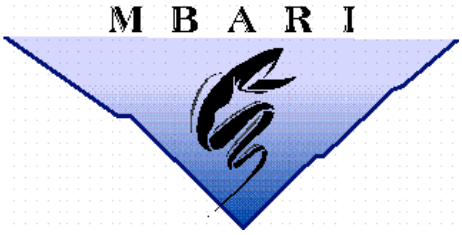
Monterey Bay Aquarium Research Institute



Demonstration Study on Applying AVED to Still Images from Station M

An Automated Video Event Detection and Classification System

Functional Requirements Specification

Version 0.1

Revision History

0.1 12 May 2009 - DEC initial draft

Automated Event Detection Software
Requirements Specification

Figure and Tables

1. Introduction

The purpose of the automated video event detection and classification system for analyzing images from the Station M site is to provide a software based system to automatically locate objects (events) in the time-lapse images and subsequently classify them into their respective genus or family. This system is intended to be a computer-aided assistant for the Ken Smith science lab and the final prototype will be handed over the Ken Smith lab for use in day-to-day use. As a minimum, this system will create a stream of data that can be assimilated by the existing tools used in the Smith lab for post deployment analysis.

1.1. Purpose

The purpose of this document is to identify the high-level functional requirements that will guide the MBARI design and development of this prototype system. This document is intended not only for the benefit of the designers and implementers of the prototype, but also for users of the system.

1.2. Scope

These requirements are confined to the automated video detection system and classification system as applied to the Station M images. It does not address requirements of any other system except as they may relate to this system. Specifically, this document does not detail the capabilities of the AVED system which can be found in the "AVED Classifier User Interface Requirements", and "AVED Software Requirements Specification" Documents described in section 1.4.

It is not the purpose of this specification document to discuss or describe how specific requirements are to be met, but rather describe the requirements of users as well as software requirements that need to be met for the application to fulfill its role as prototype system.

There may, however, be cases where non-functional requirements are specified, due to MBARI convention.

1.3. Acronyms

The following is a list of acronyms provided for the reader's convenience:

OOP	Object Oriented Programming
VARS	Video Annotation and Reference System
AVED	Automated Video Event Detection project or process
TBD	To Be Determined
UTC	Coordinated Universal Time
OS	Operating System
PPM	Portable Pixel Map
PNG	Portable Network Graphics

1.4. References

The primary documents that are referred to in this document:

1) IEEE Std 830-1998 Documentation Standard, specifically the section on *Recommended Practice for Software Requirements Specifications*. This is the reference guide that describes the form of this document.

2) "AVED Classifier User Interface Requirements version 0.4". Document

3) "AVED Software Requirements Specification version 0.6" Document

4) "Requirements Engineering", Merlin Dorfman, *SEI Interactive*, 03/99, Requirements documents for the VARS system.

1.5. Overview

The sections below begin with an overall description of the factors that led to the requirements for an automated video event detection tool for this project.

The first section describes the overall components of the system, and is followed by a section containing all of the user and software requirements to a level of detail sufficient to enable designers to design a prototype system to satisfy those requirements. This section includes specific performance requirements as well as constraints on the design.

2. Overall description

A goal of this effort is to apply the AVED software to demonstrate how computer aided vision can be used to analyze Station M images. Our goal for this demonstration is to ultimately transfer this work to the Smith lab as a tool that they will continue to use in their research. We also intend to make the work available to other MBARI scientists and the general public. Part of the motivation for this work is the Station M images are an ideal candidate because the images are from a controlled environment; support a discrete biological community, and have nominal variance in lighting; all of which make a great application of this technology. More details on the motivation of this work can be found in the project proposal document noted in References section.

2.1. Possible use cases

The following section describes the possible use cases for this system. Some of these cases were derived from the original AVED software specifications document and are not meant to be comprehensive, but rather, as a starting place for discussion. All of these use cases require the same core capability.

As a reminder, this prototype system is based on the AVED software that provides two core capabilities: the ability to find interesting events in a sequence of frames (detection/ tracking), and the ability to classify these interesting events.

The users of this system are assumed to be a single MBARI science user in the Smith lab.

One possible use case could be:

- Process a collection around major El Niño/La Niño events between 1997, searching for a few select animals targeted to ground truth against computer vision techniques employed in AVED.

Other use cases could be:

- To assist in finding targeted animals during normal science annotation sessions, leaving the annotator time to focus on other annotations.

- Find animals that were annotated incorrectly, or missed entirely on previously annotated dives.

- To look for specific behaviors that might otherwise be difficult to capture. (This might require the use of ancillary data to provide contextual information.)

2.1.1. The AVED Data Flow

In the case of using AVED as an annotation assitant to automate the the search for events, the still image sequences would be recorded in digital format, then sent to the *AVED Process* where events of interest are detected and tracked. Following *AVED* processing, event images (cropped images of events) are then sent to the *Classification Process* to . Classifier results results are then merged with the AVED detection results. The final AVED detection results are then sent back to the Science Annotator in its raw XML format for integration with an existing data mangement system. If the user wanted to optionally edit the results, they could do so through the AVED editor.

The following diagrams illustrate the data flow in this case.

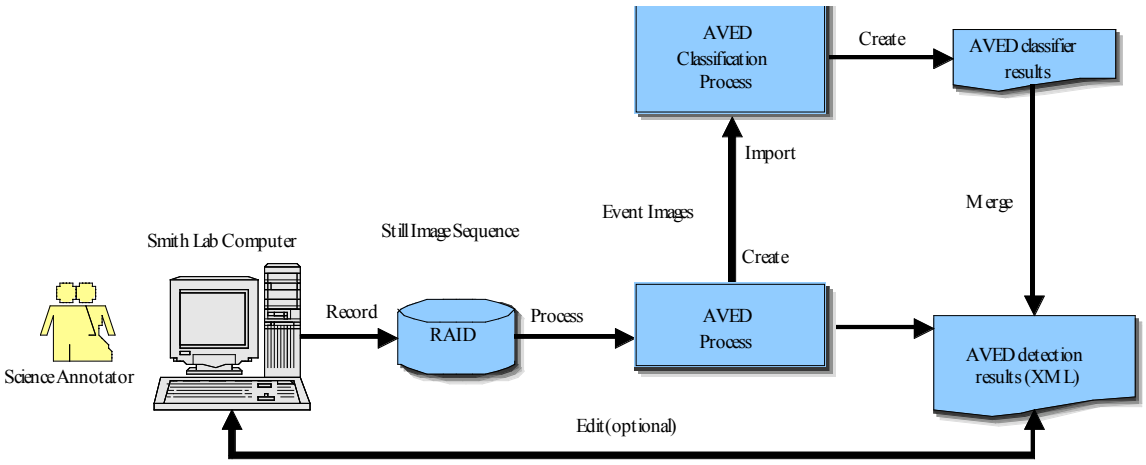


Figure 1. Data flow

2.1.2. General design

The system is to be designed and implemented reusing as much MBARI hardware and software technologies as possible. In cases where new software must be developed, it should be designed using modern, maintainable technology, such as an object oriented programming language (OOP), which is supported over the useful life of the system. The system is to be well documented and to include written test and maintenance procedures.

2.1.3. Computation

The greatest constraint on the system is the computational requirements for executing the AVED saliency-based algorithm on imagery. This impact of this constraint depends on how much data is to be processed by this system.

2.1.4. Lifespan

The project is expected to have a useful operational lifetime of over ten years. Therefore the software must be designed and implemented using tools and technologies that provide a stable development and maintenance environment over this time period.

2.2. Assumptions and dependencies

A primary assumption is that the system will interface with AVED software and provide the image sequences and required metadata in a format acceptable to the system.

The current assumption is that a single high-end computer server will be adequate to support the processing requirements demanded by this system. This high-end compute server must run the entire AVED software suite.

For executing the user interfaces, the current assumption is that the user will be using a Mac computer running OS X Tiger or Leopard. (TBD need OS version here).

For the remainder of this document, the user of this system is assumed to be the science user.

3. System input requirements

3.1. Still frame input and buffering

The input to the AVED system must be in digitized form for the saliency algorithm. The saliency algorithm processes one frame at a time, so the video input and buffering must support this frame-based system.

3.1.1. Frame input format

The digitizing system shall encode frames in 24-bit RGB formatted PNG or PPM frames.

Rationale

The saliency algorithm requires data in 24-bit RGB color space. PPM or PNG formats support this RGB format and are supported by the saliency code so using

this format leverages the existing code. These formats are also widely supported, documented, and easy to understand frame formats making them a good choice.

3.2. Automated visual event detection and tracking

The AVED software shall be used to automatically find objects (events) of interest and track them over time. Specific details on the AVED software capability can be found in the AVED software requirements document (see Reference section 1.4 above).

3.2.1. Editing the results

3.2.1.1.AVED Editor

The prototype system shall use the AVED editor, which is a graphical interface designed to assist the user in displaying and editing AVED results. Please note that the editor does not include classification results - just the detection results.

Rationale
The AVED algorithm sometimes finds "false detection". AVED events that are tracked over time can "lose" track for few frames. This editor can be used to e.g. delete false detections, or combine multiple events together for form a single event.

3.2.2. Automated processing setup and control

3.2.2.1.Processing setup

3.2.2.1.1. AVED detection processing options shall be defined by the user through an environment variable in the form of command-line switches (e.g. –mbari-cache-size=2, etc.). There are over a hundred different processing options that can be used with AVED, but the fewer more commonly used options are documented here: https://oceana.mbari.org/confluence/display/AVED/AVED++Mbarivision+Options .

3.2.2.2.Processing control

3.2.2.2.1. The user through a simple script shall initiate processing control.

3.2.3. Tracking

Events shall be tracked over frames when possible.

Rationale
The AVED tracking system is designed to work with video frames only, but may track objects that remain relatively stationary over the course of the time-lapse imagery, and is therefore included.

A tracked event shall be one that is satisfies the requirements:

3.2.3.1. A tracked event is a sequence of objects persisting at least three frames between three and the *maximum event duration*, and bounded by the *minimum* and *maximum event area* specified by the user (see Table 1. in the AVED requirements specification document for more details).

3.2.3.2. When an event exceeds the maximum duration defined by the user programmed m*aximum event duration*, it shall be reset and tracked as new event (see Table 1. in the AVED requirements specification document for more details).

Rationale
This is somewhat arbitrary, but a long or indefinite event sequence would be difficult to manage, and is therefore bounded to make data management easier. This may not be applicable for this application but is included for thought.

3.2.3.3. The system shall detect and track between one and a user specified number of objects in any given frame (see Table 1. in the AVED requirements specification document for more details).

3.3. Automated event classification

It is assumed in this prototype, the AVED classifier shall be used to identify events for science study. Functional requirements of the classifier are defined as follows:

3.3.1. Classification definitions

Classification can be confusing term so this section will define what these terms mean for this prototype.

3.3.1.1. Classification learning is the process of learning a discrete-valued function from discrete inputs (images in this case).

3.3.1.2. Learning in this context is done through a "training" phase, whereby the discrete input images (training class) shall be defined as a collection of images that represent a particular genus of species.

Rationale
The foundation of the classifier is a collection of images files. Each image file collection represents a "class" which typically represents a single animal in its various sizes and orientations, e.g. a Rathbunaster class, Poeobius class, etc. A class is defined as genus or species here because the AVED classifier will work for this resolution. To be clear, this does not mean the AVED classifier will not work for e.g. family. It has, however, not been tested this way would probably require more effort than time allows for in this prototype.

3.3.2. Editing classes

3.3.2.1. A graphical tool to create, edit, and generally manage training classes shall be provided to the users.

Rationale:
The AVED classifier requires a set of images to train for each class. It is assumed that the training classes will need to be modified; therefore a tool is required to maintain those images. This tool would allow the user to define new classes, delete classes, or similar functions as we learn more about what works best.

3.3.3. Classifier setup

3.3.3.1. The classifier user interface shall allow the user to select what training classes, or collections of classes to test against.

Rationale:
The training classes are what a classifier searches for. The user may want to only search for individual things, or collections of things.

3.3.3.2. The classifier user interface shall allow the user to save a collection of classes into a user-defined profile that can later be reloaded.

Rationale:
The user may want to customize different profiles, e.g. for benthic versus midwater. This allows some flexibility for the users.

3.3.3.3. The classification software shall include the ability to periodically validate an individual class and preview both its correct and misclassification percentages.

Rationale:
The current validation scheme is to test the class by running the classifier on the collection of images in a particular class, against the class itself. The results of the validation include correct and misclassification rates which are indicators of how good a training class is, e.g. a 95% correct classification 5% misclassification would indicate the class is a good discriminate. This data would be required to know the efficacy of the classifier and would likely be noted in published results.

3.3.4. Classifier output

3.3.4.1. Output from the classifier shall include a class name, class probability that correlates to an input image in a comma-delimited file.

3.4. Data products and storage

3.4.1. Data listed in **Table 1** shall be stored in a directory the user defines. TBD – would you rather have this data in a database?

3.4.2. Filenames formatted according to UTC day and time convention here: **http://en.wikipedia.org/wiki/ISO_8601** will propagate the correct UTC timestamp on the AVED results.

Rationale:
This capability already exists in the AVED scripts and would be easily reused in this application.

3.4.3. The following table describes the data products.

Table 1. Data products display and storage matrix

Data product	Format	Description
AVED classifier results	Comma-delimited	AVED classifier results, including still image reference, class name, and class probability, class voting method.
AVED results	XML	AVED event metadata results for every frame. See AVED XML for more information.

3.5. Exchanging data with other applications

This primary output from this prototype will be AVE D XML files that can be assimilated by the existing tools used in the Smith lab for post deployment analysis. More details on the AVED XML and schema can be found here: https://oceana.mbari.org/confluence/display/AVED/AVED+XML+Schema+and+Example