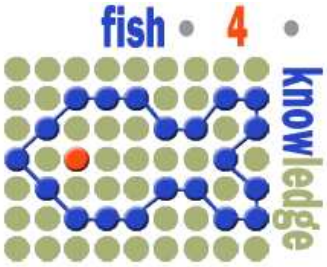

Fish4Knowledge Deliverable D3.1

Goal, Video Description and Capability Ontologies and Process Library

Principal Author: Gaya Nadarajan, Jessica Chen-Burger
Contributors: UEDIN Workflow
Dissemination: PU

Abstract: In this report, we describe our efforts in creating a set of suitable domain ontologies that are based on user requirements (from marine biologists) for our intelligent workflow system, as defined in the project proposal. The main purposes of these ontologies are (1) to support the development of appropriate functions of the workflow system, and (2) to serve as a communication media to interface with other Fish4Knowledge components. Three ontologies were designed with close collaboration with image processing experts and communications with marine biologists and user interface experts to capture the domain knowledge succinctly – **Goal, Video Description** and **Capability** ontologies. In addition, a process library was developed to describe the processes within the system. The ontologies and process library were utilised in the first version of our workflow composition and execution system to support video classification, fish detection and counting tasks. The ontologies will continue to evolve with the project's needs and are envisaged to interface with other Fish4Knowledge components in due course.

Deliverable due: Month 8

Contents

1 Introduction

The workflow component of the F4K project is responsible for the composition and execution of a set of video and image processing (VIP) modules on high performance computing (HPC) machines based on user requirements and descriptions of the video data. In order to do that, three types of knowledge are required for the design of the workflow component: (1) User requirements that can be translated into high-level VIP tasks; (2) Description of the video data available to us; and (3) The VIP resources available to us. The first knowledge is acquired through correspondence with F4K's marine biologists in Taiwan, while the second and third knowledge are acquired from the image processing experts in F4K (Edinburgh University and University Catania teams). As a result, the workflow component interprets the user requirements (from User Interface team) as high level VIP tasks, create workflows based on the procedural constraints of the modules (provided by IP teams) to ultimately invoke and manage their execution in a distributed environment (HPC team).

In our intelligent workflow system, we have chosen to use an ontological-based approach to guide the automatic generation of a "virtual workflow machine" based on a set of closely related ontologies. This allows a separation between the problem and application descriptions and the workflow mechanism. As a result, the workflow machine may work in different problem domains if the problem and application descriptions are changed. Using ontologies is an accepted methodology for describing domain knowledge and for capturing knowledge and semantics in a domain that has been used widely in several major fields including medical, linguistic and enterprise. In addition, we have also used these ontologies as a communication medium that enables us to convey and verify our understanding with the domain experts, *i.e.* in the field of marine biology and video and image processing.

Consequently, this will promote reusability and provide a conceptualisation that can be used between different domain experts, such as marine biologists, image processing experts and workflow engineers. These ontologies are also pivotal for reasoning. For instance, in the selection of optimal VIP software modules, the Capability Ontology is used to record known heuristics that we have obtained from VIP domain experts. Ontologies may also be used to guide the recording of better performing user experiences, thus improving the performance of the workflow system over time.

In this report, we therefore describe our efforts in creating a set of suitable domain ontologies that are based on user requirements (from marine biologists) for our intelligent workflow system, as defined in the project proposal. The main purposes of these ontologies are (1) to support the development of appropriate functions of the workflow system, and (2) to serve as a communication media to interface with other Fish4Knowledge components.

As a result, we first describe the 20 scientific questions as provided by the marine biologists. We then provide an analysis of these 20 questions from the workflow system's point of view - that is based on the capabilities of video and image processing (VIP) modules available to us. Based on a mapping between the user requirements and a high level abstraction of the capabilities of the VIP modules, we have constructed the Goal Ontology (Section 3). The Video Description Ontology (Section 4) contains the environmental factors related the videos. The Capability Ontology (Section 5) describes details of the VIP modules, *e.g.* heuristics on their strengths and weaknesses. To date, the Goal Ontology contains 52 classes, 85 instances and 1 property, the Video Description Ontology has 24 classes, 30 instances and 4 properties and the Capability Ontology has been populated with 42 classes, 71 instances and 2 properties.

The process library (Section 6) defines the procedures that can be used to call upon VIP modules. Currently, we have not found the use of any other domain ontologies. If such ontologies are found to be useful at a later stage, they will be created.

The set of ontologies and process library developed have supported the composition and execution of the first version of the workflow system that has been evaluated for video classification according to brightness, clearness and algal levels, fish detection and counting tasks [2, 3]. As the Fish4Knowledge project is an on-going project that user requirements and functionalities of the components may change over time, the reported ontologies and process library here are therefore the current state-of-the-art for the project.

2 Principles and Focus

Ontology development is often lengthy and requires repeated discussions between experts from different domains to reach consensus. An ontology should be non-ambiguous, correct, intuitive and concise. The main components of an ontology are classes, the relationships between these classes and the attributes and instances of these classes. Classes describe concepts in the domain and are often the focus of an ontology. They are typically a group of things that share common attributes. A class can have subclasses that represent concepts that are a specialisation the superclass. Instances are actual instantiations of classes. (Object) properties describe the relationships between classes; where as (data) properties describe the attributes of a class.

The ontologies developed for this deliverable are aimed to support workflow composition and management. That is, given a high level VIP task, such as a fish detection query, it should be able to find the sequence of optimal VIP tools to solve that task. For this purpose three ontologies have been constructed to reflect the main aspects that capture workflow composition for video processing tasks. That is 1) the **Goal** part that specifies and defines the VIP task; 2) the **Video Description** part, which describes environmental and other conditions pertaining to the videos; and 3) the VIP tools and techniques **(Capabilities)** available for use. The ontologies were developed in consultation with the Workflow, Image Processing and User Interface teams. Since the purpose of the ontologies is to help construct optimal workflow solutions for VIP tasks, close collaborations were conducted with the VIP teams in order to get insights of the VIP modules' operational details.

For ontology development and visualisation purposes, OWL 1.0 [1] was generated using Protege version 4.0. Where applicable, ontological diagrams were derived using the OntoViz plugin. Within Protege, classes, subclasses, sibling classes, instances and properties (relationships) can be created, modified and deleted wih ease. Furthermore, rules and axioms can be asserted. Annotations such as description, source and comments can be defined for each entity. The ontology created can also be verified for consistency. The next three sections will elaborate the ontologies.

3 Goal Ontology

The Goal Ontology contains the high level questions posed by the user interpreted by the system as VIP tasks, termed as goals, and the constraints to the goals. Before the Goal Ontology could be constructed, a list of queries posed by marine biologists were examined. These questions are listed in Table 1 and can be found in Deliverable 2.1 – User Information Needs.

Table 1: List of 20 scientific questions posed by Taiwanese marine biologists.

Q1	How many species appears and their abundance and body size in day and night including sunrise and sunset period.
Q2	How many species appears and their abundance and body size in certain period of time (day, week, month, season or year). Species composition change within one period.
Q3	Give the rank of above species, *i.e.* listed according to their abundance or dominance. How many percent are dominant (abundant), common, occasional and rare species.
Q4	Fish color pattern change and their behavior in the night for diurnal fish and vice versa for nocturnal fishes.
Q5	Fish activity with one day (24 hours).
Q6	Feeding, predator-prey, territorial, reproduction (mating, spawning or nursing) or other social or interaction behavior of various species.
Q7	Growth rate of certain species for a certain colony or group of observed fishes.
Q8	Population size change for certain species with one period of time.
Q9	The relationship of above population size change or species composition change with environmental factors, such as turbidity, current velocity, water temperature, salinity, typhoon, surge or wave, pollution or other human impact or disturbance *etc*.
Q10	Immigration or emigration rate of one group of fishes inside one monitoring station or one coral head.
Q11	Solitary, pairing or schooling behavior of fishes.
Q12	Settle down time or recruitment season, body size and abundance for various fishes.
Q13	In certain area or geographical region, how many species could be identified or recognized easily and how many species are difficult. The most important diagnostic character to distinguish some similar or sibling species.
Q14	Association among different fish species or fish-invertebrates.
Q15	Short term, mid-term or long term fish assemblage fluctuation at one monitoring station or comparison between experimental and control (MPA) station.
Q16	Comparison of the different study result between using diving observation or underwater real time video monitoring techniques. Or the advantage and disadvantage of using this new technique.
Q17	The difference of using different camera lens and their angle width.
Q18	Is it possible to do the same monitoring in the evening time.
Q19	How to clean the lens and solve the biofouling problem.
Q20	Hardware and information technique problem and the possible improvement based on current technology development and how much cost they are.

These 20 questions were examined carefully to understand their meaning in VIP terms. Lengthy discussions were also held with the F4K user query team and the marine biologists. Then these questions were mapped to the relevant video and image processing (VIP) tasks that represent them. These were derived based on discussions with the image processing teams. Eight main task types have been identified and listed below.

1. Fish Detection and Tracking.
   One of the most important tasks involves determining the number of fish in a range of videos over a period of time for statistical analysis. In order to do this, fish objects will

need to be identified (detected) and then tracked frame by frame in order for them to be counted later on. This is a low level image processing task that will contribute towards other analysis such as (fish) population-related queries and high-level fish information aggregated analysis.

2. Fish Species Classification.
   This task involves classifying a fish object into its relevant species. Species identification is one of the main tasks that is needed to meet the requirements laid out in the 20 questions. Out of the 3000 species of fish live in the Taiwanese sea [5], about 1500 species can be observed by camera surveillance. Further to this, a coarse clustering result has yielded about 27 species of fish from a 30,000 frames database that F4K image processing experts will aim to classify.

3. Population-Related Analysis.
   Marine biologists are interested in knowing statistical information on the population size, species composition, immigration and emigration rates of a particular group of fish. The core task here would involve fish counting, which builds on the results from fish detection and tracking.

4. Behaviour Understanding.
   One of the main interests of marine biologists is to understand the behaviours of various fish species. Behaviours can be related to fish activity, such as feeding and reproduction, or related to population, such as growth rate and species composition change or related to other behaviours, such as colour pattern change.

5. Fish Clustering.
   Marine biologists are also interested in finding similar types of fish and the association between them. Fish clustering would identify fish that look similar according to some common features (such as shape, colour, *etc*.) to allow for such analysis to be carried out.

6. Fish Feature Analysis: colour, contour, texture, size.
   Fish descriptions such as the size and colour of the fish, are of interest for marine biologists. This task can be used to answer queries that can provide descriptions of fishes but not identify which fish it is; or the other way around. For example, fish features such as colour, contour and texture are generally computed to be used for fish species recognition.

7. Event Detection.
   This task involves the identification and localisation of specified spatio-temporal patterns (events) in video, such as fish mating and preying activites. The events could also include environmental incidences such as typhoon, accumulation of algae on the camera lenses and other factors that may influence the quality of the recorded videos.

8. Aggregated Analysis.
   Often, conducting the tasks described above separately are not sufficient to perform the analysis required to answer marine biologists' queries. Their results will have to be aggregated for further analysis. For instance, ranking of fish species would involve fish detection, counting and species recognition, followed by the sorting of the number of fish according to species. Other higher-level analysis would include those that do not

involve any image processing, such as comparisons between different methodologies used to solve VIP tasks, effects of using different types of lenses, *etc*. Query 19 and 20 are examples of such type of tasks.

As a starting point, the 20 questions provided by the Taiwanese marine biologists were analysed. These are indicated as Q1 to Q20 in Table 2. They were then mapped to the VIP tasks (goals) that they fall into based on the types of tasks that are required to be carried out to address these questions. A VIP task as indicated here is a high level goal in our Goal Ontology that can be understood by an image processing expert who can write low level programs or software to solve it. The following table contains the result of the analysis after communication with image processing experts, user interface experts and marine biologists.

Table 2: Mapping of high-level VIP tasks to the 20 questions as provided by marine biologists. Legend: MB – marine biologists, IP – image processing experts, UI – user interface team, QE – query engine, NCHC – high performance computing group.

Query/Tasks	Fish Detection and Tracking	Population Related Analysis	Fish Species Classification	Behaviour Understanding	Fish Clustering	Fish Feature Analysis: colour, contour, texture, size	Event Detection	Aggregated Analysis (QE–CWI)	Assumptions, Definitions and Comments	Attention
1	x	x	x			x			Assumption: Sunrise & sunset are defined by IP teams as a 10-minute period when videos change from dark to colour & vice-versa. Night videos are not available.	MB, UI
2	x	x	x					x	Definition: *Species composition* is defined as the number of fish from each species.	
3	x	x	x	(x)		x		x		
4	x	x	x	x				x	Comment: *Diurnal* and *nocturnal* fishes will need to be determined manually by F4K researchers and populated in the database. Colour pattern change may not be possible to be performed as the change will occur within 1–2 seconds, which is too quick for current VIP techniques to detect.	MB, IP, QE
5	x	x	x	x		x			Assumption: Activity refers to activity of *all* the fish in a video clip.	IP,QE
6	x	x	x	x		x		(x)	Comment: Territorial behaviour is seen when a fish species attack other fish or human entering the area that they are in.	MB

Continued on next page

Table 2 – Continued

Query/Tasks	Fish Detection and Tracking	Population Related Analysis	Fish Species Classification	Behaviour Understanding	Fish Clustering	Fish Feature Analysis: colour, contour, texture, size	Event Detection	Aggregated Analysis (QE–CWI)	Assumptions, Definitions and Comments	Attention
7	x	x	x			x		x	Definition: *Fish group* is a group of fish (normally of the same or closely related species) living together in one area. Definition: *Fish colony* refers to the same fish species. Definition: Growth rate = birth rate – death rate + immigration rate – emigration rate	MB, UI
8	x	x	x		(x)	x	(x)	x	Assumption: *Population size change* depends on (not exclusively, but most commonly) immigration, emigration, birth rate and death rate.	MB
9	x	x	x	(x)		x	x	x	Comment: NCHC could provide the following information from its equipment and governmental archives: water salinity, temperature, turbidity, ph value and dissoved oxygen. In the future typhoon incidence is a possibility.	MB, NCHC
10	x	x	(x)			(x)		x	Comment: Mathematical formula for *immigration* and *emigration rate* are required for aggregated analysis.	MB, UI
11	x	x	x	x		x				
12	x	x	x			x		x	Definition: *Settle down time* refers to the time it takes to colonise a new area through immigration. In marine terms, settlement refers to the colonisation of a previously unavailable area, *e.g.* new structure (sunken ship) or an area that has been cleared (usually by a destructive event, *e.g.* storms, human action). Definition: *Recruitment season* is the phenomenon where new born fishes survive and are added to the overall population.	MB, UI
13	x	x	x		x	x		x	Comment: Similar/sibling species could be physically similar (*e.g.* shape, size, number of spikes), or having the same genus but not species. Comment: This task requires labelled data, *i.e.* ground truth in order to determine whether a fish species is easy or difficult to be recognised.	QE, IP
14	x	x	x	x	(x)	x		(x)	Definition: *Association* refers to the co-occurrence of species. It could be considered with the number of co-occurrences per associated species, average	MB, UI

Continued on next page

Table 2 – Continued

Query/Tasks	Fish Detection and Tracking	Population Related Analysis	Fish Species Classification	Behaviour Understanding	Fish Clustering	Fish Feature Analysis: colour, contour, texture, size	Event Detection	Aggregated Analysis (QE–CWI)	Assumptions, Definitions and Comments	Attention
									time-span and threshold of time-span between 2 occurrences. An example is the symbiotic relationship between fish and fish-invertebrates.	
15	x	x	x	(x)		x		x	Definition: A *fish assemblage* is a group of interacting populations within a specific area. A population refers to a single species, thus an assemblage refers at least 2 different species within a specific area. They must interact *e.g.* predator/prey interactions,compete for food, space or other resources. Fish assemblage fluctuation is the change within the structure of the assemblage, *e.g.* increase/decrease in numbers either overall or within individual populations. Assumption: 'Short-term', 'mid-term' and 'long-term' durations are to be defined by the user.	MB, UI
16								x	Comment: This task requires results from manual processing in order for comparisons to be performed.	MB, QE
17								x	Comment: This is not possible at present.	MB
18								x	Comment: This is not possible at present unless infra-red cameras are installed and recordings are provided.	MB, NCHC
19							x	x	Comment: This question should be rephrased as 'When to clean the lens'. How to clean the lens and solve the bio-fouling problem is beyond F4K's IP capabilites.	MB, IP, UI
20								x	Comment: This is not possible at present.	NCHC

Table 2 shows the VIP tasks for the 20 questions posed by marine biologists involved in Fish4Knowledge. The tasks were identified via discussions with IP experts (UEDIN IP and UCATANIA). For each question the relevant VIP task(s) that can be used to support it are marked with an 'x'. An 'x' marked in parentheses indicate that the particular VIP task could be associated with the question, but it could not be determined with absolute certainty that this function will be supported within this project. This may be due to technical reasons or the time constraints imposed on this project. The first seven tasks are directly related to video and image processing components while the final task 'Aggregated Analysis' is related to analysis that makes use of results generated by other VIP tasks. Where applicable, assumptions, definitions and comments are included in the table for the relevant F4K teams (image processing, query

engine, marine biologists, user interface or high performance computing). The VIP tasks identified here constitute the main goals that we want to describe and use in the Goal Ontology.

Along with these main goals, we add higher level goals to provide a classification structure and linkage to possible user queries, the Goal Ontology also includes the constraints to further specify the goal, shown in the sub-tree 'Constraint on Goal'. Figure 1 depicts the Goal Ontology pictorially. The main relationship between classes shown here is the sub-class relationship, *is-a*. Instances and properties are not shown due to the limitation of the visualisation tool. They are provided in detail in Appendix A.

The 'Goal' class is the umbrella of concept that includes the VIP tasks identified in Table 2. The eight main VIP tasks can be found in Figure 1. We also created a level of intermediate classes between the goal class and the lower level VIP goals to increase the flexibility and readability of the ontology. This set of intermediate classes are 'Object Detection', 'Object Tracking' (object detection and tracking are separated in the ontology for ease of IP processing), 'Population Related Analysis', 'Classification', 'Behaviour Understanding', 'Object Clustering', 'Feature Related Analysis', 'Event Detection' and 'Aggregated Analysis'. Three additional classes 'Display Task', 'Image Video Compression' and 'Image Video Segmentation' are included as they are useful general purpose VIP tasks that we may need. Under these general concepts, more specific goals may be defined, for example 'Fish Detection', 'Fish Tracking', 'Fish Clustering', 'Fish Species Classification' and 'Fish Size Analysis'. The principle behind keeping the top level concepts more general is also to allow the ontology to be easily extended to include other (new) tasks as appropriate as the project develops.

'Fish Detection' and 'Fish Tracking' tasks are needed for most of the 20 questions as fish detection and tracking will be needed for counting tasks. They are kept separate for extensibility and modularity. Other objects may also be detected, for example coral, fish-invertebrates and non-fish organisms, which would fall under the class 'Object Detection'. These are defined in the ontology following the hierarchy shown in Figure 2.

Using a similar principle, the classification of various fish species and other objects can be specified under the class 'Object Classification'. Its parent, 'Classification' refers to the general task of classifying videos and objects in the videos. Classifying the video according to its algal (green tone) levels, for instance, could be used to answer Question 19.

'Population Related Analysis', as its name suggests, concerns with queries regarding the population of a group or colony of fish. 'Object Counting' has been regarded as a population-related analysis as all the queries involving counting was related to population, hence it was kept separate from tracking.

Behaviour describes the changing of features and activities that is observable externally. Various fish behaviours are defined under the class 'Behaviour Understanding'. Two main types of behaviour are activity-based ones and feature-based ones. Activity-based behaviours include feeding, pairing, predation, prey escaping behaviour, reproducing, schooling, solitary behaviour, territorial behaviour and association with other fish sprecies or fish-invertebrates. These are instances of the class 'Activity Based Behaviour'. For now two instances of 'Feature Based Behaviour' have been indentified; colour pattern change in diurnal fish and colour pattern change in nocturnal fish.

Event detection is primarily concerned with the detection of changes in the current environment. In the ontology, its instances include factors such as high current velocity, high level of algae on camera, pollution incident, surge or wave, typhoon, high or low (unusual) water salinity levels, unusual water temperatures and unusual water turbidity levels. The detection of

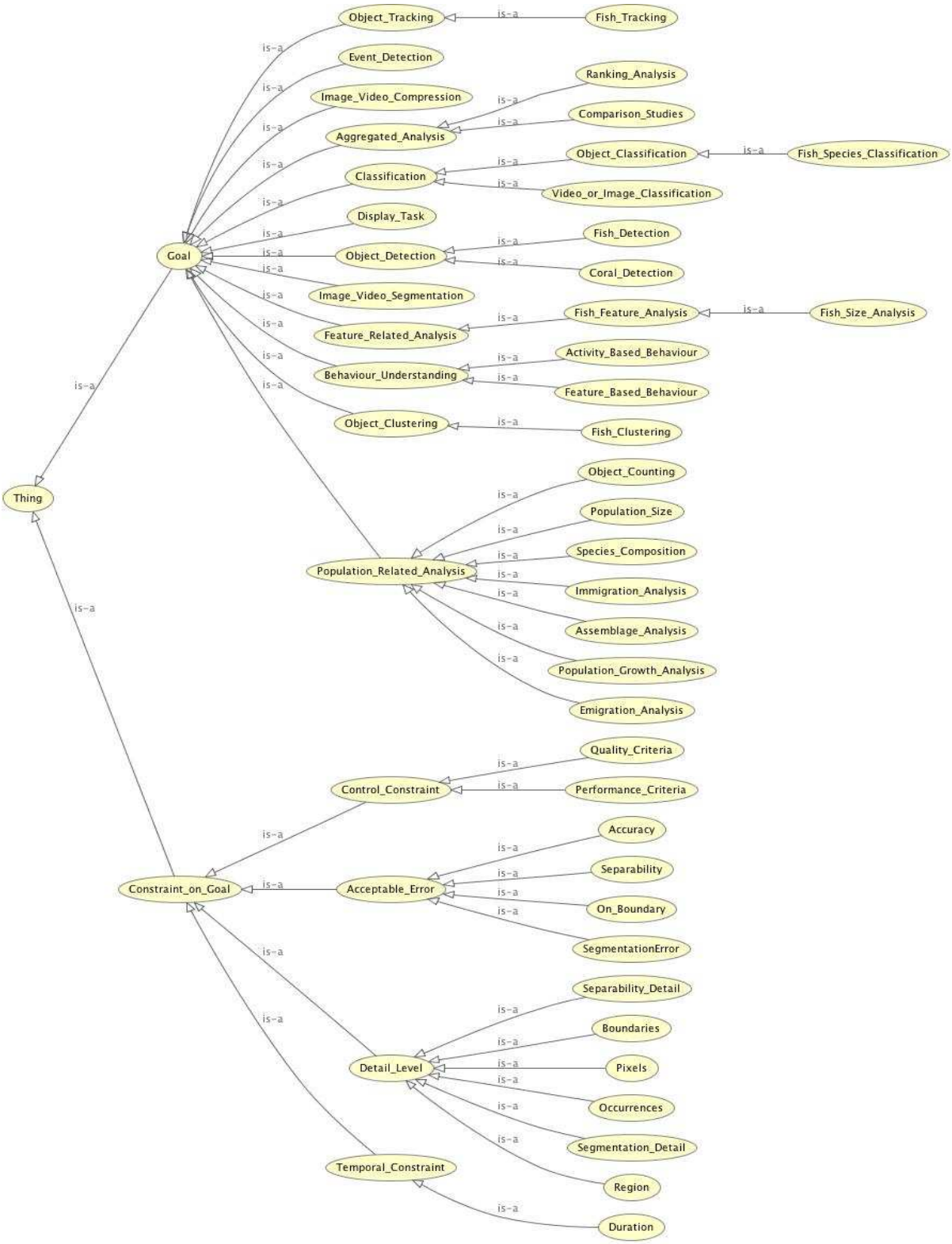

Figure 1: Goal ontology denoting the main classes of goals and constraints that were applicable to the 20 questions posed by Taiwanese marine biologists.

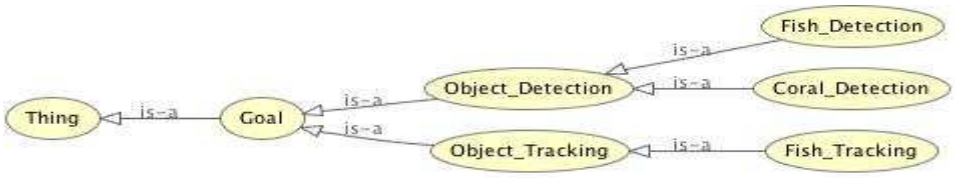

Figure 2: Two main types of goals for F4K – Fish detection and fish tracking.

an event will lead to computation of the changes in population sizes or species compositions (related to Question 9).

Finally, tasks that fall under the class 'Aggregated Analysis' are those that cannot be performed by a single VIP task alone, but those that require further aggregation and combination of results from several other VIP tasks. These include ranking tasks, comparion of results using different techniques (*e.g.* automatic versus manual), and so on. In essence most tasks will involve some form of aggregation as shown in Table 2.

'Constraint on Goal' refers to the conditions that restrict the video and image processing tasks or goals further. In F4K's context, the main constriction for a VIP goal is the 'Duration', a subclass of 'Temporal Constraint'. Each task may be performed on all the historical videos, or a portion specified by the user – within a day, night, week, month, year, season, sunrise or sunset (all specified as instances of the class 'Duration').

Other constraints types include 'Control Constraint', 'Acceptable Error' and 'Detail Level'. The control constraints are those related to the speed of VIP processing and the quality of the results expected by the user. 'Performance Criteria' allows the user to state whether the goal that they wish to perform should be executed using a faster algorithm (indicated by the criterion processing time) or whether it should take less (CPU) memory. Processing time and memory are instances of 'Performance Criteria'. Instances of the class 'Quality Criteria' are reliability and robustness. 'Quality Criteria' with the value reliability constrains the solution to be the most accurate result. If such a solution could not be found, then the system should fail rather than produce alternative options. 'Robustness' indicates the reverse; that the system should not break down completely in cases where a reliable solution could not be found, instead it should return an alternative (imperfect) result.

Another constraint that the user may want to insert is the threshold for errors, contained in the class 'Acceptable Error'. A typical example of this is contained in its sublass 'Accuracy', which states the accuracy level of a detected object. Two instances of accuracy are prefer miss than false alarm' and prefer false alarm than miss. Miss and false alarm are terminologies used within VIP tasks that involve the detection of objects to indicate the accuracy level of the detection. Consider a real object to be the object that needs to be detected. A miss (false negative) occurs when a real object exists but is not detected. A false alarm (false positive) occurs when an object that is not a real object has been detected.

The class 'Detail Level' contains constraints that are specific to particular details, for example detail of 'Occurrence'. The criteria for 'Occurrence' is used for detection tasks to constrict the number of objects to be detected. The value 'all' for occurrences imposes that all the objects should be identified.

The Goal Ontology is used for consistency checks when a user query is detected in the system. It can check that the query matches with a goal or set of goals that is achievable within the workflow system. It is also used to guide the selection of higher level tasks for workflow and

formulate input values to the reasoning engine that is responsible for searching the VIP solution set for a VIP task, *i.e.* to compose the workflow.

4 Video Description Ontology

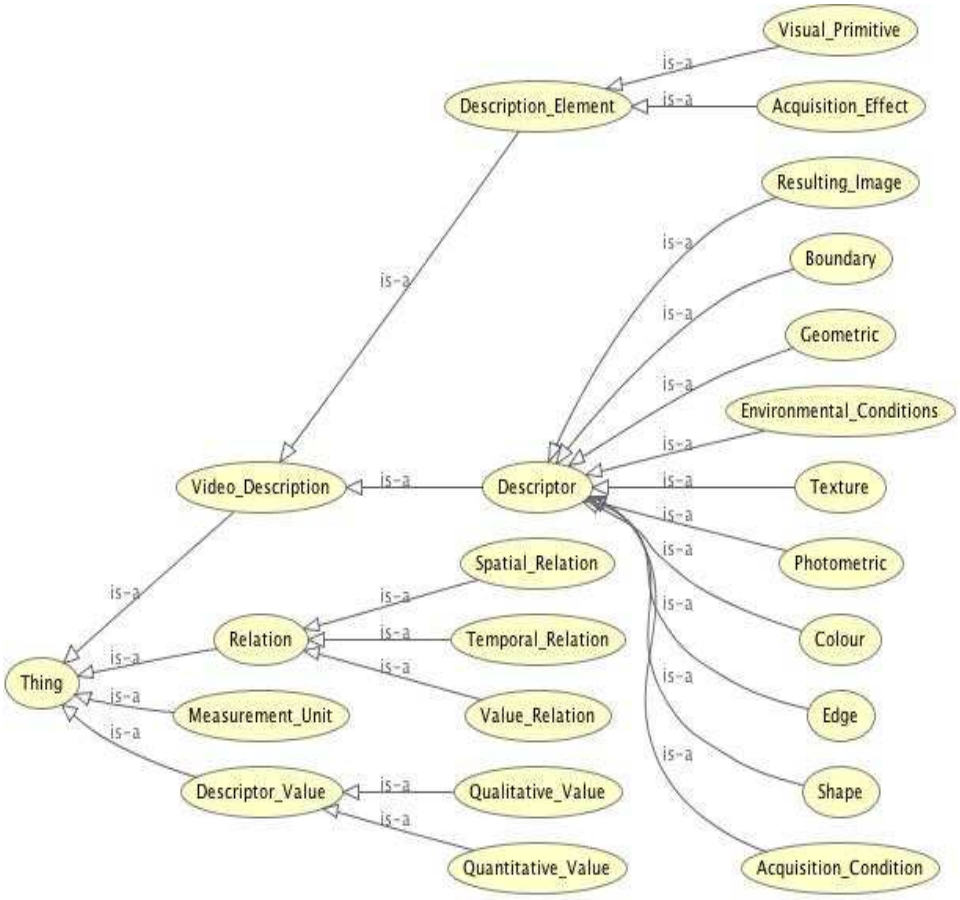

Figure 3: Main concepts of the Video Description Ontology.

The Video Description Ontology describes the concepts and relationships of the video and image data, such as what constitutes video/image data, the acquisition conditions such as lighting conditions, colour information, texture, **environmental conditions** as well as spatial relations and the range and type of their values. Fig. 3 gives a pictorial overview of the main components of the Video Description Ontology. The upper level classes include 'Video Description', 'Descriptor Value', 'Relation', and 'Measurement Unit'.

The main class of this ontology is the 'Video Description' class, which has two subclasses – 'Description Element' and 'Descriptor'. A description element can be either a 'Visual Primitive' or an 'Acquisition Effect'. A visual primitive describes visual effects of a video/image such as observed object's geometric and shape features, *e.g.* size, position and orientation while acquisition effect descriptor contains the non-visual effects of the whole video/image that contains the video/image class such as the brightness (luminosity), hue and noise conditions.

The descriptor for the description elements are contained under the ‘Descriptor’ class and are connected to the ‘Description Element’ class via the object property ‘hasDescriptionElement’ (not visible in the diagram but contained in Appendix B).

Typical descriptors include shape, edge, colour, texture and environmental conditions. Environmental conditions, which are acquisitional effects, include factors such as current velocity, pollution level, water salinity, surge or wave, water turbidity, water temperature and typhoon, specified as instances. Figure 4 shows how these instances are represented in the ontology.

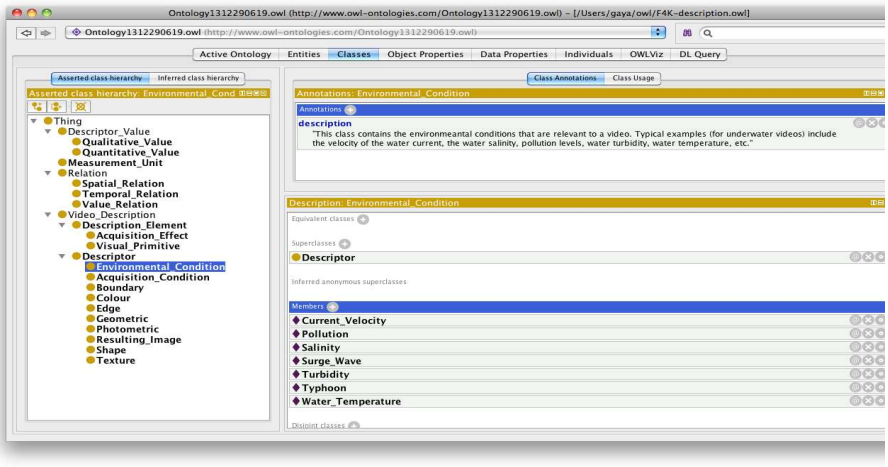

Figure 4: Environmental conditions are described as members (instances) of the ‘Environmental Condition’ class in the Video Description ontology.

Figure 5 below shows other instances (known as members or individuals in Protege) in the Video Decription ontology.

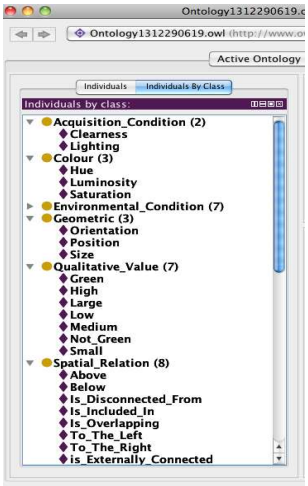

Figure 5: Environmental conditions are described as members (instances) of the ‘Environmental Condition’ class in the Video Description ontology.

These values that the descriptors can hold are specified in the ‘Descriptor Value’ class and connected by the object property ‘hasValue’. For the most part, qualitative values such as ‘low’, ‘medium’ and ‘high’ are preferred to quantitative ones (*e.g.* numerical values). ‘Qualitative’

values could be transformed to quantitative values using the 'convertTo' relation. This would require the specific measurement unit derived from one of the classes under the concept 'Measurement Unit' and conversion function for the respective descriptor *e.g.* a low velocity could be interpreted as movement with velocity within a range of 0 and 25ms^{-}1[1]. Some descriptor values can be tied to their appropriate measurement units. The property that specifies this is 'hasMeasurementUnit', which relates instances in the class 'Descriptor' to instances in the class 'Measurement Unit'.

This ontology can be used to describe the videos and external effects on it such as environmental conditions. As with the Goal Ontology it is used for consistency checking when a set of video descriptors are supplied to the system from the user interface. A complete list of the instances and object properties along with the associated class hierarchies can be found in the Appendix B.

5 Capability Ontology

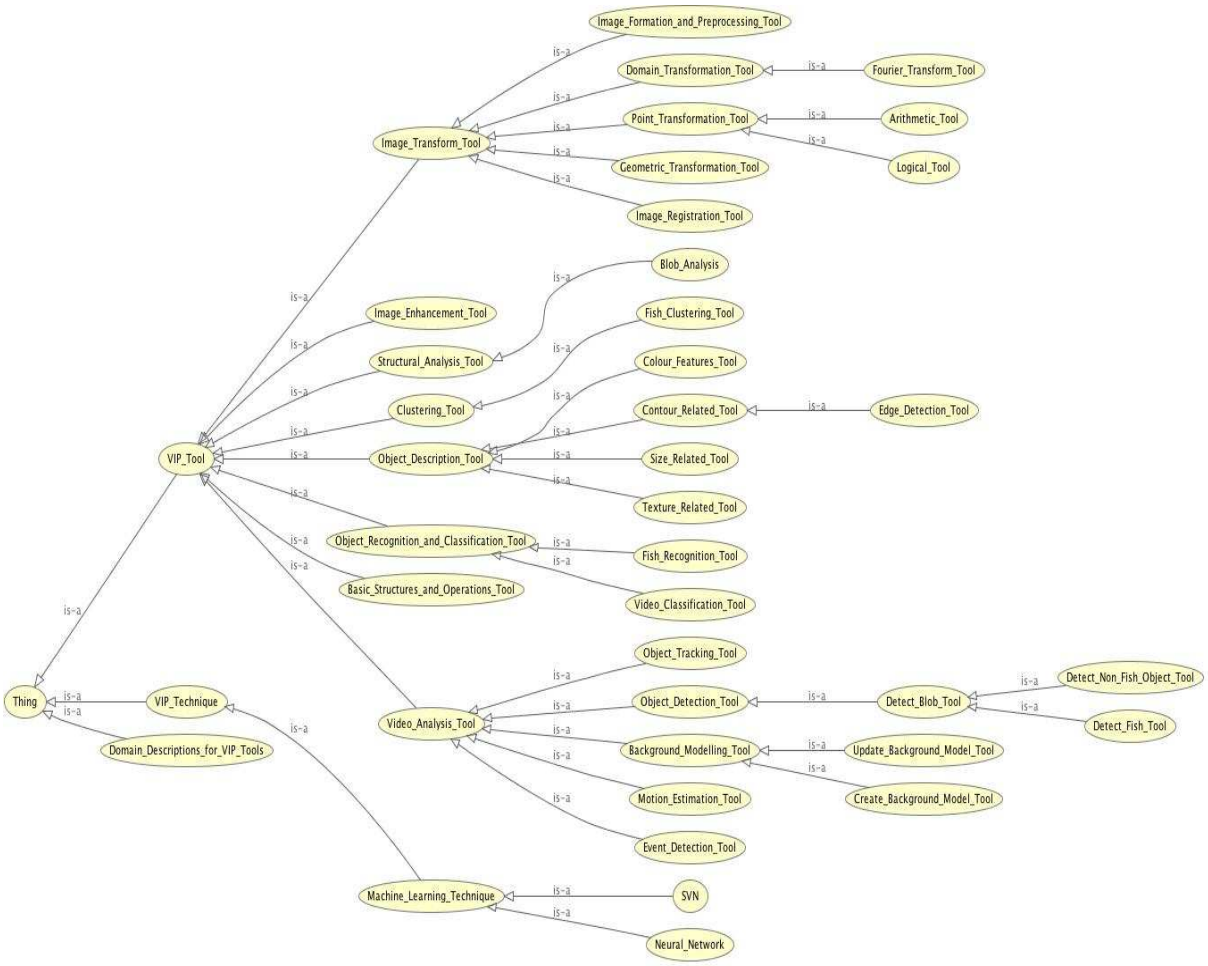

Figure 6: Main concepts of the Capability Ontology.

[1]Currently, there is not a fixed conversation formula. The actual conversion formula used will be determined as we gain more experience by using our workflow system over time.

The Capability Ontology (Figure 6) contains the classes of video and image processing tools, techniques and performance measures of the tools with known domain heuristics. This ontology will be used to identify the tools that will be used for workflow composition and execution of VIP tasks. In our context, it is used by a reasoner and a process library for the selection of optimal VIP tools. The main concepts intended for this ontology have been identified as 'VIP Tool', 'VIP Technique' and 'Domain Descriptions for VIP Tools'. Each VIP technique can be used in association with one or more VIP tools. A tool is a software component that can perform a video or image processing task independently, or a function within an integrated vision library that may be invoked with given parameters. 'Domain Description for VIP Tool' represent a combination of known domain descriptions (video descriptions and/or constraints to goals) that are recommended for a subset of the tools. This will be used to indicate the suitability of a VIP tool when a given set of domain conditions hold at a certain point of execution. At present these domain descriptions are represented as strings and tied to VIP tools, e.g. Gaussian background model would have the description 'Clear and Fast Background Movement' to indicate the best domain scenario for it to be selected.

The main types of VIP tools are video analysis tools, image enhancement tools, clustering tools, image transform tools, basic structures and operations tools, object description tools, structural analysis tools and object recognition and classification tools. At present fish detection and tracking have been performed more than the other tasks within Fish4Knowledge. Hence the ontology has been populated with most of the tools associated with these tasks. For other tasks that have not been performed, *e.g.* fish clustering, the ontology will be extended and populated in due course. Detection and tracking tools fall under the class 'Video Analysis Tool'. Other types of video analysis tools are event detection tools, background modelling tools and motion estimation tools.

The class 'Object Description Tool' specifies tools that extract features such as colour, texture, size and contour, while image transform tools are those concerned with operations such as point, geometric and domain transformations.

'VIP Technique' is a class that contains technologies that can perform VIP operations. For now, two types of machine learning techniques have been identified. These techniques could be used to accomplish the task of one or more VIP tools. For example, neural netwworks can be used as classifiers as well as detectors.

The Capability Ontology can be used for reasoning during workflow composition using planning. As planning takes into account preconditions before selecting a step or tool, it will assess the domain conditions that hold to be used in conjunction with an appropirate VIP tool.

6 Processes

To describe the processes involved in solving VIP tasks within F4K, a **process library** is more appropriate at this point of time for our system, as the implementation of the VIP modules is not complete yet. A process library contains a collection of code assembled to perform a set of related coordinating and computing tasks.

6.1 Independent Executables

Our process library consists of a group of independent VIP modules (contained as instances of VIP tools in the Capability Ontology) that can be combined in a particular sequence to solve a subset of VIP tasks that are contained in the Goal Ontology. These independent modules, termed as *independent executables*, were developed in close collaboration with IP experts using a combined top-down and bottom-up procedure [4]. The independent tasks are also primitive by nature, in that they can no longer be broken down into smaller subtasks. Thirty independent executables were designed for three VIP tasks: 1) video classification according to brightness, clearness and green levels; 2) fish detection in given individual video clips and; 3) fish counting in individual frames of a given video clip. For each independent executable, its name (user terminology), preconditions, postconditions, input and output are specified. Preconditions are a set of conditions that must hold *before* this executable can be applied, in particular domain conditions. Postconditions are a set of conditions that must hold *after* the executable is applied. Here is a list of the independent executables contained in the process library. Their full declarative syntax in Prolog is given in the Appendix D.

1. **View Video.** The aim of this executable is to display a specified video to the screen, determine its frame rate, number of frames and creation date, and compute candidate background images from it.

2. **Preliminary Analysis.** This module is responsible for capturing the initial video description of the video. A small motion detection operation is also performed to identify background movement, *e.g.* plants. Given a video file, it retrieves the initial brightness, clearness, speed of movement, percentage background object, background movement and initial texture features (variance, skewness, uniformity, entropy). These features are saved in a text file, prelim.dat in the video directory for further manipulation and will be verified with the concepts and instances contained in the Video Description Ontology. This component is generic and can be applied to any video.

3. **Grab Frame Image.** This component takes a video file and a frame number as input, retrieves the image for that frame and stores it in the video directory for further processing. This executable is essential for almost all video processing tasks because each frame of the video is processed using its image representation. This component is generic and can be applied to any video.

4. **Extract RGB Colours.** Given a frame image, this executable extracts the red, green and blue channel images of a frame. The resulting channel images are stored in the frame directory. This component is generic and can be applied to any video. An additional channel that could be derived from this executable is the yellow channel. It is computed using an arithmetical and logical combination of the red, green and blue channels and can be used for computing the video quality.

5. **Compute Histogram.** This component takes in a frame image, computes its histogram value and image representation. This component is generic and can be applied to any video.

6. **Compute Main Statistical Moments.** Based on the histogram values computed in component 5, this module determines the main statistical moments such as mean, variance,

third moment, fourth moment, entropy, uniformity and smoothness for the histogram. This component is generic and can be applied to any video.

7. **Compute Gabor Filter.** A Gabor filter is a linear filter used in image processing for edge detection. This component applies a Gabor filter to a complex image made up of a real and an imaginary part. The absolute value of the complex image is computed, followed by the mean and variance, which are texture features. For four angles and three scales, this will yield 12 mean-variance pair values, so for each image 24 values will be extracted. These features could be used for the detection of coral reef, for instance. This component is generic and can be applied to any video.

8. **Create Gaussian Background Model.** Given a frame image, this module creates a background model represented by 2 images; foreground and background. It also stores the background model in a directory within the video directory. This background model is good for videos where movement of background objects vary slightly and is thus **not** suitable for videos with *waving trees*[2].

9. **Create and Apply Gaussian Mixture Model.** Similar to component 8 above, the background model created by this executable is represented by two images and stored in a directory within the video directory. In addition, this module overcomes the limitation of component 8 and is suitable for videos with *waving trees*. This component also detects moving objects in the current frame image and stores the resulting binary image in the frame directory.

10. **Create Moving Average Model.** This module takes in a frame image, a directory to store the resulting background model and a *learning speed (alpha)* and creates an image that represents the background model. The advantage of this algorithm is that it does not require a learning phase. Alpha is dependent on changes (*e.g.* lighting, speed of movement), which could be obtained from 2 (Preliminary Analysis).

11. **Create Intra-Frame Difference Model.** This module only requires a directory where the background model needs to be stored. The background model is represented by two matrices containing the mean of the pixels of the frames stored in the buffer and the maximum variation of two consecutive pixels of the frames stored in the buffer. This algorithm overcomes the limitations of component 8 but is problematic for videos with impulsive noise (*e.g.* salt and pepper noise).

12. **Create W4 Model.** This module also only requires a directory where the background model is to be stored. The background model is represented by three matrices containing the mean of the pixels of the frames stored in the buffer, the maximum values of the pixels of the frames stored in the buffer and the minimum values of the pixels of the frames stored in the buffer. This algorithm overcomes the limitations of components 8 and 11 and is particularly suitable for videos with low changes in luminosity (*e.g.* indoor videos).

13. **Create Poisson Model.** As with components 11 and 12 this module takes in a directory where the background model is to be stored. The background model is represented by two

[2]Movement of non interesting objects such as leaves, trees, algae, *etc.*

matrices containing the mean and standard deviation of the Poisson distribution calculated from the weighted mean of the pixels' histogram (λ). This algorithm is particularly suitable for videos with uniform background colour (*e.g.* blue water, tar road).

14. **Create and Apply Adaptive Poisson Model.** Similar to the three components above, only a directory name is required to store the background model. The background model is represented by two images; foreground and background. This algorithm is similar to component 13, additionally it can manage colour variation with light changes. It also detects moving objects in the current frame image and stores the resulting binary image in the frame directory.

15. **Update Moving Average Background Model.** This component takes in a frame image and an existing moving average background model (first created using component 10) to create a new background model (1 image). It utilises absolute subtraction between pixels.

16. **Fuse Background Images.** This executable fuses two background images using the logical `AND` operator. A scenario where this component is useful is when none of the seven background models is suitable for a given set of domain description and constraints. In this case, a background model is created using the fusion of the Adaptive Gaussian Mixture Model and the Moving Average Model.

17. **Detect Moving Objects.** This module creates an image with identified blobs from a frame image and a non-adaptive background model (components 8, 11–13). The result is a binary (black and white) image. The algorithm works by removing occlusion and small objects. It also utilises statistical or derivative algorithm based on the type of the background model. This component is generic and can be applied to any video.

18. **Perform Morphological Operation: smoothing, dilation and erosion.** This operation is only applicable to non adaptive background models. Given an image with identified blobs (*e.g.* from component 17), an image with potential detected blobs (of fish) is returned. First, a median filter with a 17x17 kernel is applied, followed by 15 iterations of dilation and 10 iterations of erosion. This component is applicable to fish detection tasks only.

19. **Extract HSV Values.** This component extracts the images for the hue, saturation and value channels for a given frame image. The images are stored in the frame directory. This component is generic and can be applied to any video.

20. **Compute Backprojection.** This module depends on component 19 as it takes in a hue channel to create a histogram of the hue channel, which is then used to create an image which represents the backprojection of the hue plane. The backprojection image is stored in the frame directory and will be used for tracking purposes. The hue plane is used because it gives the most useful colour information for an image. This component is generic and can be applied to any video.

21. **Compute Connected Components.** This module takes in an image with potential blobs (*e.g.* from component 18) and returns an image with the detected blobs (of fish) and also the total number of blobs in that image. It also calculates the ratio between the area of the

convex hull of a blob and the area of the blob itself. This component is executed over all blobs in an image. This component is applicable to fish detection tasks only.

22. **Compute Camshift.** This module predicts what a blob will look like in the next frame. Given the backprojection of a hue plane (*e.g.* from component 20), an image with blobs (*e.g.* from component 21) and the number of blobs in the image, this algorithm draws a bounding box around each blob present and returns the centre, orientation and size of each blob. This information is stored in a text file in the video directory. This component is generic and can be applied to any image.

23. **Compute Closest Blob.** This component is responsible for finding the minimum Euclidian distance between the blobs in two frames. Thus, given the centres, orientations and sizes of the blobs in two frames, the corresponding closest blob in the second frame for each blob in the first frame is determined. It is executed in a loop to compare blobs in a segment of consecutive frames (*e.g.* 10 frames). The pairs of closest blobs between the two frames are stored in a text file, closest_blob.dat. This component is generic and can be applied to any video.

24. **Compute and Write Number of Fish in Frame and Video.** This module takes in the current frame image, an array of minimum distances (*e.g.* from component 23), the ratio between the blob area and frame area (*e.g.* from component 21) and computes the number of fish in the current frame and adds it to the total number of fish calculated in all the frames up to this frame. It writes the number of fish (blobs) in the frame and in the video so far onto an output frame and updates the total number of fish in a text file. Both the outputs are stored in the frame directory. This component is generic and can be applied to any image/video.

25. **Determine Presence of Fish Blocking Screen.** Given the area of the blob and the area of the frame, this module will return true (1) or false (0) to indicate if a blob is blocking the screen. The threshold value for the ratio between the blob and frame is set to 70% for the blob to be blocking the screen. This component is generic and can be applied to any image/video to determine if a blob is blocking the screen by a factor of 70% or more.

26. **Compute and Write Average Video Luminosity.** This module is responsible for determining if the video is “Bright”, “Medium” or “Dark” based on its average luminosity value, calculated using the mean and 3rd moment. The value is written onto the output frame (an image), which is stored in the output directory.

27. **Compute and Write Average Video Clearness.** This module is responsible for determining if the video is “Clear” or “Blur” based on its average clearness value, calculated using the variance, fourth moment, entropy and uniformity. The value is written onto the output frame (an image), which is stored in the output directory.

28. **Compute and Write Presence of Fish.** This module is responsible for determining if the video has “Fish” or “No Fish” based on the number of fish in the video. The value is written onto the output frame (an image), which is stored in the output directory.

29. **Compute and Write Presence of Algae.** This module is responsible for determining if the video is “Green” or “Not Green” based on its green channel value. The value is written onto the output frame (an image), which is stored in the output directory.

30. **Write to (Final) Output Video.** Given a specified directory, an output video is created using all the frames in that directory with “.jpg” or “.jpeg” extensions sorted by name, *e.g.* 1.jpg–50.jpg. This is used to create the output video containing texts indicating classification and counting results and boundary boxes around detected objects.

These independent executables are subject to change with the facilitation of computing resources from NCHC and updates in VIP results. One possibility is that the level of granularity of these modules should be coarser to enable large batch processing to take place efficiently in the cluster. As these executables were developed to work on a single video at each time of processing, it may not be feasible to use the same level of granularity for multiple videos (in the order of hundreds or thousands) and to be distributed between multiple processors (estimated up to 96 at present). We plan to merge tasks that are small and take less time to compute and keep tasks that are computationally expensive (*e.g.* detection algorithms) so that a more balanced scheduling workload can be achieved with distributed resources.

6.2 Process Models (Methods)

Apart from independent executables, the process library also contains the process models or *methods* for representing non-primitive tasks, *i.e.* tasks that can be further broken down into smaller primitive and non-primitive subtasks. Non-primitive tasks cannot be solved directly using IP tools, hence they will need to be decomposed until primitive tasks, or those that can be directly achieved using IP tools, are encountered. To illustate some examples, a few process model diagrams are included next.

A process model for an image processing task describes the process logic for that task, which can be depicted at a higher level or a lower level of computation. When executed, a process model is instantiated with input values that consist of strings that represent numerical parameters, directories, filenames, images and videos, etc. Thus one process model can be instantiated into many execution chains based on different input values, as well as different process logics due to the preconditions that hold for that process logic. One advantage of using process models is that they highlight decision points in the flow of processing clearly.

The process model diagrams in the rest of this section should be read as follows. Every process model begins with a ‘start’ node and terminates with an ‘end’ node. Processes, contained in rectangular nodes, can be decomposed to one or more subtasks. A process that can not be decomposed further is a primitive process and will be shaded. Circular nodes containing logical operators ‘AND’ or ‘OR’ indicate that all processes or at least one process, respectively, must complete before the next can begin. Circular nodes containing the operator ‘XOR’ indicate choice, meaning only one process should be selected prior to or after this junction.

Consider the task ‘Perform Video Classification, Fish Detection and Counting’ on a video clip. The process model for this task is given by Figure 7. The whole task is made up of six main subtasks and contains one main loop that executes over all the applicable frames in a video. This will be encoded in the process library as a method. The method for this task will be made up of six subtasks, ordered in a sequence. To provide further examples, a few of these subtasks are examined in more detail.

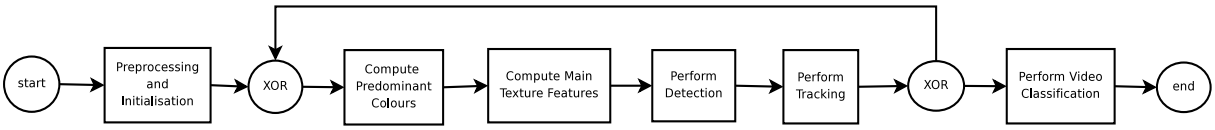

Figure 7: Process model for ‘Perform Video Classification, Fish Detection and Counting’.

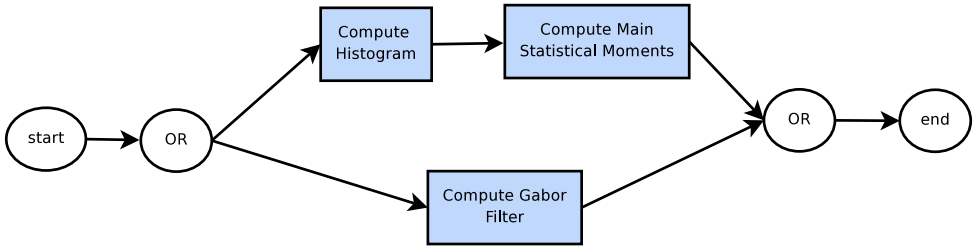

Figure 8: Process model for ‘Compute Main Texture Features’.

The subtask ‘Compute Main Texture Features’, shown in Figure 8 consists of sub processes that may be executed in parallel (given by the ‘OR’ construct). The computation of the main statistical moments (shaded in blue as it is a primitive process) is only possible after the computation of the histogram. While the computation of the Gabor filter may be performed independently. This process model will be represented in the process library. Using the declarative syntax provided by Prolog, constructs such as conjunction (AND), disjunction (OR) and choice (XOR) can be encoded relatively easily.

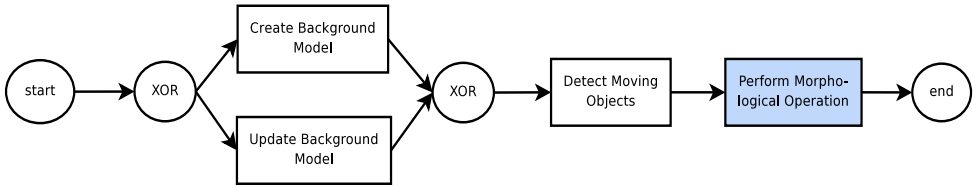

Figure 9: Process model for ‘Perform Detection’.

‘Perform Detection’ (Figure 9 is one of the main subtasks of the whole process and will be applied in a loop that takes in one frame of the video at a time starting from the first frame.

‘Create Background Model’ is applied to the first frame of a video. There are seven algorithms in which a background model can be created. These are given by the algorithms in Figure 10. Only one of them is selected for a video (illustrated by the ‘XOR’ notation). The selection of the detection algorithm is dependent on domain conditions that hold at that point in time, for example environmental conditions and user preferences. These preferences and measures to tools are encoded in the Capability Ontology, as described in Section 5.

For the subtask ‘Update Background Model’, three branches are possible depending on which type of background model has been applied on the first frame. These three options are shown in Figure 11. For the Moving Average background model, a specific algorithm is required, while for any other non-adaptive background model, the background model is recreated. Adaptive background models (Adaptive Gaussian Mixture model and Adaptive Poisson model) do not need any updating.

Finally the method for ‘Video Classification’ (Figure 12) is made up of a set of classification

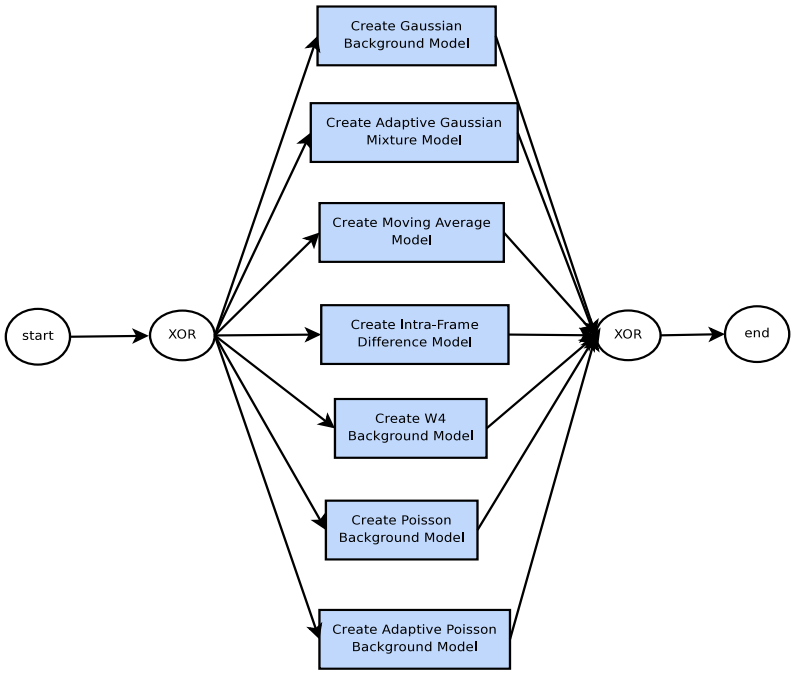

Figure 10: Process model for ‘Create Background Model’.

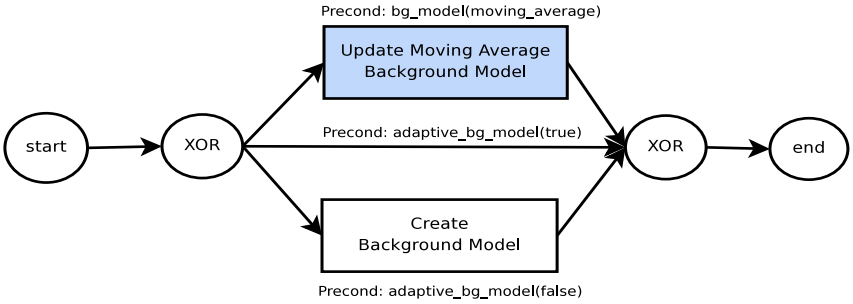

Figure 11: Process model for ‘Update Background Model’.

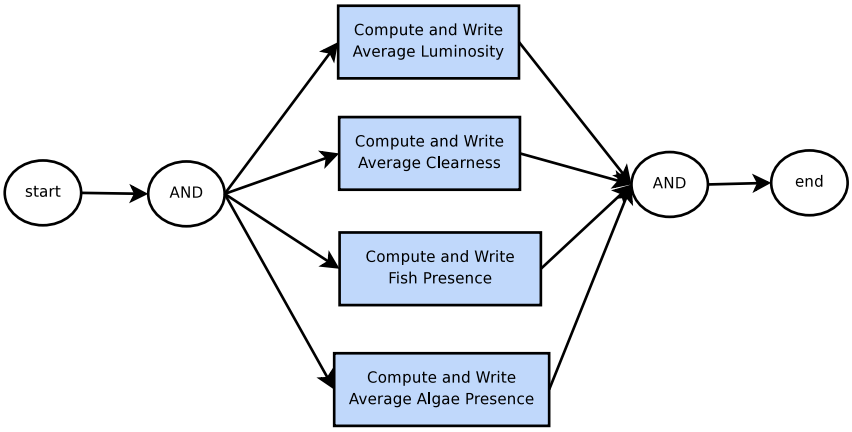

Figure 12: Process model for ‘Perform Video Classification’.

features that need to be determined and written to the output frames. All these subtasks must be performed (indicated by the 'AND' construct) and completed. However, they are not dependent on one another and could be run in parallel.

7 Conclusions

We have presented the aspects that relate to the goals, capabilities, environmental conditions and processes within the F4K project. These are encoded in three ontologies and a process library at the present status of work in progress[3]. These ontologies were constructed after communicating with IP experts, marine biologists and user interface experts. Development of these ontologies act as a starting point to bring various expertise within the project domain together (*e.g.* IP experts, marine biologists, user interface experts, workflow engineers, *etc.*). At its current status, we plan to use these ontologies to guide the construction of VIP workflows that will be executed on a many-core architecture (*e.g.* a cluster). This would require suitable packaging of the VIP modules from its current level of granularity and reconstruction of the workflows to deal with batches of jobs over multiple videos. This is planned to be followed by an appropriate scheduling and parallelisation strategy for the VIP workflows on a 96-core cluster, subject to confirmation from the high performance team at NCHC. At present the ontologies are interfacing the workflow and the IP components, it is envisaged that the ontologies would also serve as a communication media to interface with other F4K components in due course.

[3]The ontologies are available online at `http://homepages.inf.ed.ac.uk/gnadaraj/ontology/` and will be updated accordingly with changing needs within F4K.

Appendix A: Goal Ontology

```
<?xml version="1.0"?>

<!DOCTYPE rdf:RDF [
    <!ENTITY owl "http://www.w3.org/2002/07/owl#" >
    <!ENTITY dc "http://purl.org/dc/elements/1.1/" >
    <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
    <!ENTITY owl2xml "http://www.w3.org/2006/12/owl2-xml#" >
    <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#" >
    <!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >
    <!ENTITY Ontology1311945138 "http://www.owl-ontologies.com/Ontology1311945138.owl#" >
]>

<rdf:RDF xmlns="http://www.owl-ontologies.com/Ontology1311945138.owl#"
     xml:base="http://www.owl-ontologies.com/Ontology1311945138.owl"
     xmlns:dc="http://purl.org/dc/elements/1.1/"
     xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
     xmlns:owl2xml="http://www.w3.org/2006/12/owl2-xml#"
     xmlns:owl="http://www.w3.org/2002/07/owl#"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:Ontology1311945138="http://www.owl-ontologies.com/Ontology1311945138.owl#">
    <owl:Ontology rdf:about=""/>

    <!--
    ///////////////////////////////////////////////////////////////////////////////////////
    //
    // Annotation properties
    //
    ///////////////////////////////////////////////////////////////////////////////////////
     -->

    <owl:AnnotationProperty rdf:about="&dc;source"/>
    <owl:AnnotationProperty rdf:about="&dc;description"/>

    <!--
    ///////////////////////////////////////////////////////////////////////////////////////
    //
    // Object Properties
    //
    ///////////////////////////////////////////////////////////////////////////////////////
     -->

    <!-- http://www.owl-ontologies.com/Ontology1311945138.owl#isRelatedTo -->

    <owl:ObjectProperty rdf:about="#isRelatedTo">
        <rdf:type rdf:resource="&owl;SymmetricProperty"/>
        <dc:description
            >This property relates constraints to goals, each constraint can be applicable to
             one or more goals. For example the constraint ‘occurrence&#39; is tied to
             ‘detection&#39; goals.</dc:description>
        <rdfs:domain rdf:resource="#Constraint_on_Goal"/>
        <rdfs:range rdf:resource="#Goal"/>
    </owl:ObjectProperty>

    <!--
    ///////////////////////////////////////////////////////////////////////////////////////
    //
    // Classes
    //
    ///////////////////////////////////////////////////////////////////////////////////////
     -->

    <!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Acceptable_Error -->

    <owl:Class rdf:about="#Acceptable_Error">
        <rdfs:subClassOf rdf:resource="#Constraint_on_Goal"/>
    </owl:Class>
```



```
<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Accuracy -->

<owl:Class rdf:about="#Accuracy">
    <rdfs:subClassOf rdf:resource="#Acceptable_Error"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Activity_Based_Behaviour -->

<owl:Class rdf:about="#Activity_Based_Behaviour">
    <rdfs:subClassOf rdf:resource="#Behaviour_Understanding"/>
    <dc:description
        >Activities include daily tasks such as feeding habits, reproduction, solitary and
         pairing behaviours, etc.</dc:description>
    <dc:source
        >See Table 2 in document Deliverable 3.1.</dc:source>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Aggregated_Analysis -->

<owl:Class rdf:about="#Aggregated_Analysis">
    <rdfs:subClassOf rdf:resource="#Goal"/>
    <dc:description
        >Tasks that fall under this class are those that cannot be performed by a single
         image processing task alone, but those that require futher aggregation and
         combination of results from several processes. These include ranking tasks,
         comparison of different techniques (e.g. manual vs automatic), etc.</dc:description>
    <dc:source
        >Table 2 in document Deliverable 3.1 and list of 20 Questions provided by marine
         scientists.</dc:source>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Assemblage_Analysis -->

<owl:Class rdf:about="#Assemblage_Analysis">
    <rdfs:subClassOf rdf:resource="#Population_Related_Analysis"/>
    <dc:description
        >Assemblage is defined as a distinct group corresponding to different environmental
         conditions.</dc:description>
    <rdfs:comment
        >Discuss if assemblage has any association or similarity with Clustering.</rdfs:comment>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Behaviour_Understanding -->

<owl:Class rdf:about="#Behaviour_Understanding">
    <rdfs:subClassOf rdf:resource="#Goal"/>
    <dc:source
        >See Table 2 of document Deliverable 3.1</dc:source>
    <dc:description
        >Behaviours are acquired through conditioning which occurs through interaction with
         the enviornment and can be studied in a systematic and observable manner with no
         consideration of internal mental states (Watson, 1930). Typical behaviours interested
         by marine scientists include activity-based behaviours and feature-based behaviours.

         Describes changing of features and activities that is observeable externally. Examples
         are change of fish colour or interactions with other fish or behaviour during an
         event such typhoon.</dc:description>
    <rdfs:comment
        >To discuss if current taxonomy should be revised to Individual, Group and
         Interactive based behaviours.</rdfs:comment>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Boundaries -->

<owl:Class rdf:about="#Boundaries">
    <rdfs:subClassOf rdf:resource="#Detail_Level"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Classification -->
```



```
<owl:Class rdf:about="#Classification">
    <rdfs:subClassOf rdf:resource="#Goal"/>
    <dc:source
        >See Table 2 of document Deliverable 3.1</dc:source>
    <dc:description
        >This task is concerned primarily with the identification of specific species of
         fish and other marine life. Two major subtasks within classification are Recognition
         and Detection and Tracking (Counting).</dc:description>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Comparison_Studies -->

<owl:Class rdf:about="#Comparison_Studies">
    <rdfs:subClassOf rdf:resource="#Aggregated_Analysis"/>
    <dc:source
        >Table 2 in document Deliverable 3.1 and list of 20 Questions provided by marine
         scientists.</dc:source>
    <dc:description
        >These tasks compares the effect of using different tools or techniques in solving
         some analysis. These include comparing manual and automatic video analysis (Q16),
         different camera types of camera lengths (Q17) and their angle widths, etc.
    </dc:description>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Constraint_on_Goal -->

<owl:Class rdf:about="#Constraint_on_Goal">
    <owl:disjointWith rdf:resource="#Goal"/>
    <dc:description
        >This class contains the constraints on VIP goals. These include temporal constraints,
         accuracy levels, speed of performance, accuracy of solutions, etc.</dc:description>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Control_Constraint -->

<owl:Class rdf:about="#Control_Constraint">
    <rdfs:subClassOf rdf:resource="#Constraint_on_Goal"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Coral_Detection -->

<owl:Class rdf:about="#Coral_Detection">
    <rdfs:subClassOf rdf:resource="#Object_Detection"/>
    <dc:description
        >This task identifies objects that resemble coral in a video or image.</dc:description>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Detail_Level -->

<owl:Class rdf:about="#Detail_Level">
    <rdfs:subClassOf rdf:resource="#Constraint_on_Goal"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Display_Task -->

<owl:Class rdf:about="#Display_Task">
    <rdfs:subClassOf rdf:resource="#Goal"/>
    <dc:source
        >See Section 2 in document Deliverable 3.1.</dc:source>
    <dc:description
        >This task will purely display an image or video source for viewing purposes, this
         may include source images/videos or annotated images/videos.</dc:description>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Duration -->

<owl:Class rdf:about="#Duration">
    <rdfs:subClassOf rdf:resource="#Temporal_Constraint"/>
</owl:Class>
```



```
<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Emigration_Analysis -->

<owl:Class rdf:about="#Emigration_Analysis">
    <rdfs:subClassOf rdf:resource="#Population_Related_Analysis"/>
    <dc:source
        >Table 2 in document Deliverable 3.1 and list of 20 Questions provided by marine
         scientists.</dc:source>
    <dc:description
        >This analysis deals with questions concerning the number, rate and percentage of
         fish leaving a specified location.</dc:description>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Event_Detection -->

<owl:Class rdf:about="#Event_Detection">
    <rdfs:subClassOf rdf:resource="#Goal"/>
    <dc:description
        >This task concerns the identification of events in the marine domain, these include
         environmental factors such as water turbidity, velocity, temperature, typhoon,
         surge or wave, pollution, algal level on camera, human impact and other
         disturbances.</dc:description>
    <dc:source
        >See Table 2 in document Deliverable 3.1</dc:source>
    <rdfs:comment
        >To discuss if better categorisation could be achieved, i.e. sublcasses before
         instances.</rdfs:comment>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Feature_Based_Behaviour -->

<owl:Class rdf:about="#Feature_Based_Behaviour">
    <rdfs:subClassOf rdf:resource="#Behaviour_Understanding"/>
    <rdfs:comment
        >To dicsuss if this could be considered as feature identification task</rdfs:comment>
    <dc:description
        >This task identifies behaviour that are related to features of fish, i.e. fish
         descriptions. Typical examples include colour pattern change in nocturnal and
         diurnal fish.</dc:description>
    <dc:source
        >See Table 2 of document Deliverable 3.1</dc:source>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Feature_Related_Analysis -->

<owl:Class rdf:about="#Feature_Related_Analysis">
    <rdfs:subClassOf rdf:resource="#Goal"/>
    <dc:description
        >This task identifies features or descriptions of fish. These include size, colour,
         contour and texture. These features are captured during the time of observation.
    </dc:description>
    <dc:source
        >See Table is in document Deliverable 3.1</dc:source>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Fish_Clustering -->

<owl:Class rdf:about="#Fish_Clustering">
    <rdfs:subClassOf rdf:resource="#Object_Clustering"/>
    <dc:description
        >This task gathers fish of similar properties, e.g. those with same genus but
         different species names, called sibling species together.</dc:description>
    <dc:source
        >Table 2 in document Deliverable 3.1 and list of 20 Questions provided by marine
         scientists.</dc:source>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Fish_Detection -->

<owl:Class rdf:about="#Fish_Detection">
```



```
    <rdfs:subClassOf rdf:resource="#Object_Detection"/>
    <dc:description
        >This task identifies objects that resemble fish in a video or image.</dc:description>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Fish_Feature_Analysis -->

<owl:Class rdf:about="#Fish_Feature_Analysis">
    <rdfs:subClassOf rdf:resource="#Feature_Related_Analysis"/>
    <dc:source
        >Table 2 in document Deliverable 3.1 and list of 20 Questions provided by marine
         scientists.</dc:source>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Fish_Size_Analysis -->

<owl:Class rdf:about="#Fish_Size_Analysis">
    <rdfs:subClassOf rdf:resource="#Fish_Feature_Analysis"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Fish_Species_Classification -->

<owl:Class rdf:about="#Fish_Species_Classification">
    <rdfs:subClassOf rdf:resource="#Object_Classification"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Fish_Tracking -->

<owl:Class rdf:about="#Fish_Tracking">
    <rdfs:subClassOf rdf:resource="#Object_Tracking"/>
    <dc:description
        >This task keep track of objects that have been identfied as fish in a video.
    </dc:description>
    <dc:source
        >See Table 2 in document Deliverable 3.1.</dc:source>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Goal -->

<owl:Class rdf:about="#Goal">
    <dc:description
        >These are the main tasks that could be posed by marine scientists and image processing
         experts. The tasks are categorised in an image and video processing context to assist
         with workflow construction.</dc:description>
    <dc:source
        >Table 2 in document Deliverable 3.1 and list of 20 Questions provided by marine
         scientists.</dc:source>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Image_Video_Compression -->

<owl:Class rdf:about="#Image_Video_Compression">
    <rdfs:subClassOf rdf:resource="#Goal"/>
    <dc:description
        >This task will use a suitable compression algorithm to reduce the density of an
         image or video to allow for more efficient VIP processing later on.</dc:description>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Image_Video_Segmentation -->

<owl:Class rdf:about="#Image_Video_Segmentation">
    <rdfs:subClassOf rdf:resource="#Goal"/>
    <dc:description
        >This task identifies different regions within an image or a video based on a given
         categorisation, e.g. sky, water, land, trees, people, etc.</dc:description>
    <dc:source
        >See Section 2 in document Deliverable 3.1.</dc:source>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Immigration_Analysis -->
```



```
<owl:Class rdf:about="#Immigration_Analysis">
    <rdfs:subClassOf rdf:resource="#Population_Related_Analysis"/>
    <dc:description
        >This analysis deal with questions concerning the number, rate and percentage of
         fish entering into a specified location or territory.</dc:description>
    <dc:source
        >Table 2 in document Deliverable 3.1 and list of 20 Questions provided by marine
         scientists.</dc:source>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Object_Classification -->

<owl:Class rdf:about="#Object_Classification">
    <rdfs:subClassOf rdf:resource="#Classification"/>
    <dc:source
        >See Table 2 in document Deliverable 3.1.</dc:source>
    <dc:description
        >This task classifies a given object as one of the target classes that it could
         belong to. Examples include classification of fish according to its genus and
         species and classification of coral to its specific type.</dc:description>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Object_Clustering -->

<owl:Class rdf:about="#Object_Clustering">
    <rdfs:subClassOf rdf:resource="#Goal"/>
    <dc:source
        >See Table 2 in document Deliverable 3.1</dc:source>
    <dc:description
        >This task gathers objects that have similar features together, e.g. similar
         fish species which are called sibling species</dc:description>
    <rdfs:comment
        >To discuss suitabilty on name for marine scientists</rdfs:comment>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Object_Counting -->

<owl:Class rdf:about="#Object_Counting">
    <rdfs:subClassOf rdf:resource="#Population_Related_Analysis"/>
    <dc:source
        >Table 2 in document Deliverable 3.1 and list of 20 Questions provided by marine
         scientists.</dc:source>
    <dc:description
        >This task measures the number of fish existing in a certain location over a
         certain period.</dc:description>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Object_Detection -->

<owl:Class rdf:about="#Object_Detection">
    <rdfs:subClassOf rdf:resource="#Goal"/>
    <dc:source
        >See Table 2 and Section 2 in document Deliverable 3.1.</dc:source>
    <dc:description
        >This task detects objects in a video or image according to user&#39;s needs. Among
         the objects that would be of interest include fish and coral.</dc:description>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Object_Tracking -->

<owl:Class rdf:about="#Object_Tracking">
    <rdfs:subClassOf rdf:resource="#Goal"/>
    <dc:source
        >See Table 2 in document Deliverable 3.1.</dc:source>
    <dc:description
        >This task keeps track of an object in a sequence of images (video). The reason for
         tracking an object is for the purposes of counting, which is important for
         populated-related queries and aggregated analysis.</dc:description>
</owl:Class>
```



```
<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Occurrences -->

<owl:Class rdf:about="#Occurrences">
    <rdfs:subClassOf rdf:resource="#Detail_Level"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#On_Boundary -->

<owl:Class rdf:about="#On_Boundary">
    <rdfs:subClassOf rdf:resource="#Acceptable_Error"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Performance_Criteria -->

<owl:Class rdf:about="#Performance_Criteria">
    <rdfs:subClassOf rdf:resource="#Control_Constraint"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Pixels -->

<owl:Class rdf:about="#Pixels">
    <rdfs:subClassOf rdf:resource="#Detail_Level"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Population_Growth_Analysis -->

<owl:Class rdf:about="#Population_Growth_Analysis">
    <rdfs:subClassOf rdf:resource="#Population_Related_Analysis"/>
    <dc:description
        >This task deals with questions concerning the increase in the number of fish
         population in a given location.</dc:description>
    <dc:source
        >Table 2 in document Deliverable 3.1 and list of 20 Questions provided by marine
         scientists.</dc:source>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Population_Related_Analysis -->

<owl:Class rdf:about="#Population_Related_Analysis">
    <rdfs:subClassOf rdf:resource="#Goal"/>
    <dc:source
        >See Table 2 in document Deliverable 3.1</dc:source>
    <dc:description
        >This class contains goal related to population of marine life. Typical examples
         include population change, species composition change, emigration and immigration
         change.</dc:description>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Population_Size -->

<owl:Class rdf:about="#Population_Size">
    <rdfs:subClassOf rdf:resource="#Population_Related_Analysis"/>
    <dc:description
        >This task deals with questions concerning the fluctuation in the population number
         over a certain period of time.</dc:description>
    <dc:source
        >Table 2 in document Deliverable 3.1 and list of 20 Questions provided by marine
         scientists.</dc:source>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Quality_Criteria -->

<owl:Class rdf:about="#Quality_Criteria">
    <rdfs:subClassOf rdf:resource="#Control_Constraint"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Ranking_Analysis -->

<owl:Class rdf:about="#Ranking_Analysis">
```



```
    <rdfs:subClassOf rdf:resource="#Aggregated_Analysis"/>
    <dc:source
        >Table 2 in document Deliverable 3.1 and list of 20 Questions provided by marine
         scientists.</dc:source>
    <dc:description
        >Tasks that invlove ranking of species, count of species, percentage of species,
         and possibly population-related statistics.</dc:description>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Region -->

<owl:Class rdf:about="#Region">
    <rdfs:subClassOf rdf:resource="#Detail_Level"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#SegmentationError -->

<owl:Class rdf:about="#SegmentationError">
    <rdfs:subClassOf rdf:resource="#Acceptable_Error"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Segmentation_Detail -->

<owl:Class rdf:about="#Segmentation_Detail">
    <rdfs:subClassOf rdf:resource="#Detail_Level"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Separability -->

<owl:Class rdf:about="#Separability">
    <rdfs:subClassOf rdf:resource="#Acceptable_Error"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Separability_Detail -->

<owl:Class rdf:about="#Separability_Detail">
    <rdfs:subClassOf rdf:resource="#Detail_Level"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Species_Composition -->

<owl:Class rdf:about="#Species_Composition">
    <rdfs:subClassOf rdf:resource="#Population_Related_Analysis"/>
    <dc:source
        >Table 2 in document Deliverable 3.1 and list of 20 Questions provided by marine
         scientists.</dc:source>
    <dc:description
        >This task deals with questions concerning the change in the number of species per
         sample in a specified location over a period of time.</dc:description>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Temporal_Constraint -->

<owl:Class rdf:about="#Temporal_Constraint">
    <rdfs:subClassOf rdf:resource="#Constraint_on_Goal"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Video_or_Image_Classification -->

<owl:Class rdf:about="#Video_or_Image_Classification">
    <rdfs:subClassOf rdf:resource="#Classification"/>
    <dc:source
        >See Table 2 in document Deliverable 3.1.</dc:source>
    <dc:description
        >This task classifies a given video or image according to certain specifications, a
         typical example is the classification of a video according to its brightness,
         clearness and algal levels.</dc:description>
</owl:Class>

<!-- http://www.w3.org/2002/07/owl#Thing -->
```



```
    <owl:Class rdf:about="&owl;Thing"/>

    <!--
    ////////////////////////////////////////////////////////////////////////////////
    //
    // Individuals
    //
    ////////////////////////////////////////////////////////////////////////////////
     -->
    <!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Accurate_Segmentation -->
    <owl:Thing rdf:about="#Accurate_Segmentation"/>

    <!-- http://www.owl-ontologies.com/Ontology1311945138.owl#All -->
    <owl:Thing rdf:about="#All">
        <rdf:type rdf:resource="#Duration"/>
    </owl:Thing>

    <!-- http://www.owl-ontologies.com/Ontology1311945138.owl#All_Occurrences -->
    <owl:Thing rdf:about="#All_Occurrences">
        <rdf:type rdf:resource="#Occurrences"/>
    </owl:Thing>

    <!-- http://www.owl-ontologies.com/Ontology1311945138.owl#All_Pixels -->
    <owl:Thing rdf:about="#All_Pixels">
        <rdf:type rdf:resource="#Pixels"/>
    </owl:Thing>

    <!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Association_with_other_Fish_Species_
         or_Fish_Invertebrates -->
    <Activity_Based_Behaviour rdf:about="#Association_with_other_Fish_Species_or_Fish_Invertebrates">
        <rdf:type rdf:resource="&owl;Thing"/>
    </Activity_Based_Behaviour>

   <!-- http://www.owl-ontologies.com/Ontology1311945138.owl#At_Least_One_Occurrence -->
    <owl:Thing rdf:about="#At_Least_One_Occurrence">
        <rdf:type rdf:resource="#Occurrences"/>
    </owl:Thing>

    <!-- http://www.owl-ontologies.com/Ontology1311945138.owl#At_Least_One_Pixel_per_Object -->
    <owl:Thing rdf:about="#At_Least_One_Pixel_per_Object">
        <rdf:type rdf:resource="#Pixels"/>
    </owl:Thing>

    <!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Autumn -->
    <Duration rdf:about="#Autumn">
        <rdf:type rdf:resource="&owl;Thing"/>
    </Duration>

    <!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Average_Group_Fish_Size -->
    <Fish_Size_Analysis rdf:about="#Average_Group_Fish_Size">
        <rdf:type rdf:resource="&owl;Thing"/>
    </Fish_Size_Analysis>

    <!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Best_Compromise -->
    <owl:Thing rdf:about="#Best_Compromise">
        <rdf:type rdf:resource="#Quality_Criteria"/>
    </owl:Thing>

    <!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Boundaries_Inside_the_Region -->
    <Boundaries rdf:about="#Boundaries_Inside_the_Region">
        <rdf:type rdf:resource="&owl;Thing"/>
    </Boundaries>

    <!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Boundaries_Localisation -->
    <owl:Thing rdf:about="#Boundaries_Localisation"/>

    <!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Boundaries_Outside_the_Region -->
    <Boundaries rdf:about="#Boundaries_Outside_the_Region">
        <rdf:type rdf:resource="&owl;Thing"/>
```



```
    </Boundaries>

    <!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Classify_Video -->
    <Video_or_Image_Classification rdf:about="#Classify_Video">
        <rdf:type rdf:resource="&owl;Thing"/>
    </Video_or_Image_Classification>

    <!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Colour_Pattern_Change_Diurnal_Fish -->
    <owl:Thing rdf:about="#Colour_Pattern_Change_Diurnal_Fish">
        <rdf:type rdf:resource="#Feature_Based_Behaviour"/>
    </owl:Thing>

    <!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Colour_Pattern_Change_Nocturnal_Fish -->
    <Feature_Based_Behaviour rdf:about="#Colour_Pattern_Change_Nocturnal_Fish">
        <rdf:type rdf:resource="&owl;Thing"/>
    </Feature_Based_Behaviour>

    <!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Compression_with_Loss -->
    <Image_Video_Compression rdf:about="#Compression_with_Loss">
        <rdf:type rdf:resource="&owl;Thing"/>
    </Image_Video_Compression>

    <!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Compression_without_Loss -->
    <Image_Video_Compression rdf:about="#Compression_without_Loss">
        <rdf:type rdf:resource="&owl;Thing"/>
    </Image_Video_Compression>

    <!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Contour_Regularisation -->
    <Region rdf:about="#Contour_Regularisation">
        <rdf:type rdf:resource="&owl;Thing"/>
    </Region>

    <!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Day -->
    <Duration rdf:about="#Day">
        <rdf:type rdf:resource="&owl;Thing"/>
    </Duration>

    <!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Do_Not_Separate -->
    <owl:Thing rdf:about="#Do_Not_Separate">
        <rdf:type rdf:resource="#Separability_Detail"/>
    </owl:Thing>

    <!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Eliminate -->
    <Image_Video_Segmentation rdf:about="#Eliminate">
        <rdf:type rdf:resource="&owl;Thing"/>
    </Image_Video_Segmentation>

    <!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Emigration_Rate -->
    <Population_Related_Analysis rdf:about="#Emigration_Rate">
        <rdf:type rdf:resource="#Emigration_Analysis"/>
        <rdf:type rdf:resource="&owl;Thing"/>
    </Population_Related_Analysis>

    <!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Extended_Element_Descriptor -->
    <owl:Thing rdf:about="#Extended_Element_Descriptor"/>

    <!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Extract -->
    <Image_Video_Segmentation rdf:about="#Extract">
        <rdf:type rdf:resource="&owl;Thing"/>
    </Image_Video_Segmentation>

    <!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Feeding -->
    <Activity_Based_Behaviour rdf:about="#Feeding">
        <rdf:type rdf:resource="&owl;Thing"/>
    </Activity_Based_Behaviour>

  <!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Fish_Assemblage_Fluctuation -->
    <owl:Thing rdf:about="#Fish_Assemblage_Fluctuation">
        <rdf:type rdf:resource="#Assemblage_Analysis"/>
    </owl:Thing>
```



```
<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Fish_Clustering -->
<Object_Clustering rdf:about="#Fish_Clustering">
    <rdf:type rdf:resource="&owl;Thing"/>
</Object_Clustering>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Fish_Counting -->
<Object_Counting rdf:about="#Fish_Counting">
    <rdf:type rdf:resource="&owl;Thing"/>
</Object_Counting>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Growth_Rate -->
<owl:Thing rdf:about="#Growth_Rate">
    <rdf:type rdf:resource="#Population_Growth_Analysis"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#High_Current_Velocity -->
<owl:Thing rdf:about="#High_Current_Velocity">
    <rdf:type rdf:resource="#Event_Detection"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#High_Level_Algae_on_Camera -->
<owl:Thing rdf:about="#High_Level_Algae_on_Camera">
    <rdf:type rdf:resource="#Event_Detection"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Hits -->
<owl:Thing rdf:about="#Hits"/>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Id_Common_Fish -->
<Ranking_Analysis rdf:about="#Id_Common_Fish">
    <rdf:type rdf:resource="&owl;Thing"/>
    <dc:description
        >Identify the common fish species.</dc:description>
</Ranking_Analysis>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Id_Dominant_Fish -->
<owl:Thing rdf:about="#Id_Dominant_Fish">
    <rdf:type rdf:resource="#Ranking_Analysis"/>
    <dc:description
        >Identify the dominant fish species.</dc:description>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Id_Occasional_Fish -->
<Ranking_Analysis rdf:about="#Id_Occasional_Fish">
    <rdf:type rdf:resource="&owl;Thing"/>
    <dc:description
        >Identify the occasional fish species.</dc:description>
</Ranking_Analysis>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Id_Rare_Fish -->
<owl:Thing rdf:about="#Id_Rare_Fish">
    <rdf:type rdf:resource="#Ranking_Analysis"/>
    <dc:description
        >Identify rare fish species.</dc:description>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Immigration_Rate -->
<owl:Thing rdf:about="#Immigration_Rate">
    <rdf:type rdf:resource="#Immigration_Analysis"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Included_Element_Descriptor -->
<owl:Thing rdf:about="#Included_Element_Descriptor"/>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Individual_Fish_Size -->
<Fish_Feature_Analysis rdf:about="#Individual_Fish_Size">
    <rdf:type rdf:resource="&owl;Thing"/>
</Fish_Feature_Analysis>
```



```
<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Max_Group_Fish_Size -->
<owl:Thing rdf:about="#Max_Group_Fish_Size">
    <rdf:type rdf:resource="#Fish_Feature_Analysis"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Memory -->
<Performance_Criteria rdf:about="#Memory">
    <rdf:type rdf:resource="&owl;Thing"/>
</Performance_Criteria>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Month -->
<owl:Thing rdf:about="#Month">
    <rdf:type rdf:resource="#Duration"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Night -->
<Duration rdf:about="#Night">
    <rdf:type rdf:resource="&owl;Thing"/>
</Duration>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#One_per_Object_Region -->
<owl:Thing rdf:about="#One_per_Object_Region">
    <rdf:type rdf:resource="#Region"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Over_Segmentation -->
<owl:Thing rdf:about="#Over_Segmentation">
    <rdf:type rdf:resource="#Segmentation_Detail"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Pairing -->
<owl:Thing rdf:about="#Pairing">
    <rdf:type rdf:resource="#Activity_Based_Behaviour"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Partition -->
<owl:Thing rdf:about="#Partition">
    <rdf:type rdf:resource="#Image_Video_Segmentation"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Pollution_Incident -->
<Event_Detection rdf:about="#Pollution_Incident">
    <rdf:type rdf:resource="&owl;Thing"/>
</Event_Detection>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Population_Size_Change -->
<owl:Thing rdf:about="#Population_Size_Change">
    <rdf:type rdf:resource="#Population_Size"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Predation -->
<owl:Thing rdf:about="#Predation">
    <rdf:type rdf:resource="#Activity_Based_Behaviour"/>
    <dc:description
        >Describe the activities of a predator feeds on its prey.</dc:description>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Prefer_False_Alarm_Than_Miss -->
<owl:Thing rdf:about="#Prefer_False_Alarm_Than_Miss">
    <rdf:type rdf:resource="#Accuracy"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Prefer_Miss_Than_False_Alarm -->
<owl:Thing rdf:about="#Prefer_Miss_Than_False_Alarm">
    <rdf:type rdf:resource="#Accuracy"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Prey_Escaping_Behaviour -->
<Activity_Based_Behaviour rdf:about="#Prey_Escaping_Behaviour">
    <rdf:type rdf:resource="&owl;Thing"/>
```



```
</Activity_Based_Behaviour>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Processing_Time -->
<Performance_Criteria rdf:about="#Processing_Time">
    <rdf:type rdf:resource="&owl;Thing"/>
</Performance_Criteria>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Recognise_Blackbelt_Hogfish -->
<owl:Thing rdf:about="#Recognise_Blackbelt_Hogfish">
    <rdf:type rdf:resource="#Fish_Species_Classification"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Recognise_Clown_Fish -->
<owl:Thing rdf:about="#Recognise_Clown_Fish">
    <rdf:type rdf:resource="#Fish_Species_Classification"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Recognise_Crown_Fish -->
<owl:Thing rdf:about="#Recognise_Crown_Fish">
    <rdf:type rdf:resource="#Fish_Species_Classification"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Recognise_Green_Chromis -->
<owl:Thing rdf:about="#Recognise_Green_Chromis">
    <rdf:type rdf:resource="#Fish_Species_Classification"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Recognise_Two_Stripe_Damsel -->
<Fish_Species_Classification rdf:about="#Recognise_Two_Stripe_Damsel">
    <rdf:type rdf:resource="&owl;Thing"/>
</Fish_Species_Classification>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Reliability -->
<Quality_Criteria rdf:about="#Reliability">
    <rdf:type rdf:resource="&owl;Thing"/>
</Quality_Criteria>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Reproducing -->
<owl:Thing rdf:about="#Reproducing">
    <rdf:type rdf:resource="#Activity_Based_Behaviour"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Robustness -->
<Quality_Criteria rdf:about="#Robustness">
    <rdf:type rdf:resource="&owl;Thing"/>
</Quality_Criteria>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Schooling -->
<owl:Thing rdf:about="#Schooling">
    <rdf:type rdf:resource="#Activity_Based_Behaviour"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Separate_Just_Touching_and_Not_
     Aggregate -->
<Separability_Detail rdf:about="#Separate_Just_Touching_and_Not_Aggregate">
    <rdf:type rdf:resource="&owl;Thing"/>
</Separability_Detail>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Size_of_Fish -->
<owl:Thing rdf:about="#Size_of_Fish"/>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Solitary_Behaviour -->
<Activity_Based_Behaviour rdf:about="#Solitary_Behaviour">
    <rdf:type rdf:resource="&owl;Thing"/>
</Activity_Based_Behaviour>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Species_Composition_Change -->
<owl:Thing rdf:about="#Species_Composition_Change">
    <rdf:type rdf:resource="#Species_Composition"/>
</owl:Thing>
```



```
<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Spring -->
<Duration rdf:about="#Spring">
    <rdf:type rdf:resource="&owl;Thing"/>
</Duration>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Sub_Segmentation -->
<Segmentation_Detail rdf:about="#Sub_Segmentation">
    <rdf:type rdf:resource="&owl;Thing"/>
</Segmentation_Detail>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Summer -->
<Duration rdf:about="#Summer">
    <rdf:type rdf:resource="&owl;Thing"/>
</Duration>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Sunrise -->
<Duration rdf:about="#Sunrise">
    <rdf:type rdf:resource="&owl;Thing"/>
</Duration>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Sunset -->
<owl:Thing rdf:about="#Sunset">
    <rdf:type rdf:resource="#Duration"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Surge_Wave -->
<owl:Thing rdf:about="#Surge_Wave">
    <rdf:type rdf:resource="#Event_Detection"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Territorial_Behaviour -->
<owl:Thing rdf:about="#Territorial_Behaviour">
    <rdf:type rdf:resource="#Activity_Based_Behaviour"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Touching_Border -->
<owl:Thing rdf:about="#Touching_Border"/>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Typhoon -->
<Event_Detection rdf:about="#Typhoon">
    <rdf:type rdf:resource="&owl;Thing"/>
</Event_Detection>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Unusual_Water_Salinity -->
<owl:Thing rdf:about="#Unusual_Water_Salinity">
    <rdf:type rdf:resource="#Event_Detection"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Unusual_Water_Temperature -->
<owl:Thing rdf:about="#Unusual_Water_Temperature">
    <rdf:type rdf:resource="#Event_Detection"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Unusual_Water_Turbidity -->
<owl:Thing rdf:about="#Unusual_Water_Turbidity">
    <rdf:type rdf:resource="#Event_Detection"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#View_Image -->
<Display_Task rdf:about="#View_Image">
    <rdf:type rdf:resource="&owl;Thing"/>
</Display_Task>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#View_Video -->
<Display_Task rdf:about="#View_Video">
    <rdf:type rdf:resource="&owl;Thing"/>
</Display_Task>

<!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Week -->
```



```
    <owl:Thing rdf:about="#Week">
        <rdf:type rdf:resource="#Duration"/>
    </owl:Thing>

    <!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Winter -->
    <owl:Thing rdf:about="#Winter">
        <rdf:type rdf:resource="#Duration"/>
    </owl:Thing>

    <!-- http://www.owl-ontologies.com/Ontology1311945138.owl#Year -->
    <owl:Thing rdf:about="#Year">
        <rdf:type rdf:resource="#Duration"/>
    </owl:Thing>
</rdf:RDF>

<!-- Generated by the OWL API (version 2.2.1.1138) http://owlapi.sourceforge.net -->
```

Appendix B: Video Description Ontology

```
<?xml version="1.0"?>

<!DOCTYPE rdf:RDF [
    <!ENTITY owl "http://www.w3.org/2002/07/owl#" >
    <!ENTITY dc "http://purl.org/dc/elements/1.1/" >
    <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
    <!ENTITY owl2xml "http://www.w3.org/2006/12/owl2-xml#" >
    <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#" >
    <!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >
    <!ENTITY Ontology1312290619 "http://www.owl-ontologies.com/Ontology1312290619.owl#" >
]>

<rdf:RDF xmlns="http://www.owl-ontologies.com/Ontology1312290619.owl#"
     xml:base="http://www.owl-ontologies.com/Ontology1312290619.owl"
     xmlns:dc="http://purl.org/dc/elements/1.1/"
     xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
     xmlns:Ontology1312290619="http://www.owl-ontologies.com/Ontology1312290619.owl#"
     xmlns:owl2xml="http://www.w3.org/2006/12/owl2-xml#"
     xmlns:owl="http://www.w3.org/2002/07/owl#"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
    <owl:Ontology rdf:about=""/>

    <!--
    ///////////////////////////////////////////////////////////////////////////////////////
    //
    // Annotation properties
    //
    ///////////////////////////////////////////////////////////////////////////////////////
     -->

    <owl:AnnotationProperty rdf:about="&dc;source"/>
    <owl:AnnotationProperty rdf:about="&dc;description"/>

    <!--
    ///////////////////////////////////////////////////////////////////////////////////////
    //
    // Object Properties
    //
    ///////////////////////////////////////////////////////////////////////////////////////
     -->

    <!-- http://www.owl-ontologies.com/Ontology1312290619.owl#convertTo -->

    <owl:ObjectProperty rdf:about="#convertTo">
        <dc:description
            >This method converts a qualitative value to a quantitative one.</dc:description>
        <rdfs:domain rdf:resource="#Qualitative_Value"/>
```



```
    <rdfs:range rdf:resource="#Quantitative_Value"/>
</owl:ObjectProperty>

<!-- http://www.owl-ontologies.com/Ontology1312290619.owl#hasDescriptionElement -->

<owl:ObjectProperty rdf:about="#hasDescriptionElement">
    <rdf:type rdf:resource="&owl;FunctionalProperty"/>
    <rdfs:range rdf:resource="#Description_Element"/>
    <rdfs:domain rdf:resource="#Descriptor"/>
</owl:ObjectProperty>

<!-- http://www.owl-ontologies.com/Ontology1312290619.owl#hasMeasurementUnit -->

<owl:ObjectProperty rdf:about="#hasMeasurementUnit">
    <dc:description
        >Some descriptors will have a measurement unit associated with it. for example,
         percentage for salinity, celcius for temperature and FNU for turbidity.
    </dc:description>
    <rdfs:domain rdf:resource="#Descriptor"/>
    <rdfs:range rdf:resource="#Measurement_Unit"/>
</owl:ObjectProperty>

<!-- http://www.owl-ontologies.com/Ontology1312290619.owl#hasValue -->

<owl:ObjectProperty rdf:about="#hasValue">
    <rdfs:domain rdf:resource="#Descriptor"/>
    <rdfs:range rdf:resource="#Descriptor_Value"/>
</owl:ObjectProperty>

<!--
///////////////////////////////////////////////////////////////////////////////////
//
// Classes
//
///////////////////////////////////////////////////////////////////////////////////
 -->

<!-- http://www.owl-ontologies.com/Ontology1312290619.owl#Acquisition_Condition -->

<owl:Class rdf:about="#Acquisition_Condition">
    <rdfs:subClassOf rdf:resource="#Descriptor"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312290619.owl#Acquisition_Effect -->

<owl:Class rdf:about="#Acquisition_Effect">
    <rdfs:subClassOf rdf:resource="#Description_Element"/>
    <dc:description
        >This element contains the external effects on a video, for example, environmental
         conditions and acquisition conditions.</dc:description>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312290619.owl#Boundary -->

<owl:Class rdf:about="#Boundary">
    <rdfs:subClassOf rdf:resource="#Descriptor"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312290619.owl#Colour -->

<owl:Class rdf:about="#Colour">
    <rdfs:subClassOf rdf:resource="#Descriptor"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312290619.owl#Description_Element -->

<owl:Class rdf:about="#Description_Element">
    <rdfs:subClassOf rdf:resource="#Video_Description"/>
    <dc:description
        >This class contains the type of the video description, i.e whether it is a visual
```



```
         primitive or an acquisitional effect.</dc:description>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312290619.owl#Descriptor -->

<owl:Class rdf:about="#Descriptor">
    <rdfs:subClassOf rdf:resource="#Video_Description"/>
    <dc:source
        >See Section 4 of document Deliverable 3.1</dc:source>
    <dc:description
        >These are the descriptions of the videos, such as its environmental effects, shape
         and colour features.</dc:description>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312290619.owl#Descriptor_Value -->

<owl:Class rdf:about="#Descriptor_Value">
    <dc:source
        >See Section 4 of document Deliverable 3.1</dc:source>
    <dc:description
        >This class contains the value that a decriptor can have, either quantitave or
         qualitatve values. Examples for qualitative values include &quot;low&#39;,
         &quot;medium&quot; and &quot;high&quot;.</dc:description>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312290619.owl#Edge -->

<owl:Class rdf:about="#Edge">
    <rdfs:subClassOf rdf:resource="#Descriptor"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312290619.owl#Environmental_Condition -->

<owl:Class rdf:about="#Environmental_Condition">
    <rdfs:subClassOf rdf:resource="#Descriptor"/>
    <dc:description
        >This class contains the environmeantal conditions that are relevant to a video.
         Typical examples (for underwater videos) include the velocity of the water current, the
         water salinity, pollution levels, water turbidity, water temperature, etc.</dc:description>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312290619.owl#Geometric -->

<owl:Class rdf:about="#Geometric">
    <rdfs:subClassOf rdf:resource="#Descriptor"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312290619.owl#Measurement_Unit -->

<owl:Class rdf:about="#Measurement_Unit">
    <dc:description
        >This class contains the measurement units for quantitative values.</dc:description>
    <dc:source
        >See Section 4 of document Deliverable 3.1</dc:source>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312290619.owl#Photometric -->

<owl:Class rdf:about="#Photometric">
    <rdfs:subClassOf rdf:resource="#Descriptor"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312290619.owl#Qualitative_Value -->

<owl:Class rdf:about="#Qualitative_Value">
    <rdfs:subClassOf rdf:resource="#Descriptor_Value"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312290619.owl#Quantitative_Value -->
```



```
<owl:Class rdf:about="#Quantitative_Value">
    <rdfs:subClassOf rdf:resource="#Descriptor_Value"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312290619.owl#Relation -->

<owl:Class rdf:about="#Relation">
    <dc:source
        >See Section 4 of document Deliverable 3.1</dc:source>
    <dc:description
        >This class contains the spatial, temporal and value relations related to the video
         class.</dc:description>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312290619.owl#Resulting_Image -->

<owl:Class rdf:about="#Resulting_Image">
    <rdfs:subClassOf rdf:resource="#Descriptor"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312290619.owl#Shape -->

<owl:Class rdf:about="#Shape">
    <rdfs:subClassOf rdf:resource="#Descriptor"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312290619.owl#Spatial_Relation -->

<owl:Class rdf:about="#Spatial_Relation">
    <rdfs:subClassOf rdf:resource="#Relation"/>
    <dc:description
        >Spatial relation specifies the position of a video in relation to another object.
         For example, above, below, to the left, to the right, etc.</dc:description>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312290619.owl#Temporal_Relation -->

<owl:Class rdf:about="#Temporal_Relation">
    <rdfs:subClassOf rdf:resource="#Relation"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312290619.owl#Texture -->

<owl:Class rdf:about="#Texture">
    <rdfs:subClassOf rdf:resource="#Descriptor"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312290619.owl#Value_Relation -->

<owl:Class rdf:about="#Value_Relation">
    <rdfs:subClassOf rdf:resource="#Relation"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312290619.owl#Video_Description -->

<owl:Class rdf:about="#Video_Description">
    <dc:source
        >See Section 4 of document Deliverable 3.1</dc:source>
    <dc:description
        >These are the entities that describe the data, e.g. video colour, texture,
         environmental conditions, brightness levels and so on.</dc:description>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312290619.owl#Visual_Primitive -->

<owl:Class rdf:about="#Visual_Primitive">
    <rdfs:subClassOf rdf:resource="#Description_Element"/>
    <dc:description
        >This describes the video&#39;s attributes that are not affected by external factors,
         for example its shape, size and orientation.</dc:description>
```



```
</owl:Class>

<!-- http://www.w3.org/2002/07/owl#Thing -->

<owl:Class rdf:about="&owl;Thing"/>

<!--
////////////////////////////////////////////////////////////////////////////////
//
// Individuals
//
////////////////////////////////////////////////////////////////////////////////
 -->

<!-- http://www.owl-ontologies.com/Ontology1312290619.owl#Above -->

<Spatial_Relation rdf:about="#Above">
    <rdf:type rdf:resource="&owl;Thing"/>
</Spatial_Relation>

<!-- http://www.owl-ontologies.com/Ontology1312290619.owl#Below -->

<Spatial_Relation rdf:about="#Below">
    <rdf:type rdf:resource="&owl;Thing"/>
</Spatial_Relation>

<!-- http://www.owl-ontologies.com/Ontology1312290619.owl#Clearness -->

<Acquisition_Condition rdf:about="#Clearness">
    <rdf:type rdf:resource="&owl;Thing"/>
</Acquisition_Condition>

<!-- http://www.owl-ontologies.com/Ontology1312290619.owl#Current_Velocity -->

<Environmental_Condition rdf:about="#Current_Velocity">
    <rdf:type rdf:resource="&owl;Thing"/>
</Environmental_Condition>

<!-- http://www.owl-ontologies.com/Ontology1312290619.owl#Green -->

<Qualitative_Value rdf:about="#Green">
    <rdf:type rdf:resource="&owl;Thing"/>
</Qualitative_Value>

<!-- http://www.owl-ontologies.com/Ontology1312290619.owl#High -->

<Qualitative_Value rdf:about="#High">
    <rdf:type rdf:resource="&owl;Thing"/>
</Qualitative_Value>

<!-- http://www.owl-ontologies.com/Ontology1312290619.owl#Hue -->

<Colour rdf:about="#Hue">
    <rdf:type rdf:resource="&owl;Thing"/>
</Colour>

<!-- http://www.owl-ontologies.com/Ontology1312290619.owl#Is_Disconnected_From -->

<owl:Thing rdf:about="#Is_Disconnected_From">
    <rdf:type rdf:resource="#Spatial_Relation"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1312290619.owl#Is_Included_In -->

<owl:Thing rdf:about="#Is_Included_In">
    <rdf:type rdf:resource="#Spatial_Relation"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1312290619.owl#Is_Overlapping -->
```



```
<Spatial_Relation rdf:about="#Is_Overlapping">
    <rdf:type rdf:resource="&owl;Thing"/>
</Spatial_Relation>

<!-- http://www.owl-ontologies.com/Ontology1312290619.owl#Large -->

<Qualitative_Value rdf:about="#Large">
    <rdf:type rdf:resource="&owl;Thing"/>
</Qualitative_Value>

<!-- http://www.owl-ontologies.com/Ontology1312290619.owl#Lighting -->

<owl:Thing rdf:about="#Lighting">
    <rdf:type rdf:resource="#Acquisition_Condition"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1312290619.owl#Low -->

<Qualitative_Value rdf:about="#Low">
    <rdf:type rdf:resource="&owl;Thing"/>
</Qualitative_Value>

<!-- http://www.owl-ontologies.com/Ontology1312290619.owl#Luminosity -->

<owl:Thing rdf:about="#Luminosity">
    <rdf:type rdf:resource="#Colour"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1312290619.owl#Medium -->

<Qualitative_Value rdf:about="#Medium">
    <rdf:type rdf:resource="&owl;Thing"/>
</Qualitative_Value>

<!-- http://www.owl-ontologies.com/Ontology1312290619.owl#Not_Green -->

<Qualitative_Value rdf:about="#Not_Green">
    <rdf:type rdf:resource="&owl;Thing"/>
</Qualitative_Value>

<!-- http://www.owl-ontologies.com/Ontology1312290619.owl#Orientation -->

<Geometric rdf:about="#Orientation">
    <rdf:type rdf:resource="&owl;Thing"/>
</Geometric>

<!-- http://www.owl-ontologies.com/Ontology1312290619.owl#Pollution -->

<owl:Thing rdf:about="#Pollution">
    <rdf:type rdf:resource="#Environmental_Condition"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1312290619.owl#Position -->

<owl:Thing rdf:about="#Position">
    <rdf:type rdf:resource="#Geometric"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1312290619.owl#Salinity -->

<owl:Thing rdf:about="#Salinity">
    <rdf:type rdf:resource="#Environmental_Condition"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1312290619.owl#Saturation -->

<owl:Thing rdf:about="#Saturation">
    <rdf:type rdf:resource="#Colour"/>
</owl:Thing>
```



```
    <!-- http://www.owl-ontologies.com/Ontology1312290619.owl#Size -->

    <owl:Thing rdf:about="#Size">
        <rdf:type rdf:resource="#Geometric"/>
    </owl:Thing>

    <!-- http://www.owl-ontologies.com/Ontology1312290619.owl#Small -->

    <Qualitative_Value rdf:about="#Small">
        <rdf:type rdf:resource="&owl;Thing"/>
    </Qualitative_Value>

    <!-- http://www.owl-ontologies.com/Ontology1312290619.owl#Surge_Wave -->

    <Environmental_Condition rdf:about="#Surge_Wave">
        <rdf:type rdf:resource="&owl;Thing"/>
    </Environmental_Condition>

    <!-- http://www.owl-ontologies.com/Ontology1312290619.owl#To_The_Left -->

    <Spatial_Relation rdf:about="#To_The_Left">
        <rdf:type rdf:resource="&owl;Thing"/>
    </Spatial_Relation>

    <!-- http://www.owl-ontologies.com/Ontology1312290619.owl#To_The_Right -->

    <Spatial_Relation rdf:about="#To_The_Right">
        <rdf:type rdf:resource="&owl;Thing"/>
    </Spatial_Relation>

    <!-- http://www.owl-ontologies.com/Ontology1312290619.owl#Turbidity -->

    <owl:Thing rdf:about="#Turbidity">
        <rdf:type rdf:resource="#Environmental_Condition"/>
    </owl:Thing>

    <!-- http://www.owl-ontologies.com/Ontology1312290619.owl#Typhoon -->

    <owl:Thing rdf:about="#Typhoon">
        <rdf:type rdf:resource="#Environmental_Condition"/>
    </owl:Thing>

    <!-- http://www.owl-ontologies.com/Ontology1312290619.owl#Water_Temperature -->

    <owl:Thing rdf:about="#Water_Temperature">
        <rdf:type rdf:resource="#Environmental_Condition"/>
    </owl:Thing>

    <!-- http://www.owl-ontologies.com/Ontology1312290619.owl#is_Externally_Connected -->

    <Spatial_Relation rdf:about="#is_Externally_Connected">
        <rdf:type rdf:resource="&owl;Thing"/>
    </Spatial_Relation>
</rdf:RDF>

<!-- Generated by the OWL API (version 2.2.1.1138) http://owlapi.sourceforge.net -->
```

Appendix C: Capability Ontology

```
<?xml version="1.0"?>

<!DOCTYPE rdf:RDF [
    <!ENTITY owl "http://www.w3.org/2002/07/owl#" >
    <!ENTITY dc "http://purl.org/dc/elements/1.1/" >
    <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
    <!ENTITY owl2xml "http://www.w3.org/2006/12/owl2-xml#" >
    <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#" >
```



```
    <!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >
    <!ENTITY Ontology1312300009 "http://www.owl-ontologies.com/Ontology1312300009.owl#" >
    <!ENTITY compute_AAM "http://www.owl-ontologies.com/Ontology1312300009.owl#compute_AAM/" >
]>

<rdf:RDF xmlns="http://www.owl-ontologies.com/Ontology1312300009.owl#"
     xml:base="http://www.owl-ontologies.com/Ontology1312300009.owl"
     xmlns:dc="http://purl.org/dc/elements/1.1/"
     xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
     xmlns:owl2xml="http://www.w3.org/2006/12/owl2-xml#"
     xmlns:compute_AAM="&Ontology1312300009;compute_AAM/"
     xmlns:owl="http://www.w3.org/2002/07/owl#"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:Ontology1312300009="http://www.owl-ontologies.com/Ontology1312300009.owl#">
    <owl:Ontology rdf:about=""/>

    <!--
    ///////////////////////////////////////////////////////////////////////////////////////
    //
    // Annotation properties
    //
    ///////////////////////////////////////////////////////////////////////////////////////
     -->

    <owl:AnnotationProperty rdf:about="&dc;description"/>

    <!--
    ///////////////////////////////////////////////////////////////////////////////////////
    //
    // Object Properties
    //
    ///////////////////////////////////////////////////////////////////////////////////////
     -->

    <!-- http://www.owl-ontologies.com/Ontology1312300009.owl#canAccomplish -->

    <owl:ObjectProperty rdf:about="#canAccomplish">
        <dc:description
            >This property tie VIP techniques to tools that they can accomplish. For example,
             neural network can be used for fish species classification.</dc:description>
        <rdfs:domain rdf:resource="#VIP_Technique"/>
        <rdfs:range rdf:resource="#VIP_Tool"/>
    </owl:ObjectProperty>

    <!-- http://www.owl-ontologies.com/Ontology1312300009.owl#hasPerformanceMeasure -->

    <owl:ObjectProperty rdf:about="#hasPerformanceMeasure">
        <dc:description
            >This property ties VIP tools with the domain descriptions that are suitable for
             them. For example, the Gaussian background model is suitable when the video is
             clear and the background movement is fast.</dc:description>
        <rdfs:range rdf:resource="#Domain_Descriptions_for_VIP_Tools"/>
        <rdfs:domain rdf:resource="#VIP_Tool"/>
    </owl:ObjectProperty>

    <!--
    ///////////////////////////////////////////////////////////////////////////////////////
    //
    // Classes
    //
    ///////////////////////////////////////////////////////////////////////////////////////
     -->

    <!-- http://www.owl-ontologies.com/Ontology1312300009.owl#Arithmetic_Tool -->

    <owl:Class rdf:about="#Arithmetic_Tool">
        <rdfs:subClassOf rdf:resource="#Point_Transformation_Tool"/>
    </owl:Class>
```



```
<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#Background_Modelling_Tool -->

<owl:Class rdf:about="#Background_Modelling_Tool">
    <rdfs:subClassOf rdf:resource="#Video_Analysis_Tool"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#Basic_Structures_and_Operations_Tool -->

<owl:Class rdf:about="#Basic_Structures_and_Operations_Tool">
    <rdfs:subClassOf rdf:resource="#VIP_Tool"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#Blob_Analysis -->

<owl:Class rdf:about="#Blob_Analysis">
    <rdfs:subClassOf rdf:resource="#Structural_Analysis_Tool"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#Clustering_Tool -->

<owl:Class rdf:about="#Clustering_Tool">
    <rdfs:subClassOf rdf:resource="#VIP_Tool"/>
    <dc:description
        >The tools under this class can be used to cluster objects that are similar, to be
         defined by their specific features in its subclasses.</dc:description>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#Colour_Features_Tool -->

<owl:Class rdf:about="#Colour_Features_Tool">
    <rdfs:subClassOf rdf:resource="#Object_Description_Tool"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#Contour_Related_Tool -->

<owl:Class rdf:about="#Contour_Related_Tool">
    <rdfs:subClassOf rdf:resource="#Object_Description_Tool"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#Create_Background_Model_Tool -->

<owl:Class rdf:about="#Create_Background_Model_Tool">
    <rdfs:subClassOf rdf:resource="#Background_Modelling_Tool"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#Detect_Blob_Tool -->

<owl:Class rdf:about="#Detect_Blob_Tool">
    <rdfs:subClassOf rdf:resource="#Object_Detection_Tool"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#Detect_Fish_Tool -->

<owl:Class rdf:about="#Detect_Fish_Tool">
    <rdfs:subClassOf rdf:resource="#Detect_Blob_Tool"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#Detect_Non_Fish_Object_Tool -->

<owl:Class rdf:about="#Detect_Non_Fish_Object_Tool">
    <rdfs:subClassOf rdf:resource="#Detect_Blob_Tool"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#Domain_Descriptions_for_VIP_Tools -->

<owl:Class rdf:about="#Domain_Descriptions_for_VIP_Tools">
    <owl:disjointWith rdf:resource="#VIP_Tool"/>
    <dc:description
        >This class contains known domain conditions that are suitable for VIP tools. These
         criteria will be used as &#39;recommendations&#39; for workflow or user to make
```



```
         more informed informed decisions, e.g. when domain conditions are knows they can
         select appropriate tools.</dc:description>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#Domain_Transformation_Tool -->

<owl:Class rdf:about="#Domain_Transformation_Tool">
    <rdfs:subClassOf rdf:resource="#Image_Transform_Tool"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#Edge_Detection_Tool -->

<owl:Class rdf:about="#Edge_Detection_Tool">
    <rdfs:subClassOf rdf:resource="#Contour_Related_Tool"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#Event_Detection_Tool -->

<owl:Class rdf:about="#Event_Detection_Tool">
    <rdfs:subClassOf rdf:resource="#Video_Analysis_Tool"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#Fish_Clustering_Tool -->

<owl:Class rdf:about="#Fish_Clustering_Tool">
    <rdfs:subClassOf rdf:resource="#Clustering_Tool"/>
    <dc:description
        >This tool can be used to cluster fish of similar or sibling species, for example fish
         with same genus but different species, or fish that have similiar appearance.</dc:description>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#Fish_Recognition_Tool -->

<owl:Class rdf:about="#Fish_Recognition_Tool">
    <rdfs:subClassOf rdf:resource="#Object_Recognition_and_Classification_Tool"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#Fourier_Transform_Tool -->

<owl:Class rdf:about="#Fourier_Transform_Tool">
    <rdfs:subClassOf rdf:resource="#Domain_Transformation_Tool"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#Geometric_Transformation_Tool -->

<owl:Class rdf:about="#Geometric_Transformation_Tool">
    <rdfs:subClassOf rdf:resource="#Image_Transform_Tool"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#Image_Enhancement_Tool -->

<owl:Class rdf:about="#Image_Enhancement_Tool">
    <rdfs:subClassOf rdf:resource="#VIP_Tool"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#Image_Formation_and_Preprocessing_Tool -->

<owl:Class rdf:about="#Image_Formation_and_Preprocessing_Tool">
    <rdfs:subClassOf rdf:resource="#Image_Transform_Tool"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#Image_Registration_Tool -->

<owl:Class rdf:about="#Image_Registration_Tool">
    <rdfs:subClassOf rdf:resource="#Image_Transform_Tool"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#Image_Transform_Tool -->

<owl:Class rdf:about="#Image_Transform_Tool">
```



```
    <rdfs:subClassOf rdf:resource="#VIP_Tool"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#Logical_Tool -->

<owl:Class rdf:about="#Logical_Tool">
    <rdfs:subClassOf rdf:resource="#Point_Transformation_Tool"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#Machine_Learning_Technique -->

<owl:Class rdf:about="#Machine_Learning_Technique">
    <rdfs:subClassOf rdf:resource="#VIP_Technique"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#Motion_Estimation_Tool -->

<owl:Class rdf:about="#Motion_Estimation_Tool">
    <rdfs:subClassOf rdf:resource="#Video_Analysis_Tool"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#Neural_Network -->

<owl:Class rdf:about="#Neural_Network">
    <rdfs:subClassOf rdf:resource="#Machine_Learning_Technique"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#Object_Description_Tool -->

<owl:Class rdf:about="#Object_Description_Tool">
    <rdfs:subClassOf rdf:resource="#VIP_Tool"/>
    <dc:description
        >This tool will obtain descriptions of relevant objects in a video. For example,
          extraction of colour, contour, size and texture.</dc:description>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#Object_Detection_Tool -->

<owl:Class rdf:about="#Object_Detection_Tool">
    <rdfs:subClassOf rdf:resource="#Video_Analysis_Tool"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#Object_Recognition_and_Classification_Tool -->

<owl:Class rdf:about="#Object_Recognition_and_Classification_Tool">
    <rdfs:subClassOf rdf:resource="#VIP_Tool"/>
    <dc:description
        >This tool will be responsible for the recognition of objects and classification of
         video, for example recognition of the relevant fish species and the classification
         of the video according to its brightness and clearness levels.</dc:description>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#Object_Tracking_Tool -->

<owl:Class rdf:about="#Object_Tracking_Tool">
    <rdfs:subClassOf rdf:resource="#Video_Analysis_Tool"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#Point_Transformation_Tool -->

<owl:Class rdf:about="#Point_Transformation_Tool">
    <rdfs:subClassOf rdf:resource="#Image_Transform_Tool"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#SVN -->

<owl:Class rdf:about="#SVN">
    <rdfs:subClassOf rdf:resource="#Machine_Learning_Technique"/>
</owl:Class>
```



```
<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#Size_Related_Tool -->

<owl:Class rdf:about="#Size_Related_Tool">
    <rdfs:subClassOf rdf:resource="#Object_Description_Tool"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#Structural_Analysis_Tool -->

<owl:Class rdf:about="#Structural_Analysis_Tool">
    <rdfs:subClassOf rdf:resource="#VIP_Tool"/>
    <dc:description
        >This tool will be used for specific analysis of structures. For example, the
         analysis of objects identified as blobs.</dc:description>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#Texture_Related_Tool -->

<owl:Class rdf:about="#Texture_Related_Tool">
    <rdfs:subClassOf rdf:resource="#Object_Description_Tool"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#Update_Background_Model_Tool -->

<owl:Class rdf:about="#Update_Background_Model_Tool">
    <rdfs:subClassOf rdf:resource="#Background_Modelling_Tool"/>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#VIP_Technique -->

<owl:Class rdf:about="#VIP_Technique">
    <dc:description
        >This class contains techniques that can be used with one or more VIP tools or
         operations. Typical examples include machine learning techniques such as neural
         networks and support vector networks.</dc:description>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#VIP_Tool -->

<owl:Class rdf:about="#VIP_Tool">
    <dc:description
        >This class contains all the VIP tools available for use. A tool can be an independent
         software module or a function call within a computer vision library.</dc:description>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#Video_Analysis_Tool -->

<owl:Class rdf:about="#Video_Analysis_Tool">
    <rdfs:subClassOf rdf:resource="#VIP_Tool"/>
    <dc:description
        >This tool can perform a range of analysis on videos. These include object detection,
         motion estimation, object tracking, etc.</dc:description>
</owl:Class>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#Video_Classification_Tool -->

<owl:Class rdf:about="#Video_Classification_Tool">
    <rdfs:subClassOf rdf:resource="#Object_Recognition_and_Classification_Tool"/>
</owl:Class>

<!-- http://www.w3.org/2002/07/owl#Thing -->

<owl:Class rdf:about="&owl;Thing"/>

<!--
///////////////////////////////////////////////////////////////////////////////////
//
// Individuals
//
///////////////////////////////////////////////////////////////////////////////////
 -->
```



```
<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#affine_transform -->

<owl:Thing rdf:about="#affine_transform">
    <rdf:type rdf:resource="#Geometric_Transformation_Tool"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#blur_and_fast_background_movement -->

<owl:Thing rdf:about="#blur_and_fast_background_movement">
    <rdf:type rdf:resource="#Domain_Descriptions_for_VIP_Tools"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#clear_and_fast_background_movement -->

<Domain_Descriptions_for_VIP_Tools rdf:about="#clear_and_fast_background_movement">
    <rdf:type rdf:resource="&owl;Thing"/>
</Domain_Descriptions_for_VIP_Tools>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#compute_AAM/ASM -->

<owl:Thing rdf:about="#compute_AAM/ASM">
    <rdf:type rdf:resource="#Contour_Related_Tool"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#compute_Active_Contours -->

<owl:Thing rdf:about="#compute_Active_Contours">
    <rdf:type rdf:resource="#Contour_Related_Tool"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#compute_Attention_model -->

<owl:Thing rdf:about="#compute_Attention_model">
    <rdf:type rdf:resource="#Colour_Features_Tool"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#compute_CSS -->

<Contour_Related_Tool rdf:about="#compute_CSS">
    <rdf:type rdf:resource="&owl;Thing"/>
</Contour_Related_Tool>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#compute_Curvature_Points -->

<Contour_Related_Tool rdf:about="#compute_Curvature_Points">
    <rdf:type rdf:resource="&owl;Thing"/>
</Contour_Related_Tool>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#compute_FTLE -->

<Motion_Estimation_Tool rdf:about="#compute_FTLE">
    <rdf:type rdf:resource="&owl;Thing"/>
</Motion_Estimation_Tool>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#compute_Gabor_filter -->

<owl:Thing rdf:about="#compute_Gabor_filter">
    <rdf:type rdf:resource="#Texture_Related_Tool"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#compute_Gradient_vector_flow -->

<Motion_Estimation_Tool rdf:about="#compute_Gradient_vector_flow">
    <rdf:type rdf:resource="&owl;Thing"/>
</Motion_Estimation_Tool>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#compute_Grub-cut -->

<owl:Thing rdf:about="#compute_Grub-cut">
```



```
    <rdf:type rdf:resource="#Contour_Related_Tool"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#compute_HSV -->

<Colour_Features_Tool rdf:about="#compute_HSV">
    <rdf:type rdf:resource="&owl;Thing"/>
</Colour_Features_Tool>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#compute_Lab -->

<owl:Thing rdf:about="#compute_Lab">
    <rdf:type rdf:resource="#Colour_Features_Tool"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#compute_MDL -->

<owl:Thing rdf:about="#compute_MDL">
    <rdf:type rdf:resource="#Contour_Related_Tool"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#compute_Mellon_transform -->

<owl:Thing rdf:about="#compute_Mellon_transform">
    <rdf:type rdf:resource="#Domain_Transformation_Tool"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#compute_PCA_SIFT -->

<owl:Thing rdf:about="#compute_PCA_SIFT">
    <rdf:type rdf:resource="#Texture_Related_Tool"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#compute_Point_Distribution -->

<Contour_Related_Tool rdf:about="#compute_Point_Distribution">
    <rdf:type rdf:resource="&owl;Thing"/>
</Contour_Related_Tool>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#compute_RGB -->

<owl:Thing rdf:about="#compute_RGB">
    <rdf:type rdf:resource="#Colour_Features_Tool"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#compute_SIFT -->

<owl:Thing rdf:about="#compute_SIFT">
    <rdf:type rdf:resource="#Texture_Related_Tool"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#compute_SIFT_with_global_context -->

<Texture_Related_Tool rdf:about="#compute_SIFT_with_global_context">
    <rdf:type rdf:resource="&owl;Thing"/>
</Texture_Related_Tool>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#compute_Shock_Graph -->

<Contour_Related_Tool rdf:about="#compute_Shock_Graph">
    <rdf:type rdf:resource="&owl;Thing"/>
</Contour_Related_Tool>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#compute_Snakes -->

<owl:Thing rdf:about="#compute_Snakes">
    <rdf:type rdf:resource="#Contour_Related_Tool"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#compute_TPS -->
```



```
<owl:Thing rdf:about="#compute_TPS">
    <rdf:type rdf:resource="#Contour_Related_Tool"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#compute_Wavelet_transform -->

<owl:Thing rdf:about="#compute_Wavelet_transform">
    <rdf:type rdf:resource="#Domain_Transformation_Tool"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#compute_background_scoring -->

<owl:Thing rdf:about="#compute_background_scoring">
    <rdf:type rdf:resource="#Colour_Features_Tool"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#compute_backprojection -->

<owl:Thing rdf:about="#compute_backprojection">
    <rdf:type rdf:resource="#Object_Tracking_Tool"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#compute_bounding_box_colour -->

<Colour_Features_Tool rdf:about="#compute_bounding_box_colour">
    <rdf:type rdf:resource="&owl;Thing"/>
</Colour_Features_Tool>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#compute_camshift -->

<owl:Thing rdf:about="#compute_camshift">
    <rdf:type rdf:resource="#Object_Tracking_Tool"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#compute_connected_components -->

<Object_Tracking_Tool rdf:about="#compute_connected_components">
    <rdf:type rdf:resource="&owl;Thing"/>
</Object_Tracking_Tool>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#compute_covariance_matrix -->

<owl:Thing rdf:about="#compute_covariance_matrix">
    <rdf:type rdf:resource="#Texture_Related_Tool"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#compute_histogram_of_Fourier_Descriptors -->

<Contour_Related_Tool rdf:about="#compute_histogram_of_Fourier_Descriptors">
    <rdf:type rdf:resource="&owl;Thing"/>
</Contour_Related_Tool>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#compute_histogram_of_gradients -->

<owl:Thing rdf:about="#compute_histogram_of_gradients">
    <rdf:type rdf:resource="#Contour_Related_Tool"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#compute_main_statistical_moments -->

<Texture_Related_Tool rdf:about="#compute_main_statistical_moments">
    <rdf:type rdf:resource="&owl;Thing"/>
</Texture_Related_Tool>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#compute_motion_vector -->

<Motion_Estimation_Tool rdf:about="#compute_motion_vector">
    <rdf:type rdf:resource="&owl;Thing"/>
</Motion_Estimation_Tool>
```



```
<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#compute_nor_RGB -->

<Colour_Features_Tool rdf:about="#compute_nor_RGB">
    <rdf:type rdf:resource="&owl;Thing"/>
</Colour_Features_Tool>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#compute_properties_of_concurrence -->

<owl:Thing rdf:about="#compute_properties_of_concurrence">
    <rdf:type rdf:resource="#Texture_Related_Tool"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#compute_ratio_area_convex_hull_over_blob -->

<Object_Tracking_Tool rdf:about="#compute_ratio_area_convex_hull_over_blob">
    <rdf:type rdf:resource="&owl;Thing"/>
</Object_Tracking_Tool>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#compute_rigid_point -->

<Contour_Related_Tool rdf:about="#compute_rigid_point">
    <rdf:type rdf:resource="&owl;Thing"/>
</Contour_Related_Tool>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#compute_spots_or_stripes -->

<Texture_Related_Tool rdf:about="#compute_spots_or_stripes">
    <rdf:type rdf:resource="&owl;Thing"/>
</Texture_Related_Tool>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#compute_trajectories -->

<owl:Thing rdf:about="#compute_trajectories">
    <rdf:type rdf:resource="#Motion_Estimation_Tool"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#compute_write_algae_presence_in_video -->

<owl:Thing rdf:about="#compute_write_algae_presence_in_video">
    <rdf:type rdf:resource="#Video_Classification_Tool"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#compute_write_fish_presence_in_video -->

<owl:Thing rdf:about="#compute_write_fish_presence_in_video">
    <rdf:type rdf:resource="#Video_Classification_Tool"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#compute_write_num_fish_in_frame -->

<Basic_Structures_and_Operations_Tool rdf:about="#compute_write_num_fish_in_frame">
    <rdf:type rdf:resource="&owl;Thing"/>
</Basic_Structures_and_Operations_Tool>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#compute_write_num_fish_in_video -->

<owl:Thing rdf:about="#compute_write_num_fish_in_video">
    <rdf:type rdf:resource="#Basic_Structures_and_Operations_Tool"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#compute_write_video_clearness -->

<Video_Classification_Tool rdf:about="#compute_write_video_clearness">
    <rdf:type rdf:resource="&owl;Thing"/>
</Video_Classification_Tool>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#compute_write_video_luminosity -->

<Video_Classification_Tool rdf:about="#compute_write_video_luminosity">
```



```
    <rdf:type rdf:resource="&owl;Thing"/>
</Video_Classification_Tool>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#create_IFD_model -->

<Create_Background_Model_Tool rdf:about="#create_IFD_model">
    <rdf:type rdf:resource="&owl;Thing"/>
</Create_Background_Model_Tool>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#create_W4_model -->

<Create_Background_Model_Tool rdf:about="#create_W4_model">
    <rdf:type rdf:resource="&owl;Thing"/>
</Create_Background_Model_Tool>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#create_adaptive_poisson_model -->

<owl:Thing rdf:about="#create_adaptive_poisson_model">
    <rdf:type rdf:resource="#Create_Background_Model_Tool"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#create_gaussian_bg_model -->

<owl:Thing rdf:about="#create_gaussian_bg_model">
    <rdf:type rdf:resource="#Create_Background_Model_Tool"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#create_gaussian_mixture_model -->

<owl:Thing rdf:about="#create_gaussian_mixture_model">
    <rdf:type rdf:resource="#Create_Background_Model_Tool"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#create_intrinsic_model -->

<owl:Thing rdf:about="#create_intrinsic_model">
    <rdf:type rdf:resource="#Create_Background_Model_Tool"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#create_moving_average_model -->

<Create_Background_Model_Tool rdf:about="#create_moving_average_model">
    <rdf:type rdf:resource="&owl;Thing"/>
</Create_Background_Model_Tool>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#create_poisson_model -->

<Create_Background_Model_Tool rdf:about="#create_poisson_model">
    <rdf:type rdf:resource="&owl;Thing"/>
</Create_Background_Model_Tool>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#create_waveback_model -->

<Create_Background_Model_Tool rdf:about="#create_waveback_model">
    <rdf:type rdf:resource="&owl;Thing"/>
</Create_Background_Model_Tool>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#detect_moving_objects -->

<owl:Thing rdf:about="#detect_moving_objects">
    <rdf:type rdf:resource="#Object_Detection_Tool"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#determine_presence_fish_blocking_screen -->

<owl:Thing rdf:about="#determine_presence_fish_blocking_screen">
    <rdf:type rdf:resource="#Basic_Structures_and_Operations_Tool"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#fourier_transform -->
```



```
<owl:Thing rdf:about="#fourier_transform"/>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#grab_frame_image -->

<Basic_Structures_and_Operations_Tool rdf:about="#grab_frame_image">
    <rdf:type rdf:resource="&owl;Thing"/>
</Basic_Structures_and_Operations_Tool>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#kanade_optical_flow -->

<Motion_Estimation_Tool rdf:about="#kanade_optical_flow">
    <rdf:type rdf:resource="&owl;Thing"/>
</Motion_Estimation_Tool>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#perform_morphological_operation -->

<Object_Detection_Tool rdf:about="#perform_morphological_operation">
    <rdf:type rdf:resource="&owl;Thing"/>
</Object_Detection_Tool>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#preliminary_analysis -->

<owl:Thing rdf:about="#preliminary_analysis">
    <rdf:type rdf:resource="#Basic_Structures_and_Operations_Tool"/>
</owl:Thing>


<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#preprocessing_and_initialisation -->

<Image_Formation_and_Preprocessing_Tool rdf:about="#preprocessing_and_initialisation">
    <rdf:type rdf:resource="&owl;Thing"/>
</Image_Formation_and_Preprocessing_Tool>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#rotation -->

<Geometric_Transformation_Tool rdf:about="#rotation">
    <rdf:type rdf:resource="&owl;Thing"/>
</Geometric_Transformation_Tool>


<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#scaling -->

<owl:Thing rdf:about="#scaling">
    <rdf:type rdf:resource="#Geometric_Transformation_Tool"/>
</owl:Thing>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#translation -->

<Geometric_Transformation_Tool rdf:about="#translation">
    <rdf:type rdf:resource="&owl;Thing"/>
</Geometric_Transformation_Tool>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#transposing -->

<Geometric_Transformation_Tool rdf:about="#transposing">
    <rdf:type rdf:resource="&owl;Thing"/>
</Geometric_Transformation_Tool>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#uniformity_high_and_low_bg_movement_
     and_fast_processing -->

<Domain_Descriptions_for_VIP_Tools rdf:about="#uniformity_high_and_low_bg_movement_and_fast_processing">
    <rdf:type rdf:resource="&owl;Thing"/>
</Domain_Descriptions_for_VIP_Tools>

<!-- http://www.owl-ontologies.com/Ontology1312300009.owl#update_moving_average_model -->

<owl:Thing rdf:about="#update_moving_average_model">
    <rdf:type rdf:resource="#Update_Background_Model_Tool"/>
```



```
    </owl:Thing>

    <!-- http://www.owl-ontologies.com/Ontology1312300009.owl#view_video -->

    <owl:Thing rdf:about="#view_video">
        <rdf:type rdf:resource="#Basic_Structures_and_Operations_Tool"/>
    </owl:Thing>
</rdf:RDF>

<!-- Generated by the OWL API (version 2.2.1.1138) http://owlapi.sourceforge.net -->
```

Appendix D: Process Library Prolog Code: Executables and Methods

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Process library for intelligent VIP system                                  %%
%% Gayathri Nadarajan                                                          %%
%% 30/05/08                                                                    %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
:- dynamic flag/2.

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% method(Name, Precond, Decomp, Effects, Postcond).
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% classify video
method(classify_video, [], [process_frames, perform_simple_classification,
write_frames_to_video], (nl), []).

% detect and count fish
method(detect_count_fish, [], [preliminary_analysis, process_each_frame_detection,
write_frames_to_video], (nl), []).

% classify video, detect and count fish
method(classify_video_detect_count_fish, [X =< Y], [preliminary_analysis, process_
each_frame, perform_classification, write_frames_to_video], (nl), []) :-
   frame_no(X),
   num_frames(Y).

% process_frames for classify video: recursive case
method(process_frames, [X =< Y], [grab_frame_image, compute_predominant_colours,
compute_main_texture_features, skip_frames, process_frames], (nl), []) :-
   frame_no(X),
   num_frames(Y).

% process_frames: base case -- do nothing when max no. frames reached
method(process_frames, [X > Y], [], (nl), []) :-
   frame_no(X),
   num_frames(Y).

% process_each_frame_detection - recursive case
method(process_each_frame_detection, [X =< Y], [grab_frame_image, compute_pre-
dominant_colours, compute_main_texture_features, perform_detection, count_fish_
frame, compute_write_num_fish_frame, skip_frames, process_each_frame_detection],
(nl), []) :-
   frame_no(X),
   num_frames(Y).

% process_each_frame_detection - base case
method(process_each_frame_detection, [X > Y], [], (nl), []) :-
   frame_no(X),
   num_frames(Y).

% process_each_frame tracking - recursive case
method(process_each_frame, [X =< Y], [grab_frame_image, compute_predominant_
colours, compute_main_texture_features, perform_detection, perform_tracking,
```



```
skip_frames, process_each_frame], (nl), []) :-
   frame_no(X),
   num_frames(Y).

% process_each_frame tracking - base case
method(process_each_frame, [X > Y], [], (nl), []) :-
   frame_no(X),
   num_frames(Y).

% compute main texture features
method(compute_main_texture_features, [], [compute_histogram, compute_main_
statistical_moments], (nl), []).

% skip frames - skip half the number of frames for fast processing
method(skip_frames, [performance(processing_time)], [], ((frame_no(X), num_frames
(Y), X1 is X+Z, retract(frame_no(X)), assert(frame_no(X1)))), []) :- Z is Y/2.

% skip_frames -- go to next frame if it is not blocked
method(skip_frames, [blocked(0)], [], ((frame_no(X), X1 is X+1, retract(frame_
no(X)), assert(frame_no(X1)))), []) :-
   write('Going to next frame --> '),write(X1),nl.

% skip_frames -- skip 4 frames if big fish blocking screen
method(skip_frames, [blocked(1)], [], ((frame_no(X), X1 is X+4, retract(frame_
no(X)), assert(frame_no(X1)))), []) :-
   nl, write('Big fish blocking screen. Skipping 4 frames'),nl.

method(skip_frames, [quality(reliability)], [], ((frame_no(X), X1 is X+1,
retract(frame_no(X)), assert(frame_no(X1)))), []).

% Perform detection
method(perform_detection, [frame_no(1)], [create_background_model, detect_objects_
blobs], (nl), []).

% Non-adaptive first frame, create bg model and detect blobs
method(detect_objects_blobs, [non_adaptive(true)], [detect_moving_objects,
detect_correct_blobs], (nl), []).

% Adaptive first frame, skip straight to detect and count
method(detect_objects_blobs, [non_adaptive(false)], [], (nl), []).

% For non-adaptive non-first frames, recreate bg model
method(perform_detection, [non_adaptive(true)], [create_background_model,
detect_moving_objects, detect_correct_blobs], (nl), []).

% MA model non-first frame, update bg model
method(perform_detection, [background_model(moving_average)], [update_moving_
average_model, detect_moving_objects, detect_correct_blobs], (nl), []).

% For adaptive models, bg model does not need to be updated.
method(perform_detection, [], [], (nl), []).

% If no single background model is suitable, fuse GMM and MA
method(create_background_model, [], [create_gaussian_mixture_model,
create_moving_average_model, fuse_bg_images], (nl), []).

% Perform Tracking
method(perform_tracking, [], [tracking_initialisation, compute_connected_compo-
nents, compute_blob_features, compute_closest_blob_rec, compute_write_num_fish_
frame, compute_write_num_fish_video, compute_fish_blocking_screen], (nl), []).

method(tracking_initialisation, [], [extract_hsv_values, compute_backprojection],
(nl), []).

method(compute_blob_features, [X > 10], [compute_camshift],
(assert(num_segments(10))), []) :-
   frame_no(X).

method(compute_blob_features, [], [compute_camshift], (assert(num_segments(Y))),
[]) :-
```



```
   frame_no(X),
   Y is X-1.

method(compute_closest_blob_rec, [num_segments(0)], [], (nl), []) :-
   write('Skipping compute closest blob, num segments 0'), nl, !.

method(compute_closest_blob_rec, [], [compute_closest_blob, compute_closest_blob_
rec], (nl), [Y >= 0]) :-
   retract(num_segments(X)),
   Y is X-1,
   assert(num_segments(Y)).

method(perform_simple_classification, [], [compute_write_luminosity,
compute_write_clearness, compute_write_presence_algae], (nl), []).

method(perform_classification, [], [compute_write_luminosity, compute_write_clear-
ness, compute_write_presence_algae, compute_write_presence_fish], (nl), []).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% 0. Primitive tasks: primitive/5 - determines primitive      %%
% tasks from list of independent executables                  %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
primitive(PP, Input, Preconds, Add_list_file, Postconds) :-
   independent_executable(PP, Fn_call, Preconds, Add_list_file, Input, Output,
   Postconds),
   write('Independent executable '), write(PP), write(' found '),nl.

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% independent_executable(User_terminology, Fn_call, Preconds_list, %%
%% Assert_list_file, Input_list, Output_list, Postconds_list)       %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% 1. View Video videos/1.mpeg 50
independent_executable(view_video, view_video, [], [], [Filename, Num_Frames],
[Displayed_video, Frame_rate, No_frames, Format, Date, Bg_images], []) :-
   input_file(File), atom_concat('videos/', File, Filename),
   num_frames(Num_Frames).

%% 2. Preliminary Analysis videos/1.mpeg
independent_executable(preliminary_analysis, preliminary_analysis, [], [Result_
file], [Filename], [Init_brightness, Init_clearness, Init_speed, Init_texture,
Percent_bg_obj, Noise_type], []) :-
   input_file(File), atom_concat('videos/', File, Filename),
   atom_concat('Prelim_', File, File1),
   atom_concat(File1, '.dat', Result_file).

%% 3. Grab Frame Image videos/1.mpeg
independent_executable(grab_frame_image, grab_frame_image, [], [], [Filename,
Fr_no], [Current_frame_image], []) :-
   input_file(File), atom_concat('videos/', File, Filename), frame_no(Fr_no).

%% 4. Extract RGB Colours Video_1.mpeg/2
independent_executable(extract_RGB_colours, extract_RGB_colours, [], [],
[Current_frame_image], [Red_image, Green_image, Blue_image, Yellow_image], []) :-
   input_file(I),
   frame_no(Fno), number_chars(Fno,Fno1),atom_chars(Fnum,Fno1),
   atom_concat('Video_',I,X),
   atom_concat(X,'/',Y), atom_concat(Y,Fnum,Current_frame_image).

%% 5. Compute Histogram Video_1.mpeg/2
independent_executable(compute_histogram, compute_histogram, [], [], [Current_
frame_image], [Histogram_value, Histogram_image], []) :-
          input_file(I), frame_no(Fno),
          number_chars(Fno,Fno1), atom_chars(Fnum,Fno1),
          atom_concat('Video_',I,X),
          atom_concat(X,'/',Y), atom_concat(Y,Fnum,Current_frame_image).

%% 6. Compute Main Statistical Moments Video_1.mpeg/2/Hist_Array_2.dat
independent_executable(compute_main_statistical_moments, compute_main_statisti-
cal_moments, [], [], [Hist_file], [Mean, Variance, Third_moment, Fourth_moment,
Entropy, Uniformity, Smoothness], []) :-
```



```
   input_file(I), frame_no(Fno),
   number_chars(Fno,Fno1), atom_chars(Fnum,Fno1),
   atom_concat('Video_',I,X),
   atom_concat(X,'/',Y), atom_concat(Y,Fnum,Current_frame_image),
   atom_concat(Current_frame_image,'/Hist_Array_',H1),
   atom_concat(H1,Fnum,H2), atom_concat(H2,'.dat',Hist_file).

%% 7. Compute Gabor Filter Video_1.mpeg/2.jpg Video_1.mpeg/ (3 45)
independent_executable(compute_Gabor_filter, compute_Gabor_filter, [], [],
[Current_frame_image, V_dir], [Gabor_filter], []) :-
   retrieve_video_dir(V_dir),
   retrieve_frame_image(Current_frame_image).

%% 8. Create Gaussian Background Model Video_1.mpeg/BackgroundImages/
independent_executable(create_gaussian_bg_model, create_gaussian_bg_model, [],
['gaussian.dat'],[Directory],[Bg_image1, Bg_image2],[background_model(gaussian)]):-
   retrieve_video_dir(V_dir),
   atom_concat(V_dir,'BackgroundImages/',Directory).

%% 9. Create Gaussian Mixture Model videos/1.mpeg Video_1.mpeg/GMM/ 50
independent_executable(create_gaussian_mixture_model, create_gaussian_mixture_model,
[], ['gaussian_mixture.dat'], [Filename, V_dir, NumFrames], [Bg_image],
[background_model(gaussian_mixture)]) :-
   input_file(File), atom_concat('videos/', File, Filename),
   retrieve_video_dir(V_dir),
   atom_concat(V_dir,'GMM/',GMM_dir),
   num_frames(NumFrames).

%% 10. Create Moving Average Model Video_1.mpeg/3.jpg Video_1.mpeg/Moving_Average/ 0.020
independent_executable(create_moving_average_model, create_moving_average_model,
[], ['moving_average.dat'], [Frame_image, MA_dir, Learning_speed], [Bg_image],
[background_model(moving_average)]) :-
   retrieve_frame_image(Frame_image),
   retrieve_video_dir(V_dir),
   atom_concat(V_dir,'Moving_Average/',MA_dir),
   Learning_speed = '0.020'.

%% 11. Create Intra-Frame Difference Model Video_1.mpeg/ Video_1.mpeg/IntraFrame-Difference/
independent_executable(create_intra_frame_difference_model, create_intra_frame_
model, [], ['intra_frame.dat'], [Dir, Output_dir], [Bg_matrix1, Bg_matrix2],
[background_model(ifd)]) :-
   retrieve_video_dir(Dir),
   atom_concat(Dir,'IntraFrameDifference/',Output_dir).

%% 12. Create W4 Model Video_1.mpeg/(Images/) Video_1.mpeg/W4/
independent_executable(create_w4_model, create_w4_model, [], ['w4.dat'],
[Dir, W4_dir], [Bg_matrix1, Bg_matrix2, Bg_matrix3], [background_model(w4)]) :-
   retrieve_video_dir(Dir),
   atom_concat(Dir,'W4/',W4_dir).

%% 13. Create Poisson Model Video_1.mpeg/Images/ Video_1.mpeg/Poisson/
independent_executable(create_poisson_model, create_poisson_model, [], ['poisson.
dat'], [Dir, Bg_dir], [Bg_matrix1, Bg_matrix2], [background_model(poisson)]) :-
   retrieve_video_dir(Dir),
   atom_concat(Dir,'Poisson/',Bg_dir).

%% 14. Create Adaptive Poisson Model videos/1.mpeg Video_1.mpeg/
independent_executable(create_adaptive_poisson_model, create_adaptive_poisson_model,
[background_model(adaptive_poisson)]) :-
   input_file(I),
   atom_concat('videos/', I, Video),
   retrieve_video_dir(V_dir).

%% 15. Update Moving Average Background Model Video_1.mpeg/2.jpg
independent_executable(update_moving_average_model, update_moving_average_model, [],
[], [Frame_image, Old_bg_image, Learning_speed], [Updated_bg_image], []) :-
   retrieve_frame_image(Frame_image),
   retrieve_video_dir(V_dir),
   atom_concat(V_dir,'Moving_Average/Background.jpg',Old_bg_image),
   Learning_speed = '0.020'.
```



```
%% 16. Fuse BG Images
independent_executable(fuse_bg_images, fuse_bg_images, [], [], [GMM_model, MA_model,
F_Dir], [Fused_image], []) :-
   retrieve_video_dir(V_dir),
   atom_concat(V_dir, 'GMM/Background.jpeg', GMM_model),
   atom_concat(V_dir, 'Moving_Average/Background.jpg',MA_model),
   retrieve_frame_dir(F_dir).

%% 17. Detect Moving Objects 2 Video_1.mpeg/IntraFrameDifference/intraFrame.jpg
Video_1.mpeg/2.jpg Video_1.mpeg/2/
independent_executable(detect_moving_objects, detect_moving_objects, [], [],
[Bg_model_type, Bg_frame_image, Frame_image, F_dir], [BW_image_with_blobs], []) :-
   get_bg_model(Bg_model_type),
   retrieve_frame_dir(F_dir),
   retrieve_frame_image(Frame_image),
   load_bg_model_image(Bg_model_type, Bg_frame_image).

%% 18. Perform Morphological Operation Video_1.mpeg/2/IntraFrameDifference/Binary_
Image.jpeg Video_1.mpeg/2/
independent_executable(detect_correct_blobs, detect_correct_blobs, [], [],
[BW_image_with_blobs, F_dir], [Img_potential_correct_blobs], []) :-
   get_bg_model(T),
   retrieve_frame_dir(F_dir),
   atom_concat(F_dir, 'Binary_Image.jpeg', BW_image_with_blobs).

%% 19. Extract HSV Values Video_1.mpeg/2.jpg Video_1.mpeg/2/
independent_executable(extract_hsv_values, extract_hsv_values, [], [], [F_image,
F_dir], [Hue_image, Saturation_image, Value_image], []) :-
   retrieve_frame_image(F_image),
   retrieve_frame_dir(F_dir).

%% 20. Compute Backprojection Video_1.mpeg/2/Hue.jpg Video_1.mpeg/2/
independent_executable(compute_backprojection, compute_backprojection, [], [],
[Hue_image, F_dir], [Hue_backprojection_image], []) :-
   retrieve_frame_dir(F_dir),
   atom_concat(F_dir, 'Hue.jpg', Hue_image).

%% 21. Compute Connected Components Video_1.mpeg/2/CorrectBlobs.jpeg Video_1.mpeg/2/
independent_executable(compute_connected_components, compute_connected_components,
[], [], [BW_image_with_blobs, F_dir], [Image_with_blobs, Num_blobs], []) :-
   retrieve_frame_dir(F_dir),
   atom_concat(F_dir, 'CorrectBlobs.jpeg', BW_image_with_blobs).

%% 22. Compute Camshift and Ratio Video_1.mpeg/2.jpg Video_1.mpeg/2/BlobsBW.jpeg Video_1.mpeg/2/
independent_executable(compute_camshift, compute_camshift, [], [],
[F_img, Image_with_blobs, F_dir], [Centre, Orientation, Size], []) :-
   retrieve_frame_dir(F_dir),
   retrieve_frame_image(F_img),
   atom_concat(F_dir,'BlobsBW.jpeg', Image_with_blobs).

%% 23. Compute Closest Blob: Video_1.mpeg/1/ Video_1.mpeg/2/
independent_executable(compute_closest_blob, compute_closest_blob, [], [], [F_dir1,
F_dir2, Num_blobs1, Num_blobs2], [Minimum_euclidian_distance], []) :-
   retrieve_frame_dir(F_dir1),
   num_segments(Seg),
   frame_no(X), Y is X-Seg,
   retrieve_video_dir(V_dir),
   atom_concat(V_dir,Y, Y1),
   atom_concat(Y1, '/', F_dir2).

%% 24. Compute and Write Number of Fish in Frame and Video: Video_1.mpeg/2/
Video_1.mpeg/2fish.dat Video_1.mpeg/2.jpg Video_1.mpeg/ 2.jpg (true)
independent_executable(compute_write_num_fish_frame, count_write_num_fish_frame,
[], [], [F_dir, Blob_file, F_img, V_dir, F_name], [Num_fish_frame, Blob_counted,
Final_frame_image], []) :-
   retrieve_frame_dir(F_dir),
   retrieve_frame_image(F_img),
   frame_no(X), number_chars(X,XC), atom_chars(XA,XC),
   retrieve_video_dir(V_dir),
```



```
   atom_concat(V_dir, XA, FXA), atom_concat(FXA,'fish.dat', Blob_file),
   atom_concat(XA,'.jpg',F_name).

%% 25. Determine Presence of Fish Blocking Screen Video_1.mpeg/3/CorrectBlobs.jpeg
Video_1.mpeg/3/
independent_executable(compute_fish_blocking_screen, compute_fish_blocking_screen,
[], [Screen_blocked_file], [Image_with_blob, F_dir], [Presence], []):-
   retrieve_frame_dir(F_dir),
   atom_concat(F_dir, 'CorrectBlobs.jpeg', Image_with_blob),
   atom_concat(F_dir, 'block.dat', Screen_blocked_file).

%% 26. Compute and Write Average Luminosity Video_1.mpeg/
independent_executable(compute_write_luminosity, compute_write_luminosity, [],
['brightness.dat'], [V_dir], [Final_frame_image], []) :-
   retrieve_video_dir(V_dir).

%% 27. Compute and Write Average Clearness Video_1.mpeg/
independent_executable(compute_write_clearness, compute_write_clearness, [],
['clearness.dat'], [V_dir], [Final_frame_image], []) :-
   retrieve_video_dir(V_dir).

%% 28. Compute and Write Presence of Fish Video_1.mpeg/
independent_executable(compute_write_presence_fish, compute_write_presence_fish,
[], [], [V_dir], [Final_frame], []) :-
   retrieve_video_dir(V_dir).

%% 29. Compute and Write Presence of Algae Video_1.mpeg/
independent_executable(compute_write_presence_algae, compute_write_presence_algae,
[], ['green.dat'], [V_dir], [Final_image], []) :-
   retrieve_video_dir(V_dir).

%% 30. Write to (Final) Output Video Final_1.mpeg.avi Video_1.mpeg/Output/
independent_executable(write_frames_to_video, write_frames_to_video, [], [],
[Final_video, Images_dir], [], []) :-
   input_file(I),
   atom_concat('Final_', I, FV),
   atom_concat(FV, '.avi', Final_video),
   retrieve_video_dir(V_dir),
   atom_concat(V_dir, 'Output/', Images_dir).
```

References

[1] D. McGuinness and F. van Harmelen. *OWL Web Ontology Language*. World Wide Web Consortium (W3C), 2004. `http://www.w3.org/TR/owl-features`.

[2] G. Nadarajan, Y. H. Chen-Burger, and R. B. Fisher. A Knowledge-Based Planner for Processing Unconstrained Underwater Videos. In *IJCAI'09 Workshop on Learning Structural Knowledge From Observations (STRUCK'09)*, 2009.

[3] G. Nadarajan, Y. H. Chen-Burger, and R. B. Fisher. SWAV: Semantics-based Workflows for Automatic Video Analysis. In *Special Session on Intelligent Workflow, Cloud Computing and Systems, (KES-AMSTA'11)*, 2011.

[4] G. Nadarajan, C. Spampinato, Y. H. Chen-Burger, and R. B. Fisher. A Flexible System for Automated Composition of Intelligent Video Analysis. In *7th International Symposium on Image and Signal Processing and Analysis (ISPA'11)*, 2011.

[5] K. T. Shao. *Fish Database of Taiwan*. Research Center for Biodiversity Academia Sinica, 2011. `http://fishdb.sinica.edu.tw/`.

