengths included in a certain range (for example, $T_{test,\leq 10}$ contained the test trajectories with up to 10 points, $T_{test,11-15}$ contained trajectories with 11 to 15 points, and so on), and finding for each of them the best probability threshold. The HMM's numbers of states and number of output mixtures were 15 and 16, since these values yielded the best results in the previous test. Table 4 shows the best threshold for each range, and the corresponding detection rate and false alarm rate for the each of the new test sets.

It is clear how the trajectory length actually influences the capability of HMMs to match them, and how a length-depending threshold allows to obtain a much higher accuracy than using a fixed threshold.

Length range	*T*	*DR*	*FAR*
$\leq$10	-80	88.8%	4.3%
11-15	-100	80.2%	14.2%
16-20	-160	85.4%	29.2%
21-25	-190	96.2%	16.7%
$\geq$30	-230	82.4%	16.6%

Table 4: Performance of the system when the threshold is varied according to the length of the training and test trajectories.

2.3.2 Rule Based Filtering Mechanism for Anomaly Detection

This module aims at identifying trajectory features which are then used in a subsequent refinement mechanism to filter out the common trajectories (e.g., fish freely swimming), thus providing as output the anomalous ones. In this case, trajectories are defined as sequence of the centroids of fish bounding boxes in consecutive frames. The block diagram of the filtering mechanism is shown in Fig. 3. In this context, first, all fish trajectories are filtered by filter1 (event rule 1). In each step, the trajectories satisfying the rule are defined as normal trajectories (such as Normal1, Normal2). The trajectories which do not satisfy the rule are called the remainders of the corresponding filter and are used as inputs to the next filter. This is continued until all the filters are used. At the end, the remainders of all filters are our abnormal trajectories (which is a set with many fewer normal trajectories). Filters can be applied in any order since the rules of filters are independent.
Primitive motions are defined in two categories: straight and/or cross movements and being stationary. Straight and/or cross movements are defined in three ways: the center of fish bounding boxes over the whole trajectory is inside an area (search area) which is determined by the first detection's bounding box boundary while the fish is going only one direction such as left to right, right to left, up to down and down to up, the center of the fish bounding box in frame $f + i$ is inside an area which is determined by the detection bounding box in frame $f + i - 1$ for $i = 1$ to N (N represents trajectory lengths) while the fish is going only in one direction such as left to right, right to left, up to down and down to up, the center of the fish bounding boxes over whole trajectory are inside an area which is determined by the first and last bounding box boundaries while fish is going only in one direction such as left to right, right to left, up to down and down to up. Being stationary is defined as the state that the center of the fish bounding box is inside an area which is defined in terms of first detection bounding box. Filters are defined as one, two and three length combinations of these primitive motions such as moving left to right (length is one), moving left to right and then being stationary (length is two), moving right to left and then down to up (length is two), being stationary for a while, then moving up to down and then right to left (length is three) etc. Similar behaviors like going left to right and right to left are modeled by same filter and altogether 21 rules were used.

To test the proposed method, 271 underwater videos including 4 different locations and 2370 trajectories (45 abnormal, 2335 normal) belonging to 9 different species were used (see Fig. 4). The normal and unusual behaviors were determined based on visual inspection. In this context, freely swimming fish was considered as normal behavior since this is the most frequent behavior in the data set. The abnormal or rare behaviors were: stationary fish for a long time

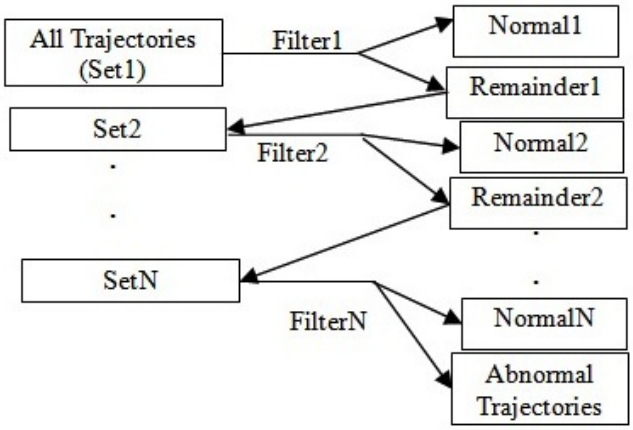


Figure 3: Block diagram of the filtering mechanism for trajectory anomaly detection

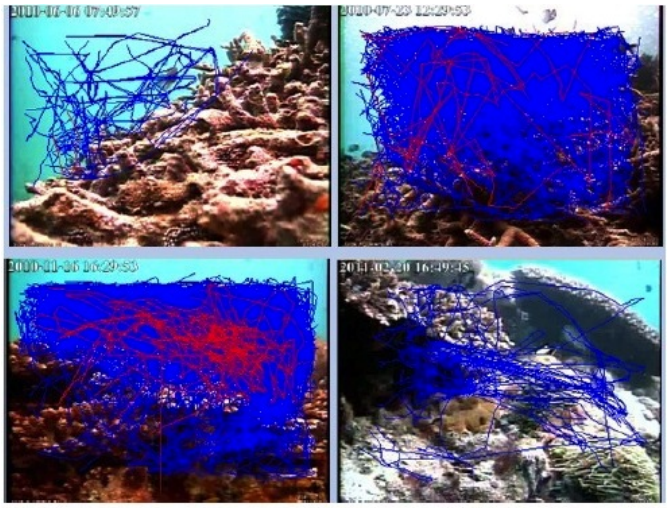


Figure 4: Dataset: normal (blue) and unusual (red) trajectories

inside of coral (this kind of a behavior assumed to be an eating behavior hence differentiated from swimming), fish biting at coral, fish suddenly (in one frame) diving, fish suddenly (in one frame) changing direction. To evaluate the proposed filtering mechanism a 5 fold cross validation test was performed. Train and test sets were constituted randomly while the normal and abnormal trajectories were distributed equally. In the training phase, for each filter the best parameters (search area for straight and/or cross movements, search area for being stationary and using only definition 1, 2, 3, definitions 1 and 2 together, 2 and 3 together, 1, 2 and 3 together) were found and those were used in the test phase. When finding the best parameter values those which did not filter out any abnormal trajectories were chosen. In the case of having more than one parameter set which did not filter out any abnormal trajectories, the one that filtered the most normal trajectories was selected. As a result, this method filtered out nearly half of normal trajectories with 99% precision and 25% of abnormal trajectories which ideally should be zero. However, we believe that this is still a good result since fish species and location implicitly modeled by filters, all abnormalities treated uniformly which makes the proposed method general, agnostic to abnormality type and not data dependent since it has already tested with varying camera parameters and geographic locations.

3 Environment Description

The Fish4Knowledge repository contains about 500.000 videos (10 minute videoclips at different spatial and temporal resolutions) showing different scenes (different cameras, changes of viewpoints, etc..) under different conditions. Automatic analysis of the recorded videos involving scene classification and segmentation has been, therefore, necessary to support both higher analysis levels (e.g. fish behaviour understanding) and underwater monitoring system maintenance. In particular, identifying the components of the analysed scene and classifying the scene itself would allow marine biologists to study how fish interact with the surrounding environment and behave under particular conditions (e.g. during typhoons, storms, etc.). In addition, often video quality is compromised by i) the presence of algae on camera lens (due to the direct contact of seawater with the lens), ii) encoding problems due to transmission bandwidth limit, and iii) atmospheric phenomena (e.g. typhoon), that may displace or disconnect the cameras resulting either in black videos or in recording scenes of no interest. All these cases must be identified and the maintenance service informed as promptly as possible in order to restore the system and keep the information flow.
In fact, each camera videostream depicts many *different scenes* due to camera movements. Environment description is carried out in the F4K scope at different granularity levels according to the nature of camera video-stream (see Fig. 5). In detail, it involves three main steps: 1) video segmentation, which aims at splitting the video-stream of operational cameras into a set of meaningful and manageable segments (scenes) used as basic elements for higher level information (e.g. location of fish congregation), 2) video classification operating on each videoclip which supports live fish behaviour understanding and uncertainty management in the user interface for marine biologists and underwater monitoring system maintenance, and 3) image segmentation which aims at identifying background objects such as rocks, open sea, seabed, etc. While the video classification approach is reliable and ready for production run, video and image segmentation approaches are still under development.

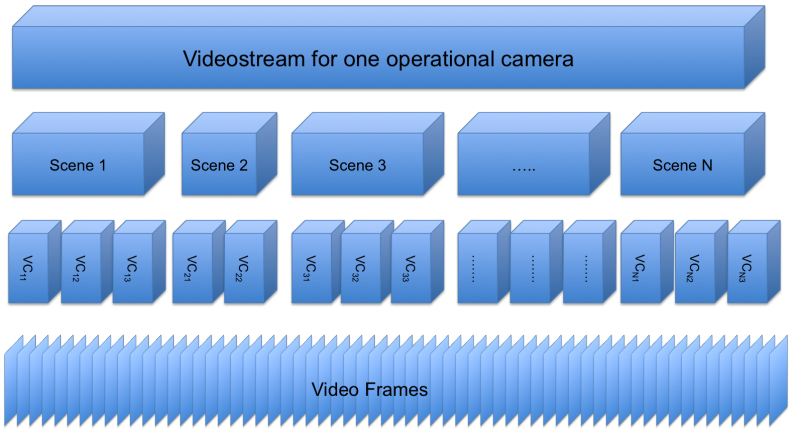


Figure 5: Videostream anatomy

3.1 Video and Image Segmentation

Temporal video segmentation is the first step towards automatic analysis of digital videos for underwater scene understanding and aims at identifying different scenes which correspond to abrupt transitions among consecutive frames due to camera movements. Our scene detection compares the histogram of blocks in consecutive frames to reduce sensitivity to slight camera and object movements.

Our block-based-histogram comparison approach divides the videoframe into $N \times N$ blocks and then compares color distributions of the blocks in consecutive frames. For each block k, we compute the 2-dimensional distribution of I and Q for the YIQ color space, $a*$ and $b*$ for the $L * a * b*$ color space, and hue and chroma components for the Munsell space. The likelihood between the two compared blocks is computed as the weighted average LR of the three following values on the considered spaces:

- $D_1^k(i, i+1) = \sum_{j=1}^{n} |H_i(j) - H_{i+1}(j)|$
- $D_2^k(i, i+1) = 1 - \frac{\sum_{j=1}^{n} min(H_i(j), H_{i+1}(j))}{\sum_{j=1}^{n} max(H_i(j), H_{i+1}(j))}$
- $D_3^k(i, i+1) = \sum_{j=1}^{n} \sum_{k \in N(k)} w(k) \cdot |H_i(j) - H_{i+1}(k)|$

If the likelihood ratio LR between the compared blocks of consecutive frames is smaller than a threshold T_k then the block is labeled. A scene change is identified if the number of labeled blocks is over a threshold T_f. Fig. 6 shows an example of the proposed video segmentation approach, which at the current stage is suboptimal, as demonstrated by the experimental results.

We tested the above approach on 1837 videos of one operational camera (i.e. NPP3 camera 2). We manually labeled 98 scene changes. On this dataset we achieved a precision of 0.803 and a recall of 0.887. This shows that the proposed method performs fairly well in detecting scene changes but it also misclassifies gradual transitions and slight camera movements as scene changes.

Once scene changes are detected, we apply image segmentation to provide a comprehensive description of the background objects. At the current stage, we adopted the image segmentation method proposed by Borsch *et al.* in [15], extending the feature extraction phase by adding

Figure 6: Example of Video segmentation

SIFT [16] object corners since they proved to be reliable descriptors.
We tested the image segmentation approach on 79 images (manually segmented) and 5 object classes: “Sea”, “Rock”, “Coral”, “Sea Bed” and “Unknown” and the results are shown in Table 5.

	Sea	Rock	Coral	Sea Bed	Unknown
Sea	**97.32%**	0.37%	0.93%	0%	2.55%
Rock	0%	**79.55%**	9.12%	4.59%	6.74%
Coral	0%	3.72%	**86.70%**	3.04%	6.54%
Sea Bed	0%	13.23%	13.97%	**69.52%**	3.28%
Unknown	4.21%	1.87%	1.59%	1.21%	**91.12%**

Table 5: Confusion matrix of the adopted image segmentation approach

3.2 Video Classification

The strategy devised for scene classification resorts to the Bag of Features (BoF) approach [17] and works at video level, i.e., given an input video, a model vector sequence is obtained and its classification as a specific scene class is carried out by means of SVM operating on the extracted descriptors.
The employed features extraction method computes for each video a set of features on a subset of video frames and the flowchart is shown in Fig.7. More specifically, for each frame, we compute non-rotation-invariant SIFT descriptors [16] of image corners detected with the Harris approach; then the image is described as the histogram of the squared distances between the set of computed features and the centroids of descriptor clusters previously computed on the specific scene classes.

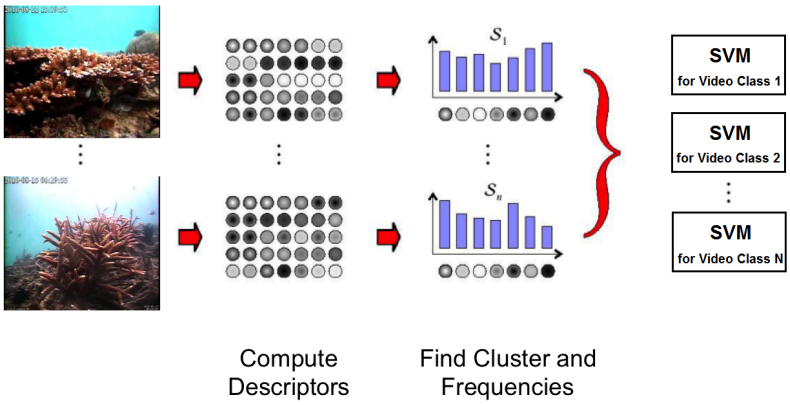


Figure 7: Flowchart of the BoF approach for video classification

The video classification is carried out by grabbing a subset of frames (from 15 to 100) from the video under analysis, extracting the BoF and then feeding it to a set of SVMs, previously

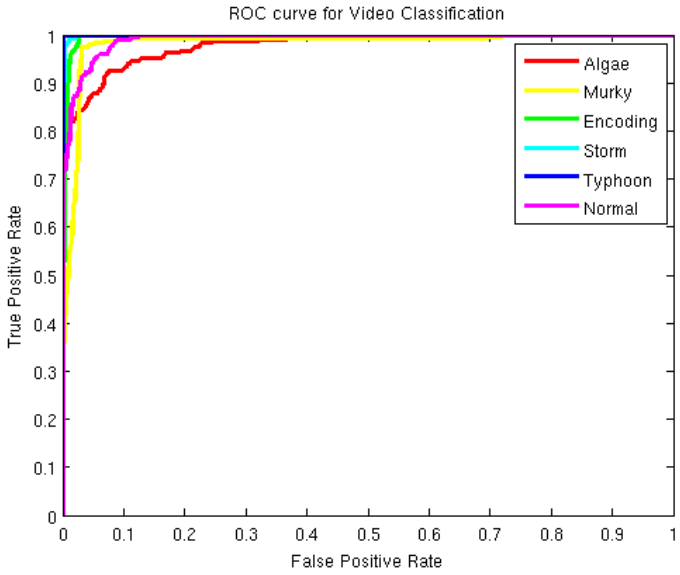


Figure 8: ROC Curves per Video class

Video Class	Average Correct Rate	Standard Deviation
Algae	0.996	0.0008
Murky	0.968	0.0032
Encoding	0.987	0.0016
Storm	0.982	0.003
Typhoon	1	0
Normal	0.998	0.001

Table 6: Average Classification Performance per Video class

trained on the scene classes we want to identify (one SVM for one scene class). We tested our approach on a set of 102 videos (20 frames per video, 2040 frames), manually labeled into 6 classes: “Normal”, “Encoding Problem”,“Murky Water”, “Algae on Camera Lens”, “Typhoon” and “Storm”, with a 5-fold cross validation procedure and we achieved an average performance in terms of correct rate of 91%. Table 6 shows the achieved results for each video class, and Fig. 8 shows the ROC curves per video class.

4 Conclusions

In this deliverable we described the work done for fish and environment property description showing also for each set of employed features its utility in supporting a specific image analysis task. For fish description, the features adopted for fish/background objects classification and fish species recognition are quite reliable and they were extensively tested on large datasets, showing promising results. Fish behaviour understanding, instead, is still at a preliminary stage since it was applied to a limited set of behaviours (solitary and pairing for a few species) due to the difficulty to gather ground truth data and to the sensitivity of the HMM based approach to

the fish trajectory length.
Finally, we also investigated techniques for environment description which rely on video segmentation to identify scene changes, scene classification to understand how the recorded scenes change over time due to the environmental factors and scene segmentation which aim at identifying scene areas such as rocks, open sea, etc. The scene classification approach is ready for production run, whereas the approaches for temporal video segmentation and image segmentation are suboptimal and must be improved. In detail, we are also considering to adopt HMM to model scene changes thus avoiding using an extrinsic scene representation of frame changes which is too sensitive to the used thresholds. For image segmentation, we will follow the approach proposed by Bosch *et al.* in [15] suitably extended with new set of features and to adopt Markov Random Fields (MRF) spatio-temporal regularisation [18] to refine the output classification. Of course, new approaches will be also developed in order to investigate efficiency gains.

References

[1] Z. Liu, D. W. Jacobs, and R. Basri. The role of convexity in perceptual completion: beyond good continuation. *Vision Res*, 39(25):4244–4257, 1999.

[2] Chang Cheng, Andreas Koschan, Chung-Hao Chen, David L. Page, and Mongi A. Abidi. Outdoor scene image segmentation based on background recognition and perceptual organization. *IEEE Trans. on Image Processing*, 21(3):1007–1019, 2012.

[3] P. Felzenszwalb and D. Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision*, 59(2):167–181, 2004.

[4] Bogdan Alexe, Thomas Deselaers, and Vittorio Ferrari. Measuring the objectness of image windows. *IEEE Transactions on PAMI*, 99(PrePrints), 2012.

[5] Jean-Yves Bouguet. Pyramidal Implementation of the Lucas Kanade Feature Tracker Description of the algorithm. Technical report, Intel Corporation Microprocessor Research Labs, 2000.

[6] C. Spampinato and S. Palazzo. Enhancing object detection performance by integrating motion objectness and perceptual organization. In *To appear on Proc. of the 21st International Conference on Pattern Recognition (ICPR'12)*, 2012.

[7] C. Spampinato, D. Giordano, R. Di Salvo, Y. H.J Chen-Burger, R. B. Fisher, and G. Nadarajan. Automatic fish classification for underwater species behavior understanding. In *Proceedings of the first ACM international workshop on analysis and retrieval of tracked events and motion in imagery streams*, page 45–50, New York, NY, USA, 2010.

[8] Ronald Poppe. A survey on vision-based human action recognition. *Image and Vision Computing*, 28(6):976 – 990, 2010.

[9] H. Jhuang, E. Garrote, J. Mutch, X. Yu, V. Khilnani, T. Poggio, A. D. Steele, and T. Serre. Automated home-cage behavioural phenotyping of mice. *Nat Commun*, 1:68, 2010.

[10] L R Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[11] Naohiko Suzuki, Kosuke Hirasawa, Kenichi Tanaka, Yoshinori Kobayashi, Yoichi Sato, and Yozo Fujino. Learning motion patterns and anomaly detection by Human trajectory analysis. In *Systems, Man and Cybernetics, 2007. ISIC. IEEE International Conference on*, pages 498–503. Ieee, 2007.

[12] G Young and A S Householder. Discussion of a set of points in terms of their mutual distances. *Psychometrika*, 3(1):19–22, 1938.

[13] B. H. Juang and L. R. Rabiner. A probabilistic distance measure for hidden Markov models. *AT&T Technical Journal*, 64(2):391–408, 1985.

[14] T Calinski and J Harabasz. A dendrite method for cluster analysis. *Communications in Statistics Theory and Methods*, 3(1):1–27, 1974.

[15] A. Bosch, X. Munoz, and J. Freixenet. Segmentation and description of natural outdoor scenes. *Image and Vision Computing*, 25(5):727 – 740, 2007.

[16] David G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2):91–110, November 2004.

[17] Eric Nowak, Frdric Jurie, and Bill Triggs. Sampling strategies for bag-of-features image classification. In Ale Leonardis, Horst Bischof, and Axel Pinz, editors, *Computer Vision ECCV 2006*, volume 3954 of *Lecture Notes in Computer Science*, pages 490–503. Springer Berlin / Heidelberg, 2006.

[18] L. Carminati and J. Benois-Pineau. Gaussian mixture classification for moving object detection in video surveillance environment. In *Image Processing, 2005. ICIP 2005. IEEE International Conference on*, volume 3, pages III – 113–16, sept. 2005.

