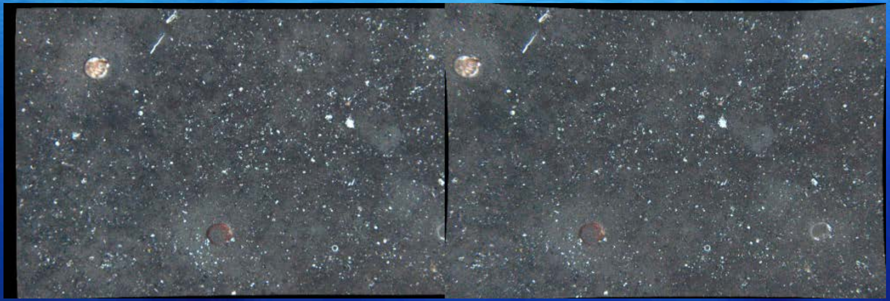
VIAME



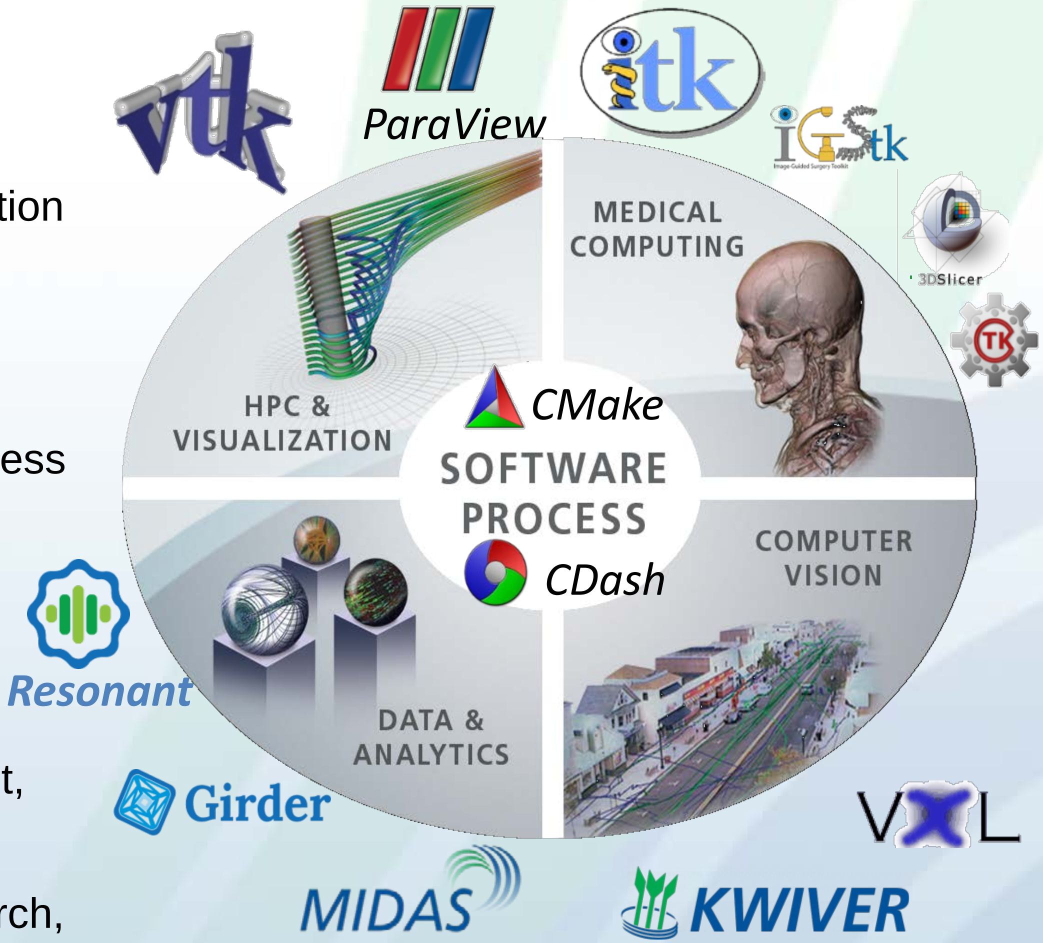
VIAME: An Open-Source Framework for Underwater Image and Video Analytics

Matthew Dawkins[1], Linus Sherrill[1], Keith Fieldhouse[1], Anthony Hoogs[1], Benjamin Richards[2], David Zhang[3], Jon Crall[1], Lakshman Prasad[4], Nathan Lauffenburger[2], Gaoang Wang[5]

[1]Kitware Inc., [2]National Oceanic and Atmospheric Administration, [3]SRI International, [4]Los Alamos National Lab, [5]University of Washington

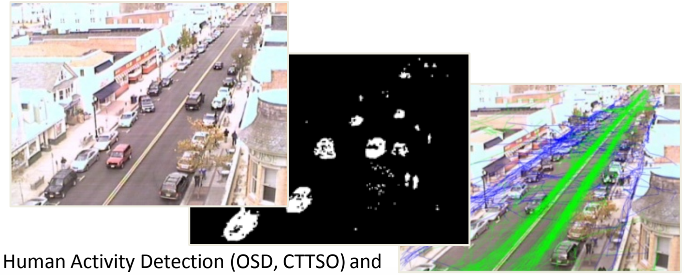
Kitware Open Source Platforms

- **KWIVER** Kitware Imagery and Video Exploitation and Retrieval
- **VTK** the visualization toolkit
- **ParaView** large data analysis & visualization application
- **ITK** insight image analysis toolkit
- **CMake** cross-platform build system
  - CDash, CTest, CPack, software process tools
- **Resonant/Girder** informatics and information visualization
- **Kiwi & VES** mobile visualization
- IGSTK, CTK, vxl, Open Chemistry Project, VolView, tubeTk, and more…
- **MIDAS** for computational scientific research, testing, and visualization

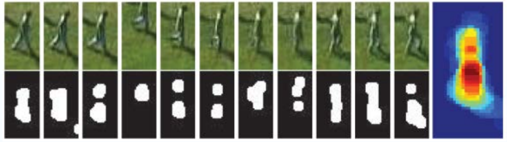


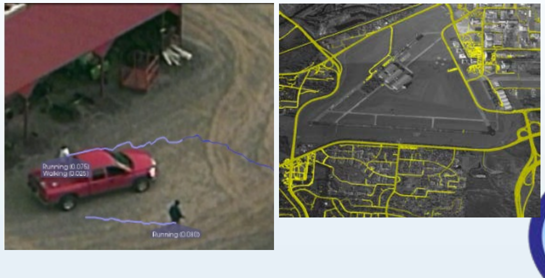
- 25+ team members
- 12 PhDs
- Founded in 2007
- 35+ contracts

Dr. Anthony Hoogs
anthony.hoogs@kitware.com
518-881-4910

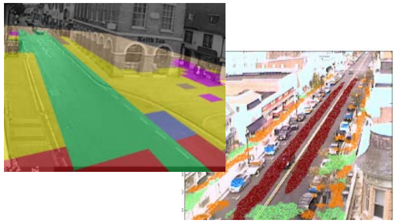


Human Activity Detection (OSD, CTTSO) and Tracking in Wide-Area Video (AFRL)



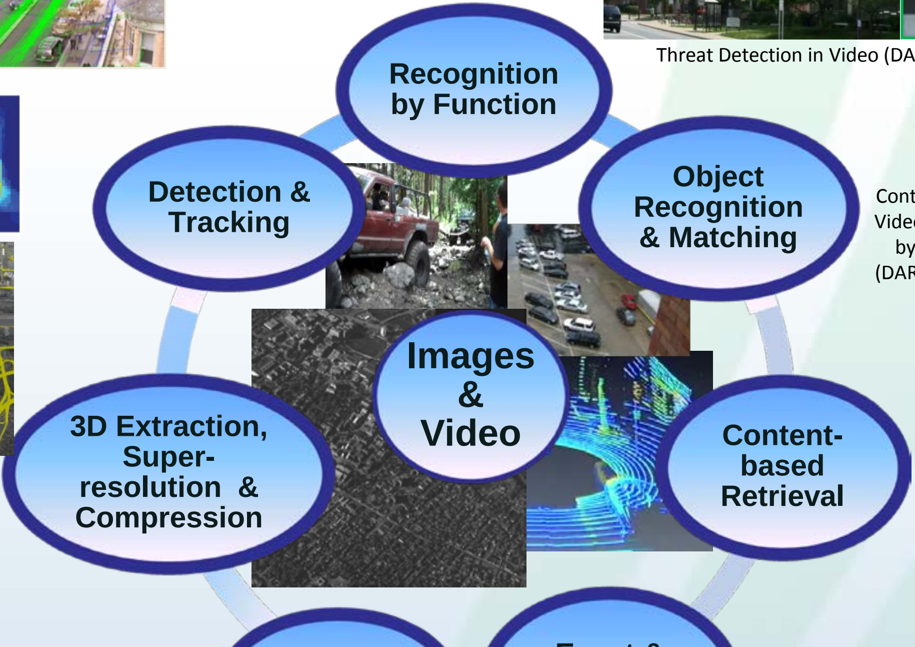


Object and Building Recognition by Function (DARPA)



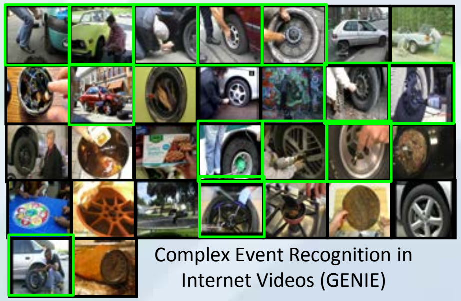
Threat Detection in Video (DARPA)

Content-based Video Retrieval by Actions (DARPA VIRAT)



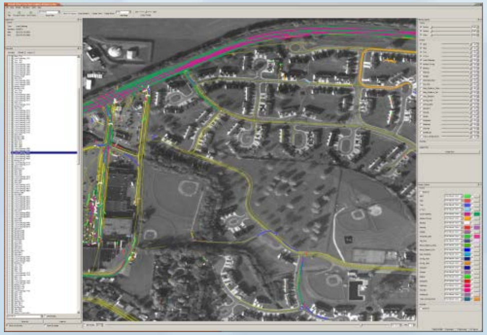
Anomaly Detection

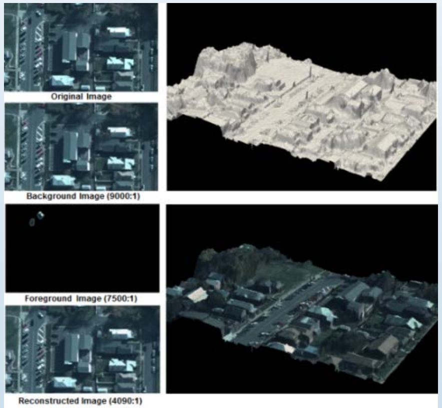
Event & Activity Recognition



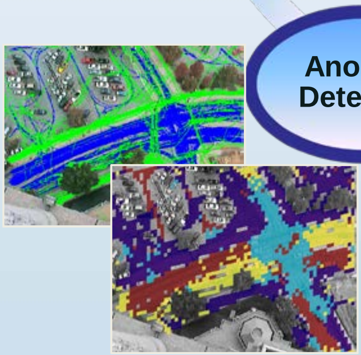
Complex Event Recognition in Internet Videos (GENIE)

Wide-Area Motion Imagery Event, Anomaly and Activity Detection (OSD Data to Decisions, DARPA PerSEAS)



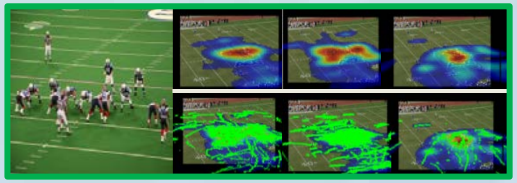

Original Image
Background Image (9000:1)
Foreground Image (7500:1)
Reconstructed Image (4090:1)

3D model-based video compression (DARPA) and super-resolved 3D reconstruction (DARPA)



Normalcy Modeling and Anomaly Detection (DARPA PANDA and PerSEAS)

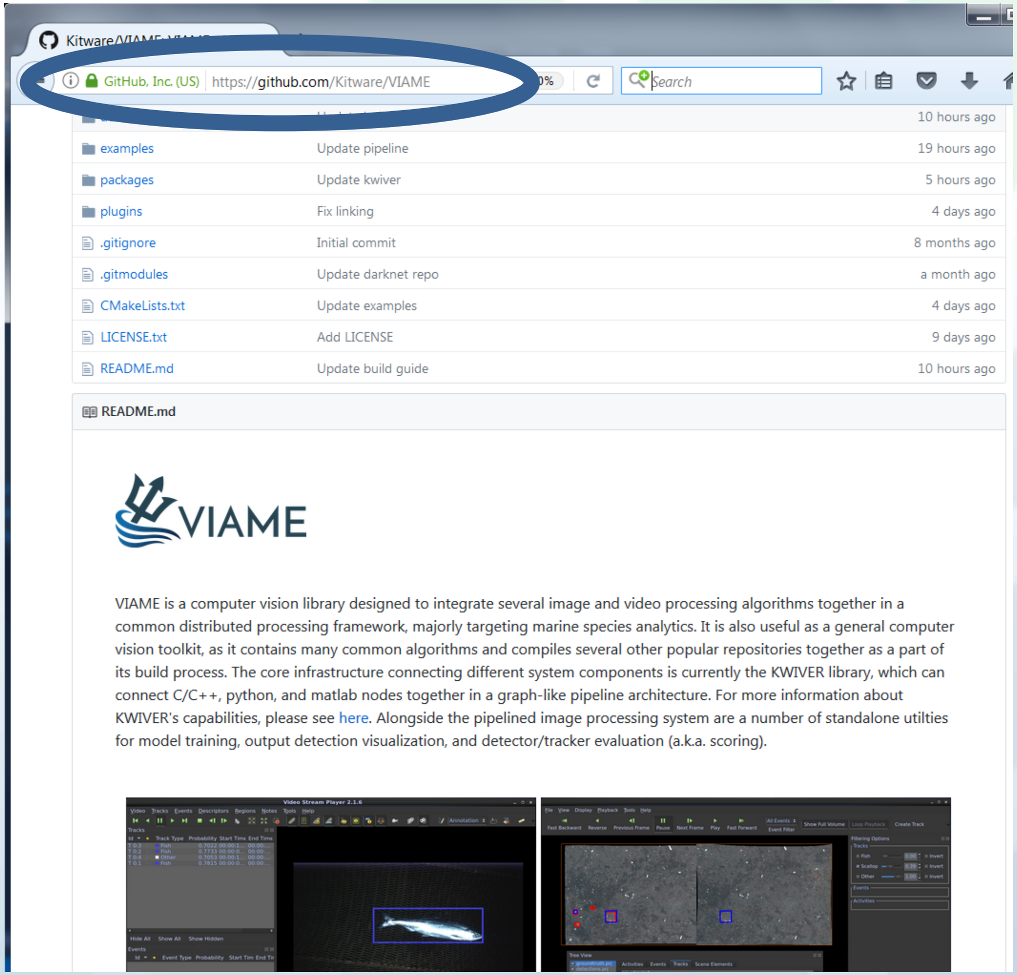
Football Play Recognition (DARPA CARVE)



VIAME

https://github.com/Kitware/VIAME

- **V**ideo and **I**mage **A**nalytics for the **M**arine **E**nvironment
- Goal: Develop an open-source software platform for NMFS image and video analysis
  - In close coordination with NOAA community



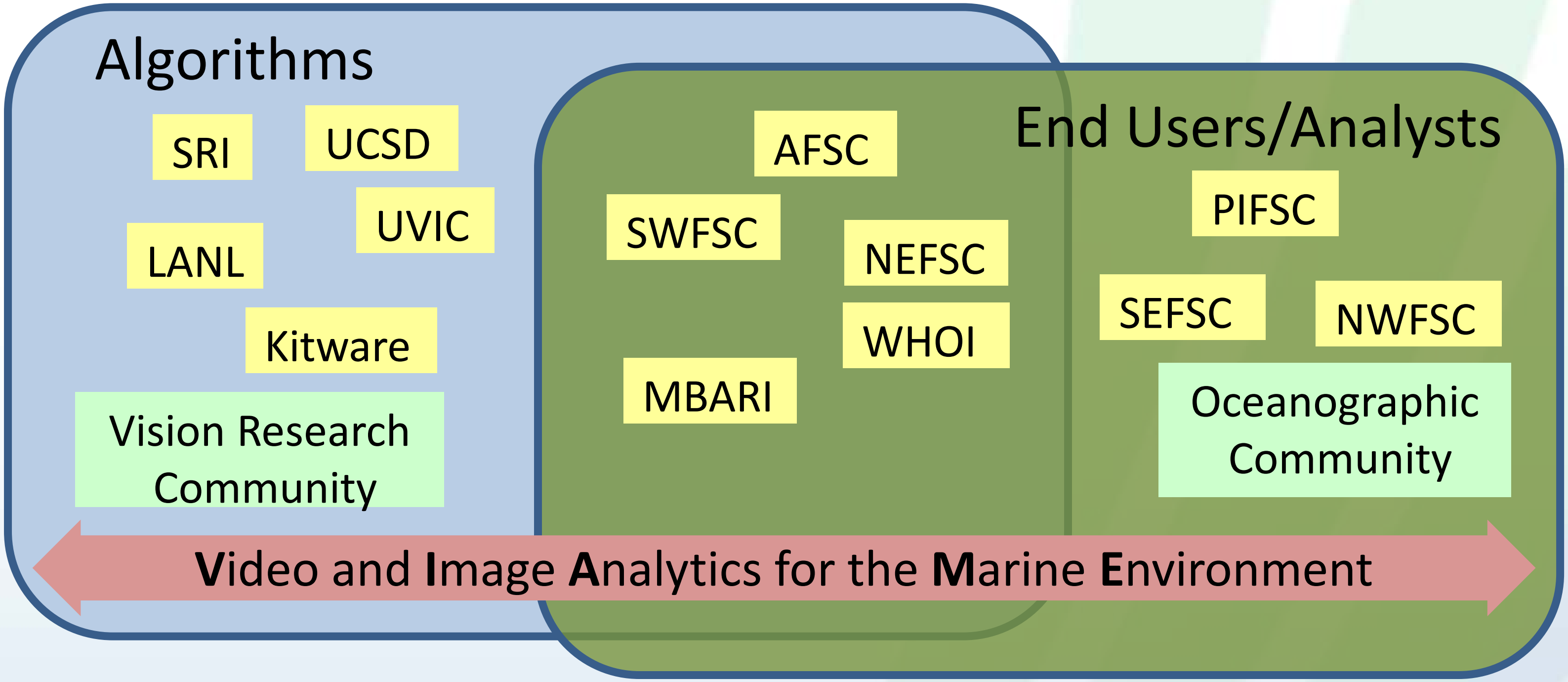
[illegible]	[illegible]	10 hours ago
examples	Update pipeline	19 hours ago
packages	Update kwiver	5 hours ago
plugins	Fix linking	4 days ago
.gitignore	Initial commit	8 months ago
.gitmodules	Update darknet repo	a month ago
CMakeLists.txt	Update examples	4 days ago
LICENSE.txt	Add LICENSE	9 days ago
README.md	Update build guide	10 hours ago

README.md

VIAME

VIAME is a computer vision library designed to integrate several image and video processing algorithms together in a common distributed processing framework, majorly targeting marine species analytics. It is also useful as a general computer vision toolkit, as it contains many common algorithms and compiles several other popular repositories together as a part of its build process. The core infrastructure connecting different system components is currently the KWIVER library, which can connect C/C++, python, and matlab nodes together in a graph-like pipeline architecture. For more information about KWIVER's capabilities, please see here. Alongside the pipelined image processing system are a number of standalone utilties for model training, output detection visualization, and detector/tracker evaluation (a.k.a. scoring).

Lots of Data, Analytics and Users



Algorithms

SRI

UCSD

UVIC

LANL

Kitware

Vision Research Community

AFSC

SWFSC

NEFSC

WHOI

MBARI

End Users/Analysts

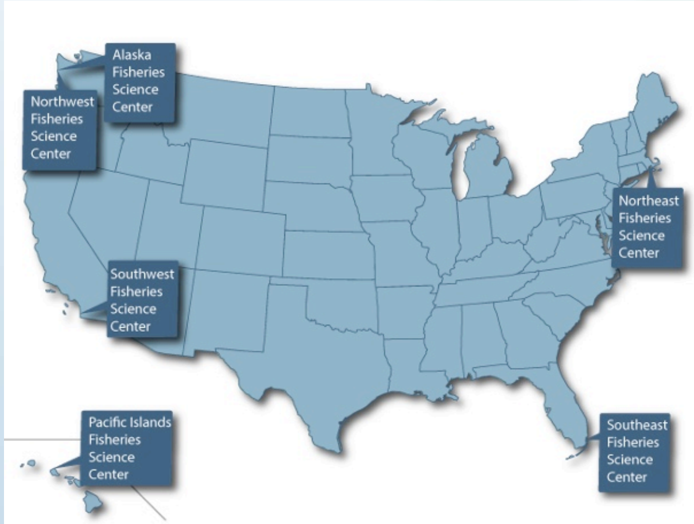
PIFSC

SEFSC

NWFSC

Oceanographic Community

Video and **I**mage **A**nalytics for the **M**arine **E**nvironment


Alaska Fisheries Science Center
Northwest Fisheries Science Center
Northeast Fisheries Science Center
Southwest Fisheries Science Center
Pacific Islands Fisheries Science Center
Southeast Fisheries Science Center

Current Image/Video Analytics

Capability	Primary data source	POC	Stereo calibration	Stereo processing	Video	Color, contrast correction	scallop detection	fish detection	fish length, sizing	fish tracking	fish classification	anomaly det.	habitat classification	image segmentation
NW SC CamTrawl	Cam Trawl	Williams	🟩 yes	🟩 yes	4 Hz	no, grayscale		🟩 yes	🟩 automatic	🟨 yes	🟩 yes			
ROV video fish detection and tracking	SWFSC ROV video	Cutter	no	no	30 Hz	🟨 yes		🟩 yes, DPM (UW)	no	🟥 yes (UW student)	▨ desired	▨ desired		
ROV stereo fish measurement	SWFSC ROV GigE stereo	Cutter	🟩 yes	🟩 yes	2-4 Hz	🟨 yes		no	manual	no				
WHOI/NEFSC scallop detector	HABCAM towed rig	Dvora	🟩 yes	🟩 yes	no	🟩 yes	🟩 yes							
RPI/Kitware scallop detector	HABCAM towed rig	Hoogs	no	no	no	🟩 yes	🟩 yes							
SRI fish detection, classification, size	PI FSC MOUSS/BotCam	Ben/Mike	🟩 yes, accept cal files	🟨 yes	30 Hz	no, grayscale		🟩 yes	▨	🟩 yes	🟨 yes			
SEFSC stereo proc	Drop cams from SEFSC	Thompson	🟩 yes	🟩 yes	yes			🟨 yes, basic background	manual	no	no			
Toyon SBIR I	Drop cams from SEFSC	Thompson	🟩 yes	🟩 yes	yes			🟨 yes, basic HOG	manual	🟨 yes	🟥 yes			
LANL segmentation and shape analysis	HABCAM towed rig	Lakshman	no	▨ yes	no	no	🟥 yes	🟥 yes	no	no	▨ yes	🟨 yes (image)	▨ Yes	🟩 yes (polygonal)
Toyon SBIR II	Still Images AUV, drop, towed	Clarke	🟩 yes	🟥 yes	no	🟩 yes (Hanu)		🟨 yes	▨ yes	no	▨ yes			
WHOI/NEFSC habitat classifier	HABCAM towed rig	Dvora	🟩 yes	🟩 yes	no	🟩 yes							🟨 yes	
NWFSC clustering	AUV and MOUSS	Clarke	no		no	no						🟨 yes	🟨 partially	

🟩 Green	well-implemented; quantified, comparative performance assessment; ready for integration	🟨 Yellow	Existing implementation as mature research code; some performance quantification	🟥 Red	preliminary research code with ongoing work against major problems	▨ Gray	idea or concept; no implementation

VIAME High-Level Components

Multi-Processing Pipeline Framework

Calibration and Stabilization

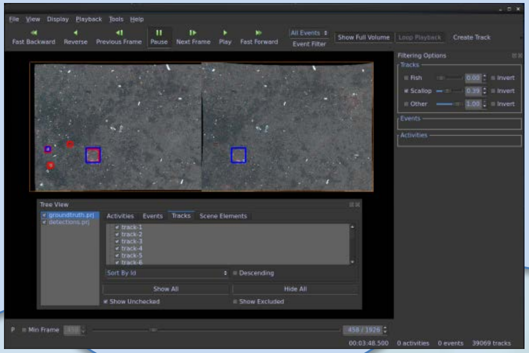
Detection and Tracking

Classification

Other Analytics

Data Abstraction and Conversion
(not a native DB)

Web based
services and tools



Fish 0.00 Invert
☑ Scallop 0.39 Invert
Other 1.00 Invert

☑ groundtruth.prj
☑ detections.prj

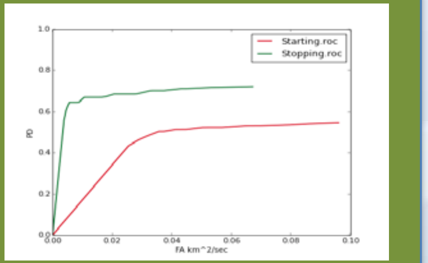
☑ track-1
☑ track-2
☑ track-3
☑ track-4
☑ track-5
☑ track-6

Sort By Id
Descending
☑ Show Unchecked
Show Excluded

Min Frame 458 / 1926

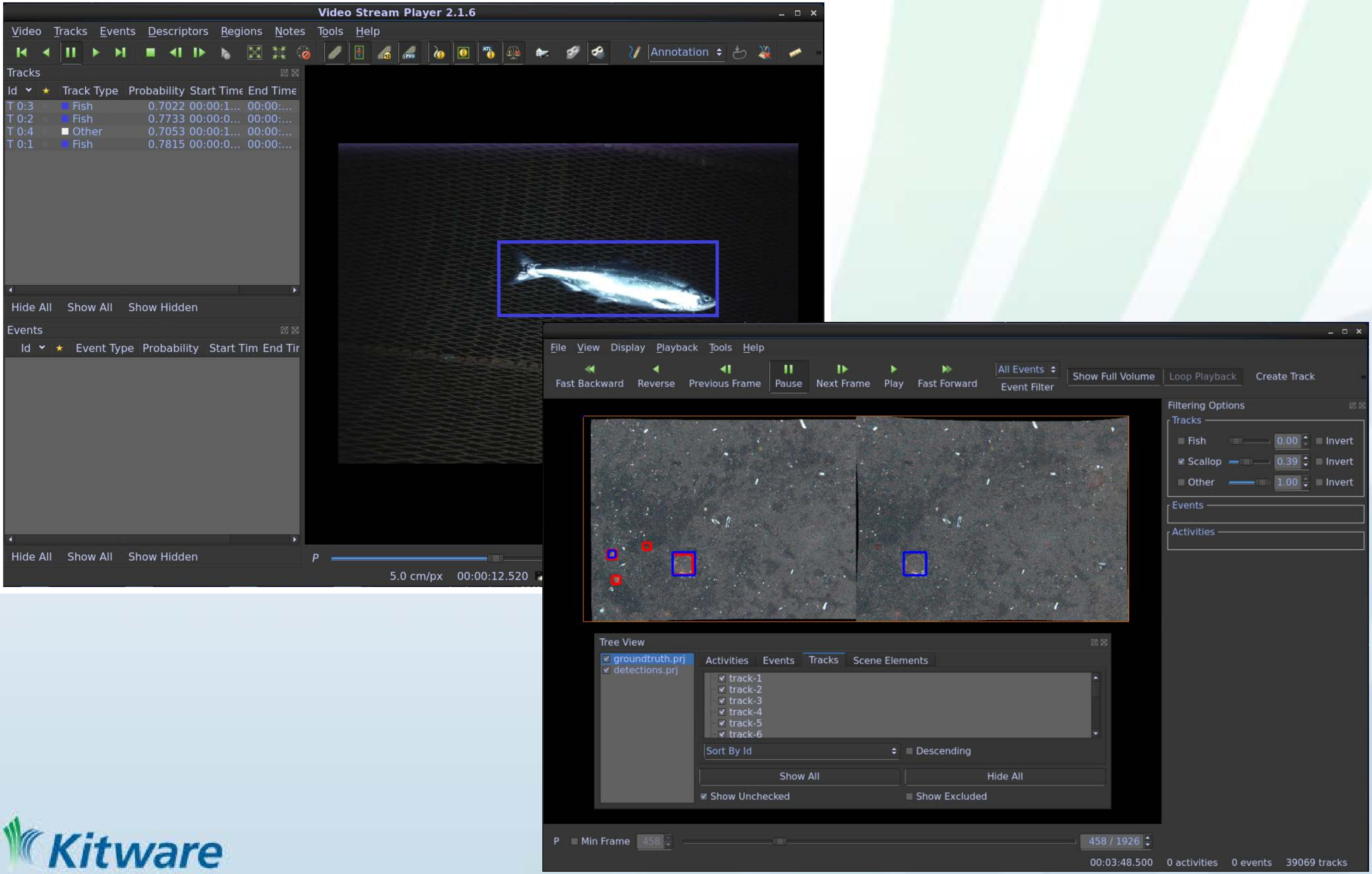
00:03:48.500 0 activities 0 events 39069 tracks

Visualization Tools



Evaluation and Quantification Tools

Goal: Object Detection



Id	★	Track Type	Probability	Start Time	End Time
T 0:3		Fish	0.7022	00:00:1...	00:00:...
T 0:2		Fish	0.7733	00:00:0...	00:00:...
T 0:4		Other	0.7053	00:00:1...	00:00:...
T 0:1		Fish	0.7815	00:00:0...	00:00:...

5.0 cm/px 00:00:12.520

Tracks

☐ Fish	0.00	☐ Invert
☑ Scallop	0.39	☐ Invert
☐ Other	1.00	☐ Invert

☑ groundtruth.prj
☑ detections.prj

☑ track-1
☑ track-2
☑ track-3
☑ track-4
☑ track-5
☑ track-6

Sort By Id ☐ Descending

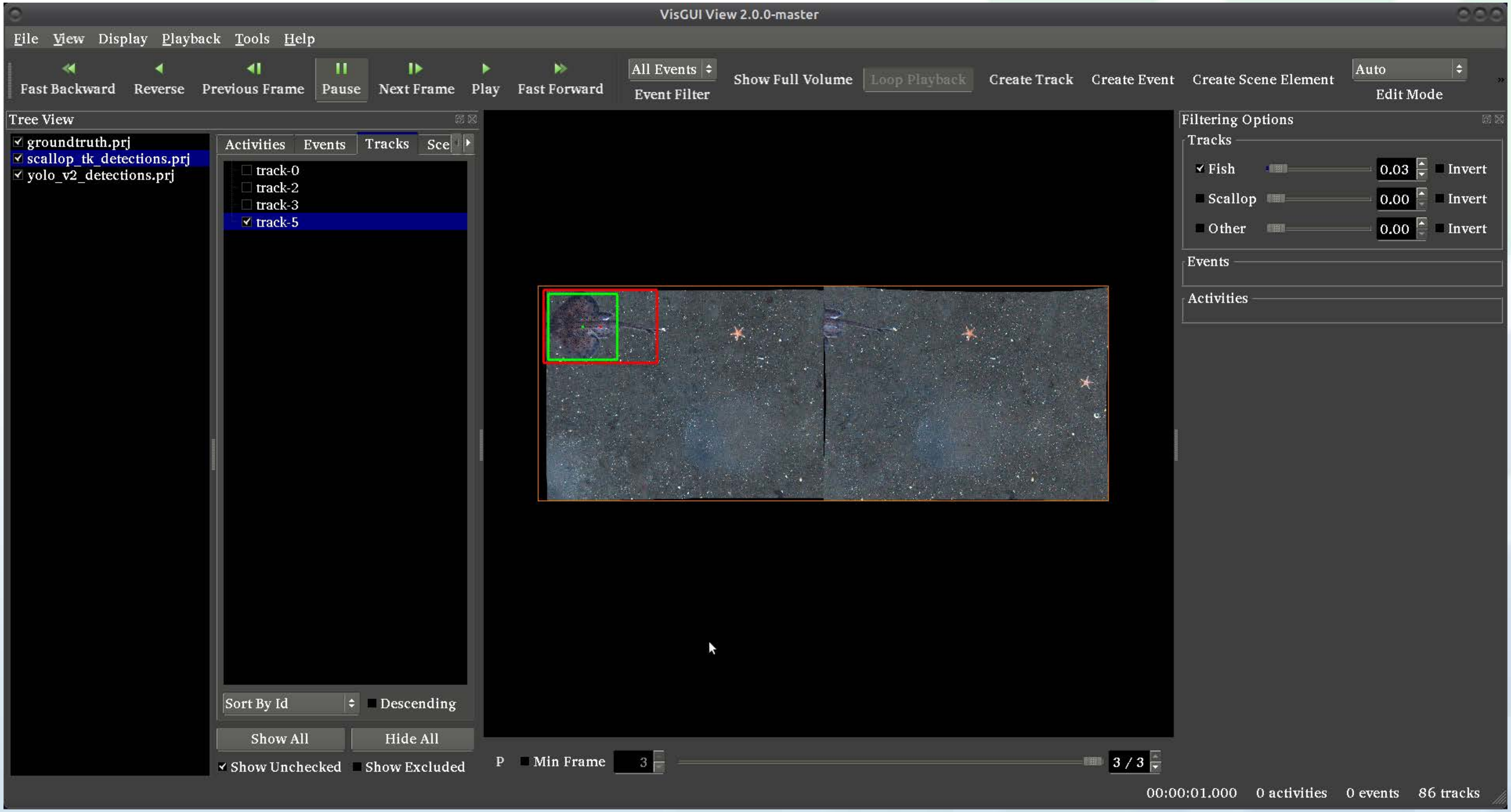
☑ Show Unchecked ☐ Show Excluded

☐ Min Frame 458

458 / 1926

00:03:48.500 0 activities 0 events 39069 tracks

Goal: Object Detection Viewing and Adjudication



☑ groundtruth.prj
☑ scallop_tk_detections.prj
☑ yolo_v2_detections.prj

□ track-0
□ track-2
□ track-3
☑ track-5

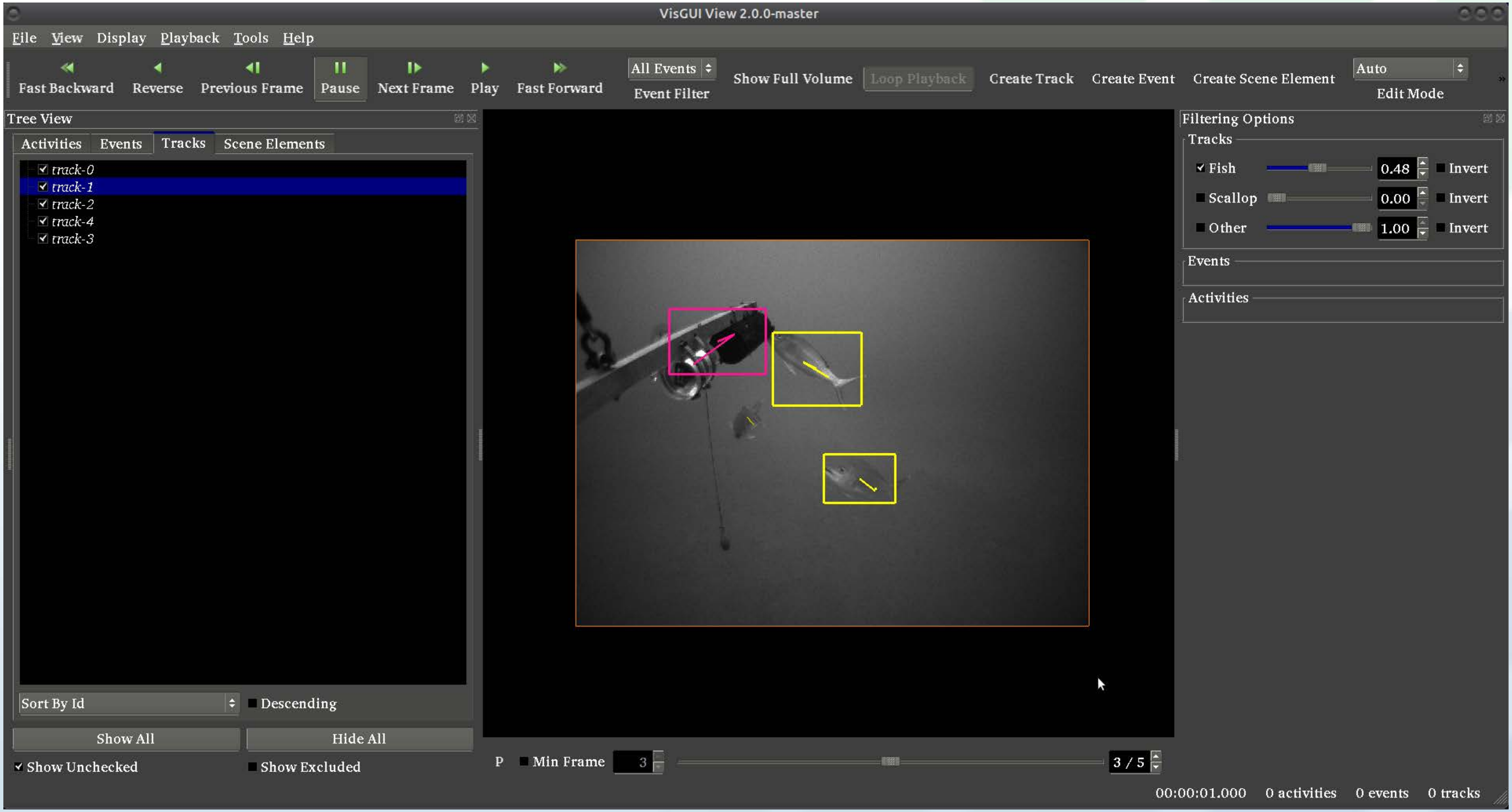
☑ Fish 0.03 ■ Invert
■ Scallop 0.00 ■ Invert
■ Other 0.00 ■ Invert

☑ Show Unchecked ■ Show Excluded

■ Min Frame 3 3 / 3

00:00:01.000 0 activities 0 events 86 tracks

Goal: Object Tracking



☑ track-0
☑ track-1
☑ track-2
☑ track-4
☑ track-3

□ Descending

☑ Show Unchecked □ Show Excluded

Tracks

☑ Fish	0.48	□ Invert
□ Scallop	0.00	□ Invert
□ Other	1.00	□ Invert

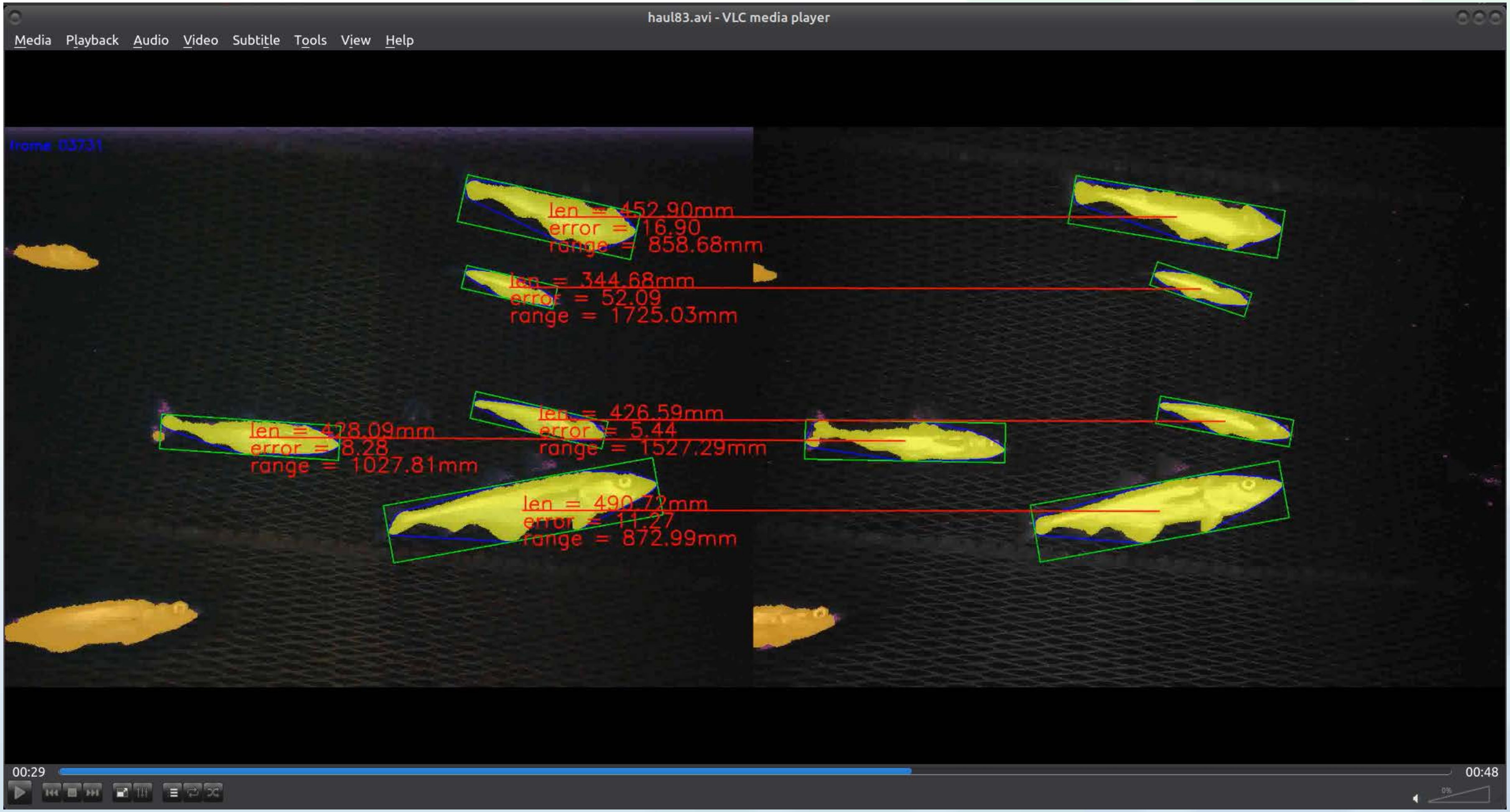
Events

Activities

□ Min Frame 3 3 / 5

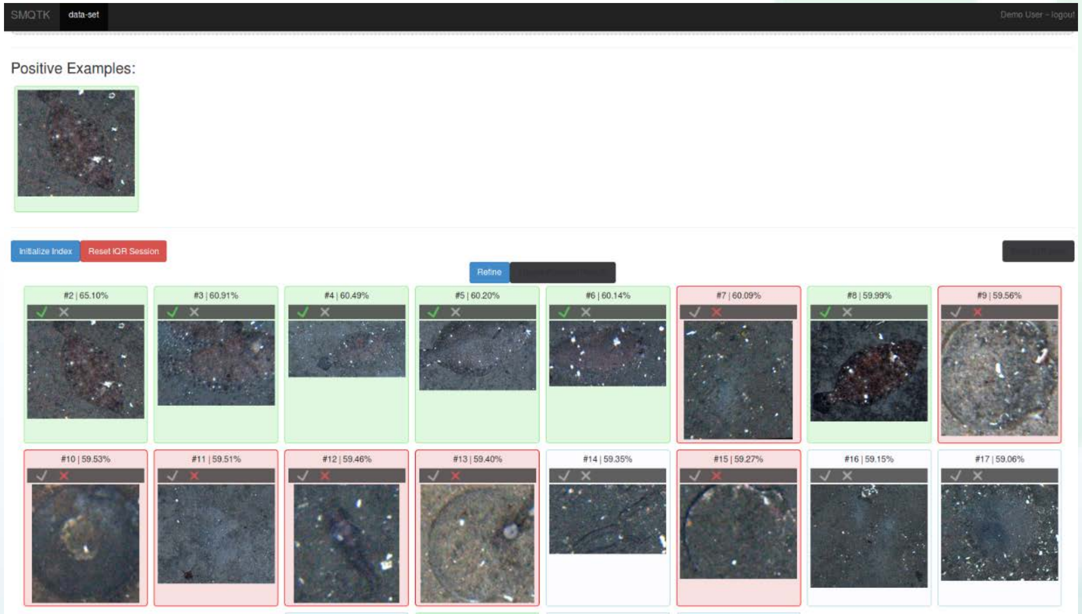
00:00:01.000 0 activities 0 events 0 tracks

Goal: Fish Measurement

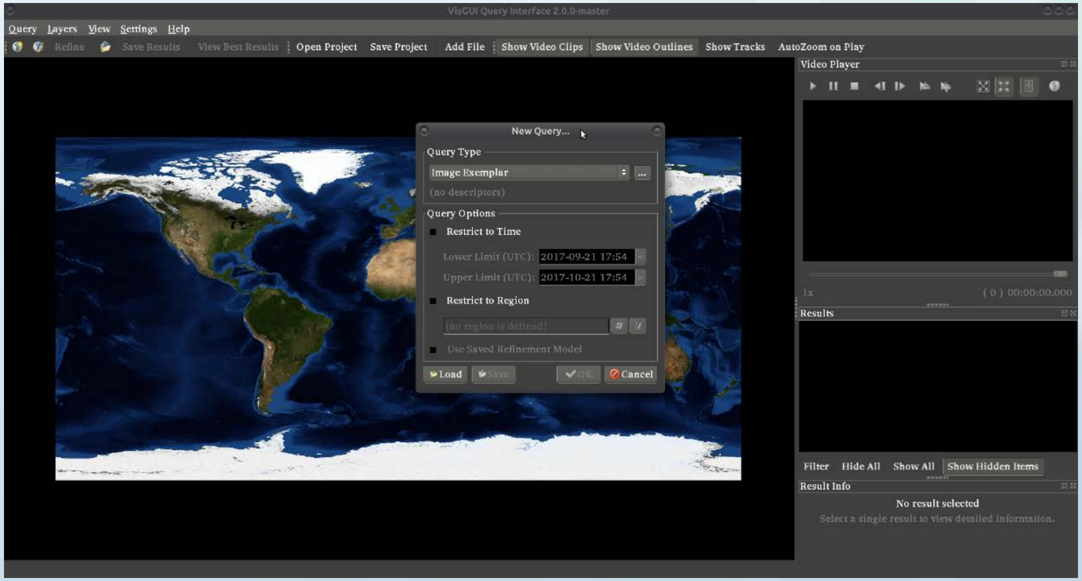

haul83.avi - VLC media player
Media Playback Audio Video Subtitle Tools View Help
00:29
00:48
0%

```
frame 03731
len = 452.90mm
error = 16.90
range = 858.68mm
len = 344.68mm
error = 52.09
range = 1725.03mm
len = 426.59mm
error = 5.44
range = 1527.29mm
len = 428.09mm
error = 8.28
range = 1027.81mm
len = 490.72mm
error = 11.27
range = 872.99mm
```

Goal: Image and Video Search



Positive Examples:



Query Type

Image Exemplar

(no descriptors)

Query Options

■ Restrict to Time

Lower Limit (UTC): 2017-09-21 17:54

Upper Limit (UTC): 2017-10-21 17:54

■ Restrict to Region

(no region is defined)

■ Use Saved Refinement Model

No result selected

Select a single result to view detailed information.

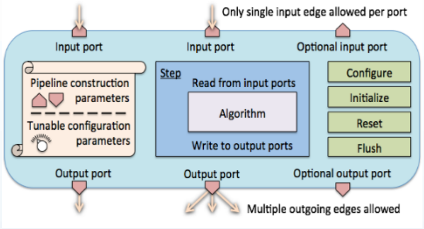
VIAME System Components

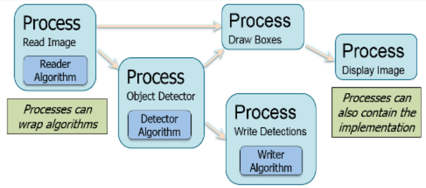
Fletch

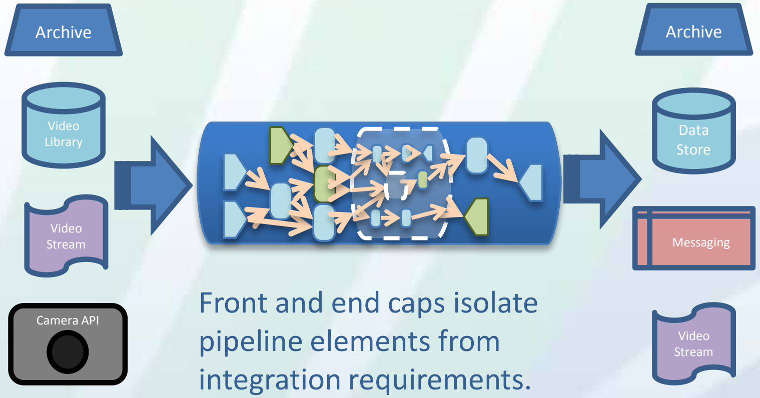
Builds common computer vision dependencies

Sprokit/KWIVER

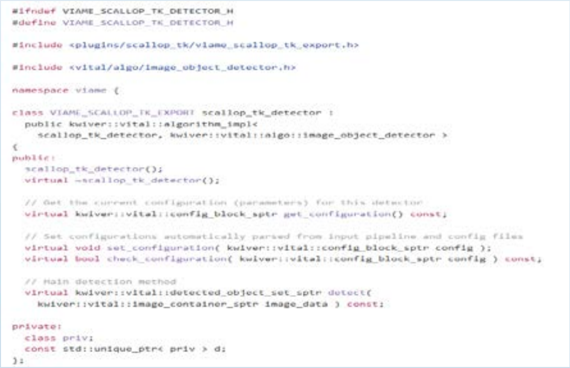
Connects up different algorithm nodes in runnable pipelines. Nodes can be implemented in C++, Python, or Matlab.







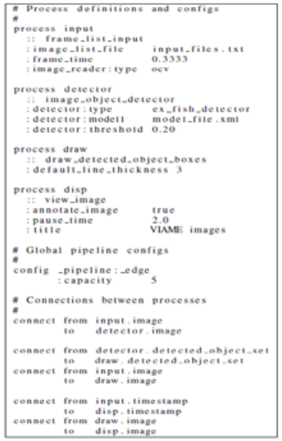
Front and end caps isolate pipeline elements from integration requirements.



```
#ifndef VIAME_SCALLOP_TK_DETECTOR_H
#define VIAME_SCALLOP_TK_DETECTOR_H

#include <plugins/scallop_tk/viame_scallop_tk_export.h>

#include <vital/algo/image_object_detector.h>

namespace viame {

class VIAME_SCALLOP_TK_EXPORT scallop_tk_detector :
  public kwiver::vital::algorithm_impl<
    scallop_tk_detector, kwiver::vital::algo::image_object_detector >
{
public:
  scallop_tk_detector();
  virtual ~scallop_tk_detector();

  // Get the current configuration (parameters) for this detector
  virtual kwiver::vital::config_block_sptr get_configuration() const;

  // Set configurations automatically parsed from input pipeline and config files
  virtual void set_configuration( kwiver::vital::config_block_sptr config );
  virtual bool check_configuration( kwiver::vital::config_block_sptr config ) const;

  // Main detection method
  virtual kwiver::vital::detected_object_set_sptr detect(
    kwiver::vital::image_container_sptr image_data ) const;

private:
  class priv;
  const std::unique_ptr< priv > d;
};
```

VIAME-Core

Contains domain-specific algorithms in “super-build” comprised of multiple projects and the above.

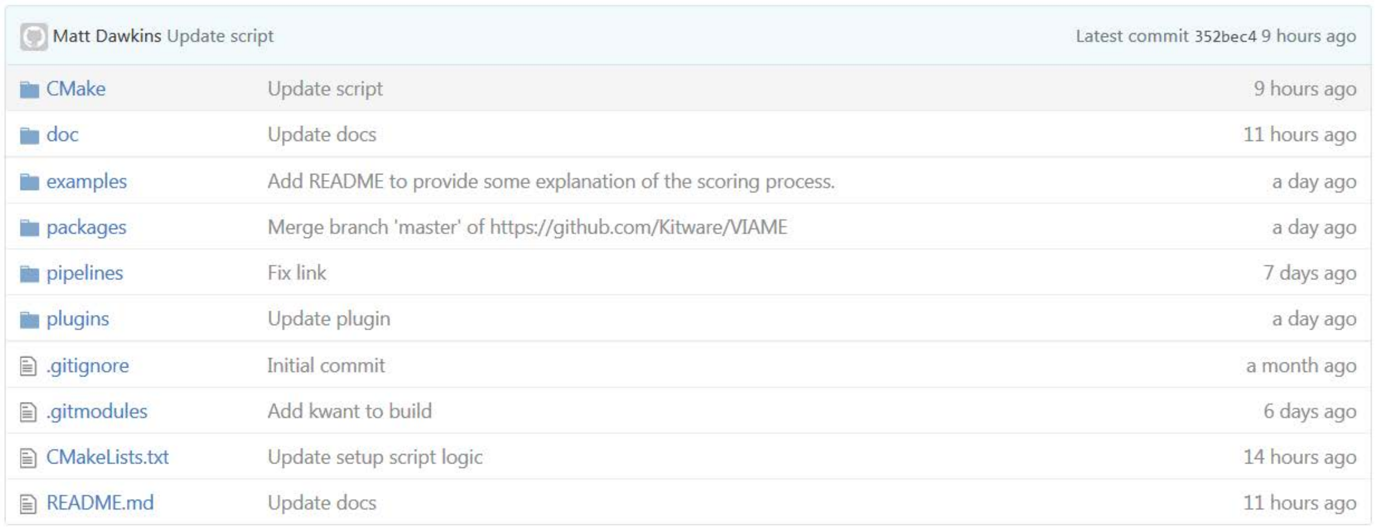


```
# Process definitions and configs
#
process input
  :: frame_list_input
  :image_list_file      input_files.txt
  :frame_time           0.3333
  :image_reader:type    ocv

process detector
  :: image_object_detector
  :detector:type        ex_fish_detector
  :detector:model1      model_file.xml
  :detector:threshold   0.20

process draw
  :: draw_detected_object_boxes
  :default_line_thickness 3

process disp
  :: view_image
  :annotate_image       true
  :pause_time           2.0
  :title                VIAME images

# Global pipeline configs
#
config _pipeline:_edge
       :capacity        5

# Connections between processes
#
connect from input.image
        to   detector.image

connect from detector.detected_object_set
        to   draw.detected_object_set
connect from input.image
        to   draw.image

connect from input.timestamp
        to   disp.timestamp
connect from draw.image
        to   disp.image
```

CMake Build System

Why?

- Cross-platform as long as code supports it
- Underlying pipeline code written primarily in C/C++
- Already used by several popular vision projects (OpenCV, VXL, Caffe)
- Free to use
- Kitware



Matt Dawkins Update script — Latest commit 352bec4 9 hours ago

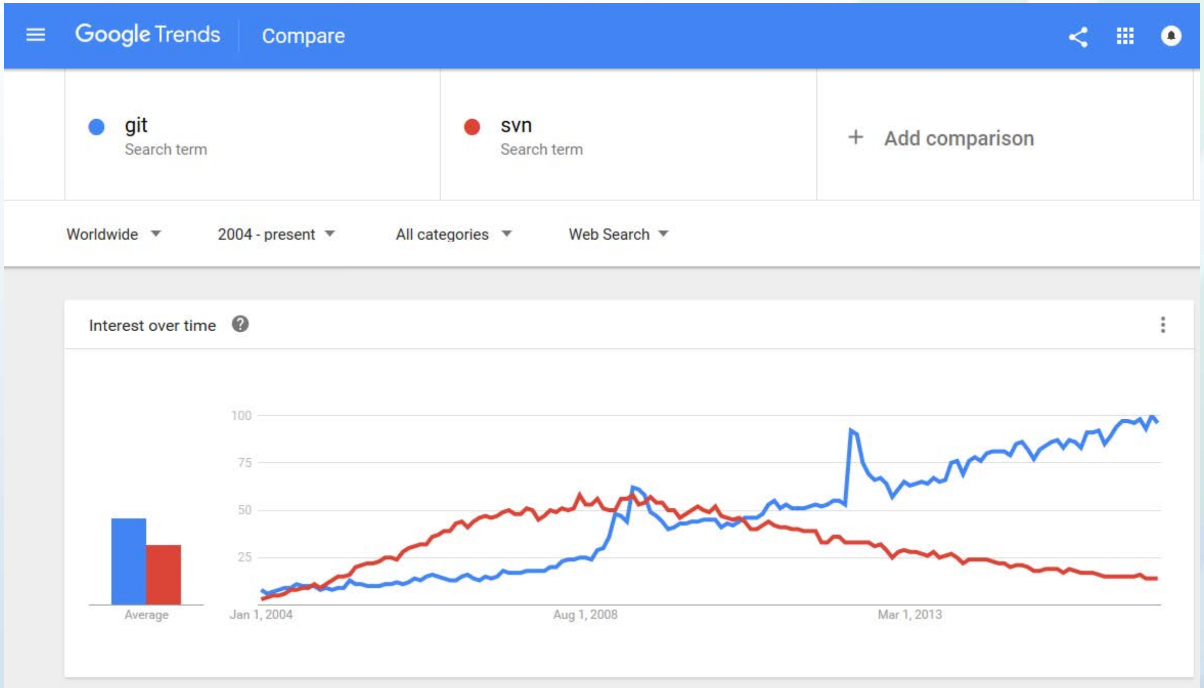
CMake	Update script	9 hours ago
doc	Update docs	11 hours ago
examples	Add README to provide some explanation of the scoring process.	a day ago
packages	Merge branch 'master' of https://github.com/Kitware/VIAME	a day ago
pipelines	Fix link	7 days ago
plugins	Update plugin	a day ago
.gitignore	Initial commit	a month ago
.gitmodules	Add kwant to build	6 days ago
CMakeLists.txt	Update setup script logic	14 hours ago
README.md	Update docs	11 hours ago

Fundamentals: https://cmake.org/runningcmake/

Git Version Control

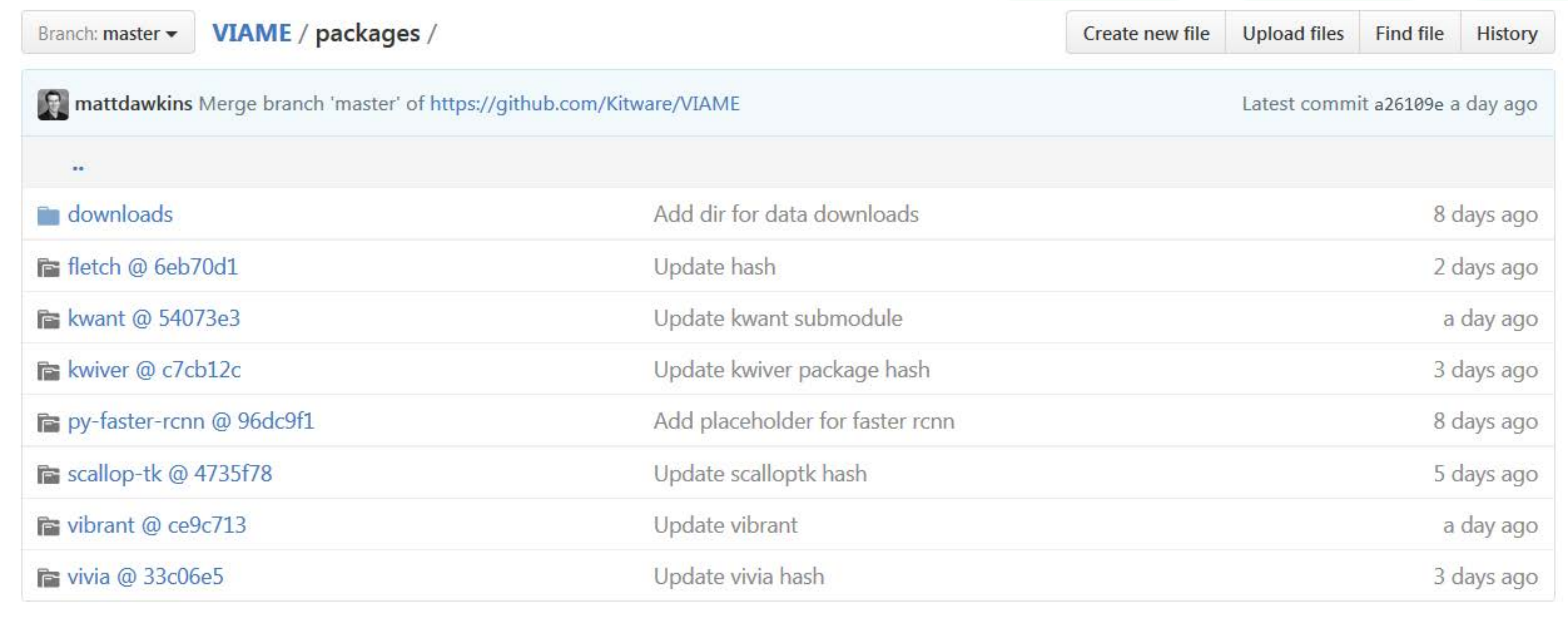
Alternatives: Use Nothing, Google Drive, CVS, SVN, Mercurial

Why Git? Like CMake, also used by many other vision toolkits. Slightly harder to learn than SVN, but more extensible, distributed.



Fundamentals: https://git-scm.com/book/en/v2/Getting-Started-Git-Basics

Super-Build Comprised of Multiple Projects



VIAME / packages /

mattdawkins Merge branch 'master' of https://github.com/Kitware/VIAME — Latest commit a26109e a day ago

..

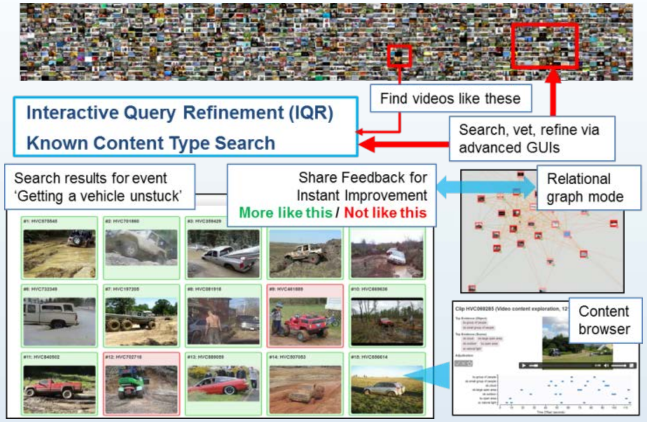
downloads	Add dir for data downloads	8 days ago
fletch @ 6eb70d1	Update hash	2 days ago
kwant @ 54073e3	Update kwant submodule	a day ago
kwiver @ c7cb12c	Update kwiver package hash	3 days ago
py-faster-rcnn @ 96dc9f1	Add placeholder for faster rcnn	8 days ago
scallop-tk @ 4735f78	Update scalloptk hash	5 days ago
vibrant @ ce9c713	Update vibrant	a day ago
vivia @ 33c06e5	Update vivia hash	3 days ago

KWIVER.org

Kitware Image and Video Exploitation and Retrieval Toolkit

An Open Source, production-quality video analytics toolkit

Social Multimedia Query ToolKit



Find videos like these

Interactive Query Refinement (IQR)

Known Content Type Search

Search, vet, refine via advanced GUIs

Search results for event 'Getting a vehicle unstuck'

Share Feedback for Instant Improvement

More like this / Not like this

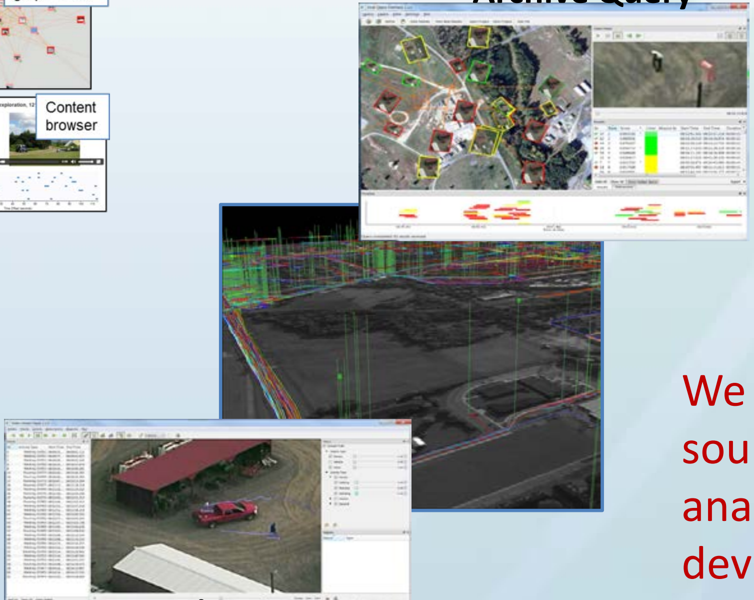
Relational graph mode

Content browser

***VIBRANT**:*

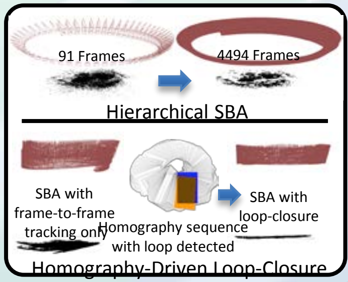
Video and Image-Based Retrieval and Analysis Toolkit

Archive Query



Streaming FMV

Motion-imagery Aerial Photogrammetry Toolkit



91 Frames
4494 Frames
Hierarchical SBA
SBA with frame-to-frame tracking only
Homography sequence with loop detected
SBA with loop-closure
Homography-Driven Loop-Closure

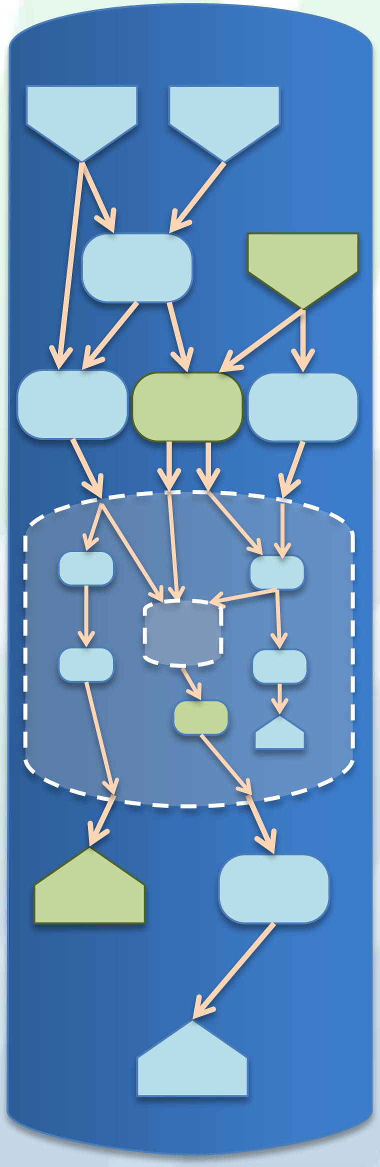
We hope to establish an open-source community for video analytics research and development

KWIVER.org

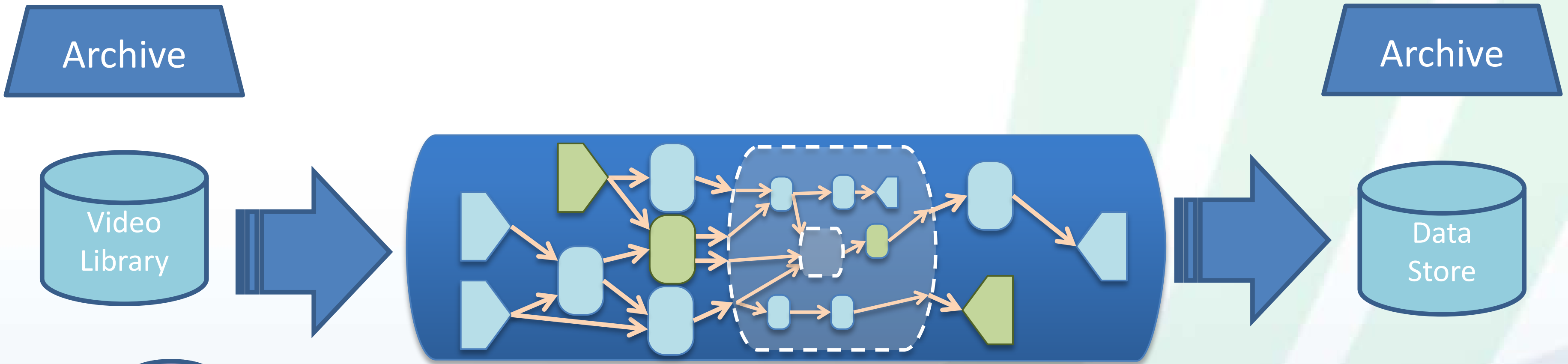
- Source code repositories maintained at GitHub
- Current toolkits available:
  - Motion-imagery Aerial Photogrammetry Toolkit for video stabilization and online bundle adjustment **MAP-TK**
  - Social Multimedia Query Toolkit for visual context extraction and querying for social multimedia **SMQTK**
  - Stream Processing Toolkit to facilitate multi-state, pipelined processing of data streams **Sprokit**
  - Common data structures and abstractions for computer vision and machine learning systems **VITAL**
  - GPU acceleration of core vision algorithms using OpenCL **VisCL**
  - GUIs and sophisticated visualization tools for content automatically extracted from video, based on VTK **ViVIA**
  - Detection and tracking of movers in video **VIBRANT**
  - CMake tools to set up complex build environments and dependencies **Fletch**

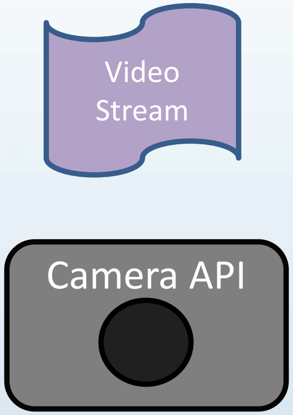
Sprokit – A Framework for Streaming Data Processing

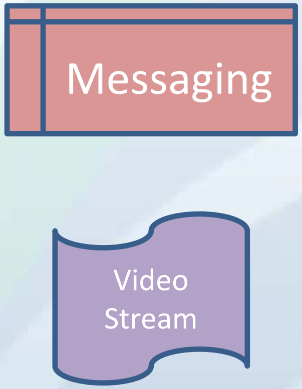
- What is Sprokit?
  - Plugin-based **streaming data processing engine** on which to build modular streaming data processing applications (especially video processing applications).
- What does Sprokit do?
  - Chains processing elements into a directed acyclic graph (DAG)
  - Executes a constructed pipeline on streaming data (e.g. video)
  - Provides dynamic construction/configuration via configuration files
  - Supports algorithms written in C++, Matlab, and Python
- Why was Sprokit developed?
  - To build complex streaming algorithms from simple components
  - To replace an older, much more restrictive, pipeline framework
  - Because existing open source frameworks (e.g. Gstreamer, Ecto, etc.) did not meet all requirements in the list above.



Swappable Front and End Caps

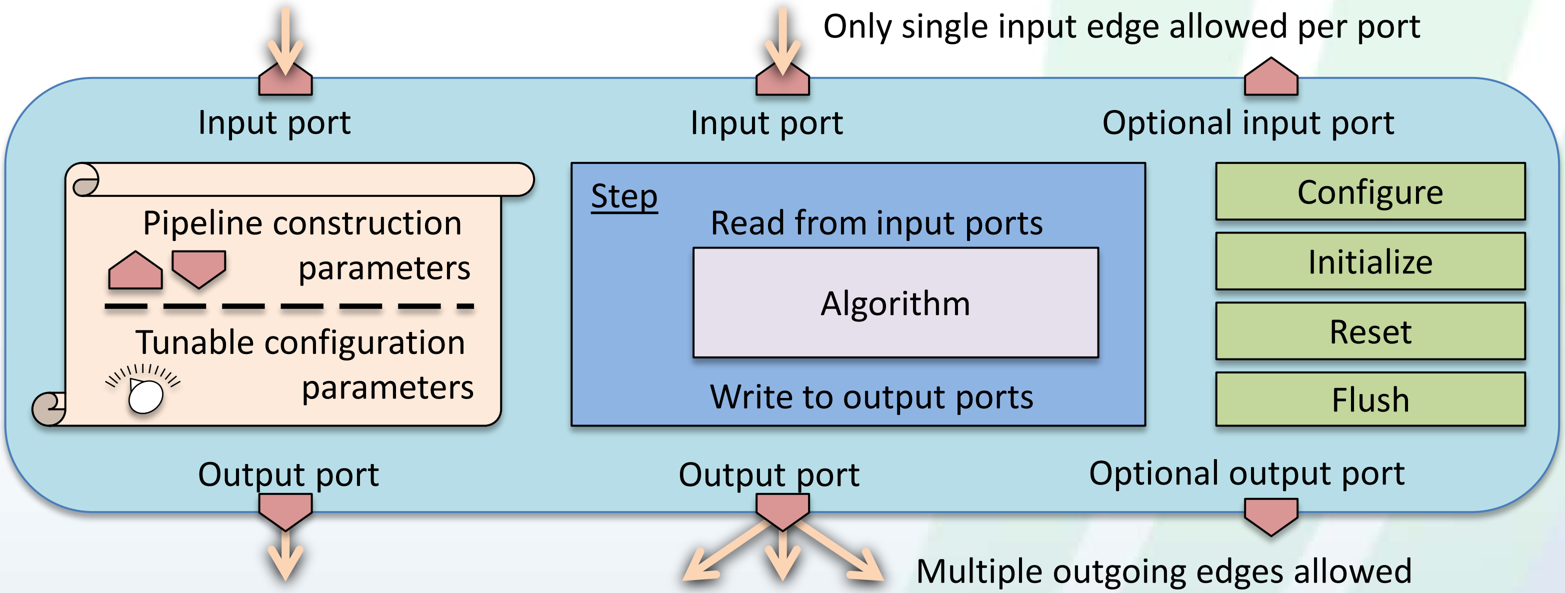






End caps and front caps isolate pipeline elements from integration requirements. Data from a variety of input sources is normalized, analytics are sent to a variety of destinations. Processing elements merely consume their inputs and produce their outputs.

Individual Processing Nodes



Only single input edge allowed per port

Input port

Input port

Optional input port

Pipeline construction parameters

Tunable configuration parameters

Step

Read from input ports

Algorithm

Write to output ports

Configure

Initialize

Reset

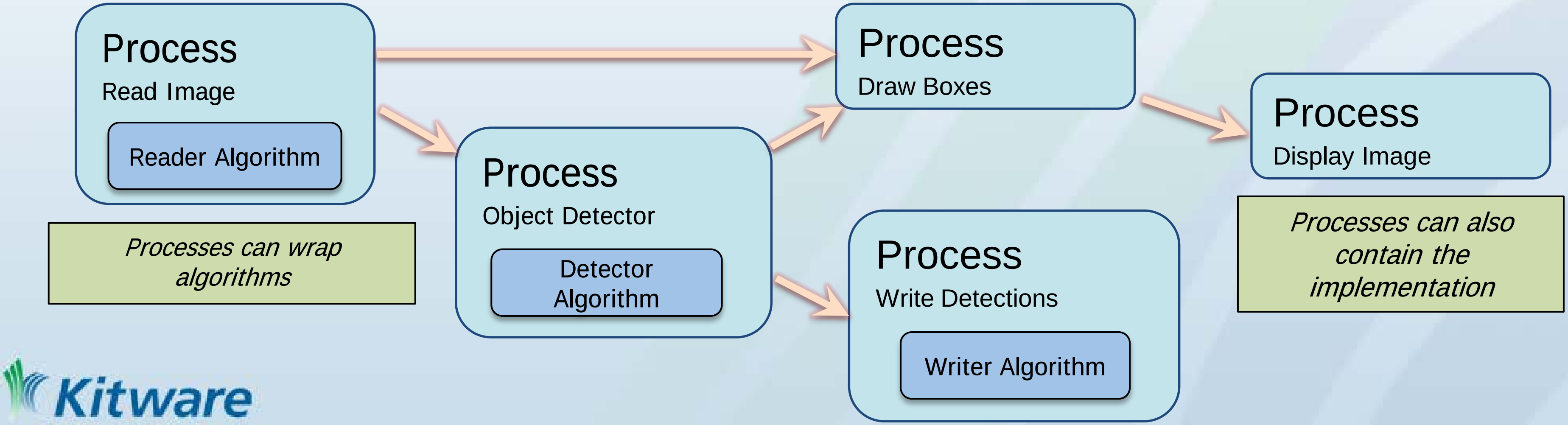
Flush

Output port

Output port

Optional output port

Multiple outgoing edges allowed



Process

Read Image

Reader Algorithm

Processes can wrap algorithms

Process

Object Detector

Detector Algorithm

Process

Draw Boxes

Process

Write Detections

Writer Algorithm

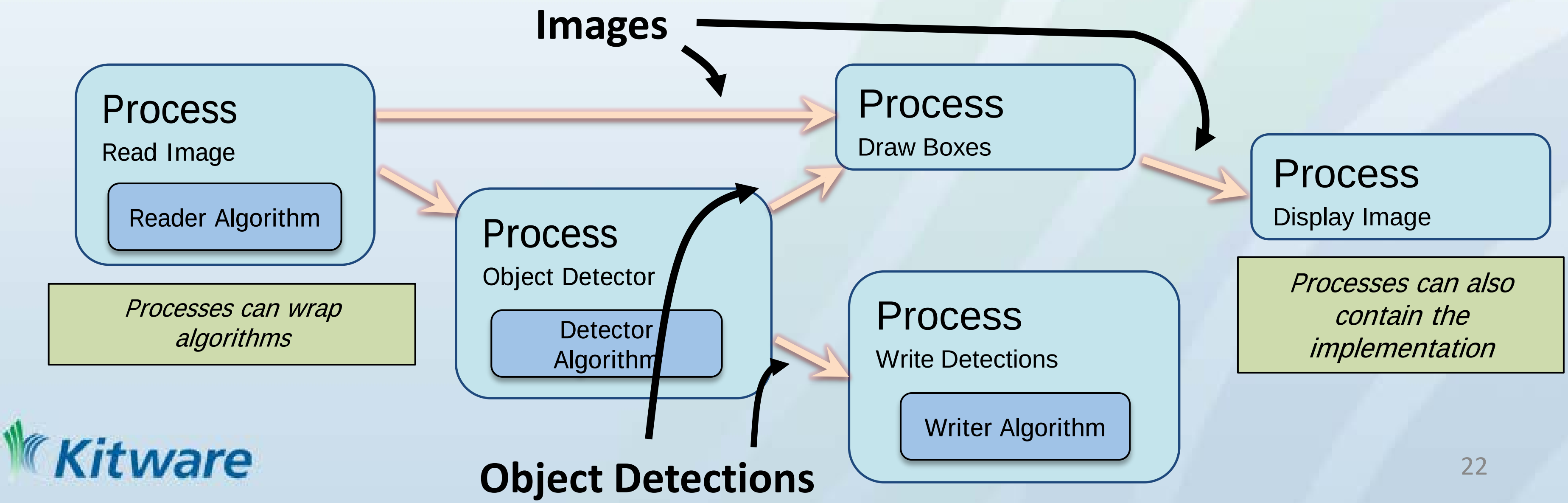
Process

Display Image

Processes can also contain the implementation

VITAL: Common Data Structures for Edges

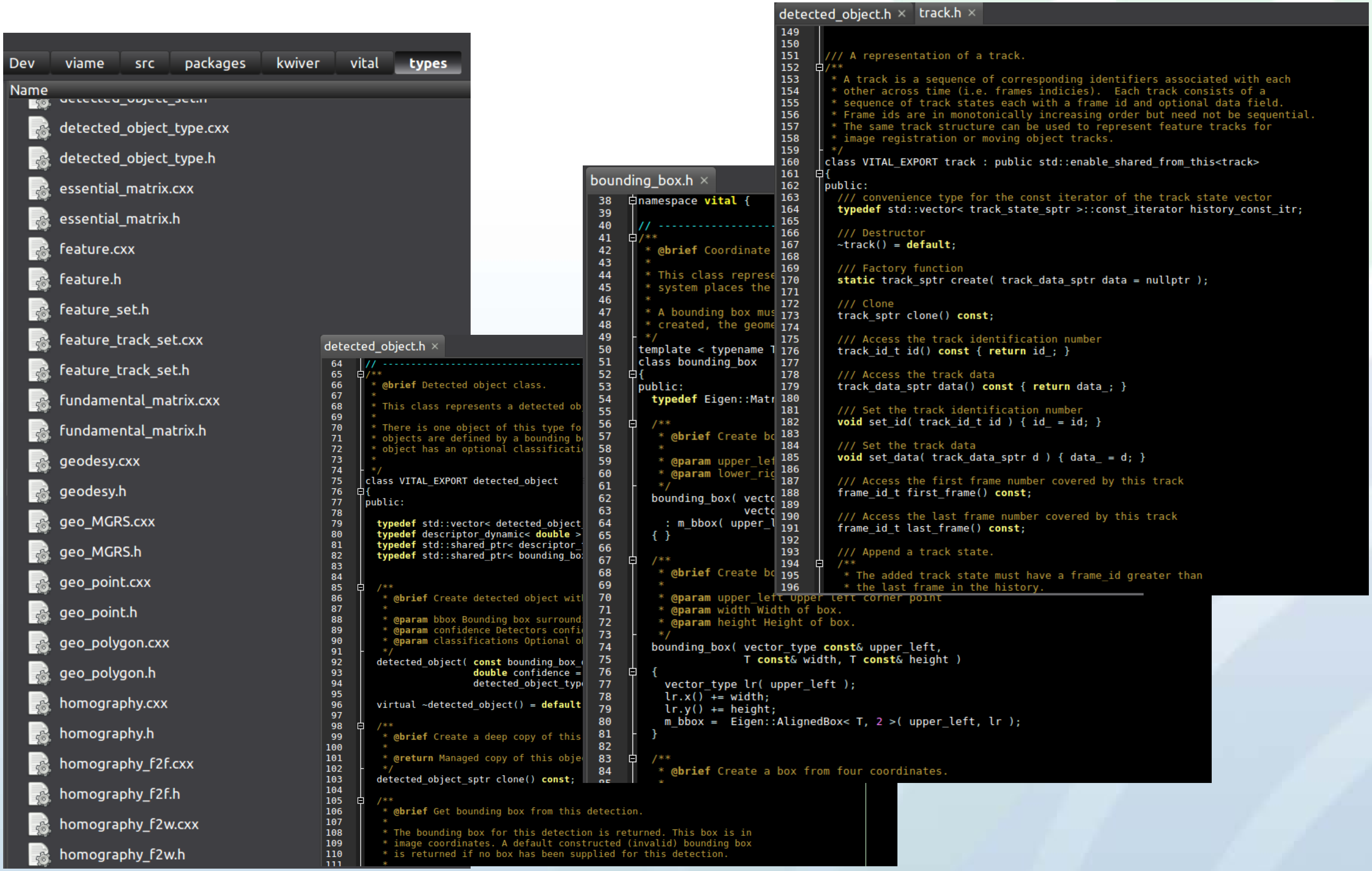
- Normalize and standardize data structures to facilitate integration across different algorithms
- Isolate system integration issues by providing test fixtures for remote development
- Encourage collaboration by providing a framework for data sharing and replay
- Encourage modular development with pipeline based architecture with common data structured passed on edges



Processes can wrap algorithms

Processes can also contain the implementation

VITAL: Common Example Types



detected_object_set.h
detected_object_type.cxx
detected_object_type.h
essential_matrix.cxx
essential_matrix.h
feature.cxx
feature.h
feature_set.h
feature_track_set.cxx
feature_track_set.h
fundamental_matrix.cxx
fundamental_matrix.h
geodesy.cxx
geodesy.h
geo_MGRS.cxx
geo_MGRS.h
geo_point.cxx
geo_point.h
geo_polygon.cxx
geo_polygon.h
homography.cxx
homography.h
homography_f2f.cxx
homography_f2f.h
homography_f2w.cxx
homography_f2w.h

```
// ----------------------------------
/**
 * @brief Detected object class.
 *
 * This class represents a detected ob
 *
 * There is one object of this type fo
 * objects are defined by a bounding b
 * object has an optional classificati
 *
 */
class VITAL_EXPORT detected_object
{
public:

  typedef std::vector< detected_object
  typedef descriptor_dynamic< double >
  typedef std::shared_ptr< descriptor_
  typedef std::shared_ptr< bounding_bo


  /**
   * @brief Create detected object with
   *
   * @param bbox Bounding box surround
   * @param confidence Detectors confi
   * @param classifications Optional o
   */
  detected_object( const bounding_box_
                   double confidence =
                   detected_object_typ

  virtual ~detected_object() = default

  /**
   * @brief Create a deep copy of this
   *
   * @return Managed copy of this obje
   */
  detected_object_sptr clone() const;

  /**
   * @brief Get bounding box from this detection.
   *
   * The bounding box for this detection is returned. This box is in
   * image coordinates. A default constructed (invalid) bounding box
   * is returned if no box has been supplied for this detection.
   *
```

```
namespace vital {

// ------------------
/**
 * @brief Coordinate
 *
 * This class represe
 * system places the
 *
 * A bounding box mus
 * created, the geome
 */
template < typename T
class bounding_box
{
public:
  typedef Eigen::Matr

  /**
   * @brief Create bo
   *
   * @param upper_lef
   * @param lower_rig
   */
  bounding_box( vecto
                vecto
    : m_bbox( upper_l
  { }

  /**
   * @brief Create bo
   *
   * @param upper_left Upper left corner point
   * @param width Width of box.
   * @param height Height of box.
   */
  bounding_box( vector_type const& upper_left,
                T const& width, T const& height )
  {
    vector_type lr( upper_left );
    lr.x() += width;
    lr.y() += height;
    m_bbox =  Eigen::AlignedBox< T, 2 >( upper_left, lr );
  }

  /**
   * @brief Create a box from four coordinates.
   *
```

```


/// A representation of a track.
/**
 * A track is a sequence of corresponding identifiers associated with each
 * other across time (i.e. frames indicies).  Each track consists of a
 * sequence of track states each with a frame id and optional data field.
 * Frame ids are in monotonically increasing order but need not be sequential.
 * The same track structure can be used to represent feature tracks for
 * image registration or moving object tracks.
 */
class VITAL_EXPORT track : public std::enable_shared_from_this<track>
{
public:
  /// convenience type for the const iterator of the track state vector
  typedef std::vector< track_state_sptr >::const_iterator history_const_itr;

  /// Destructor
  ~track() = default;

  /// Factory function
  static track_sptr create( track_data_sptr data = nullptr );

  /// Clone
  track_sptr clone() const;

  /// Access the track identification number
  track_id_t id() const { return id_; }

  /// Access the track data
  track_data_sptr data() const { return data_; }

  /// Set the track identification number
  void set_id( track_id_t id ) { id_ = id; }

  /// Set the track data
  void set_data( track_data_sptr d ) { data_ = d; }

  /// Access the first frame number covered by this track
  frame_id_t first_frame() const;

  /// Access the last frame number covered by this track
  frame_id_t last_frame() const;

  /// Append a track state.
  /**
   * The added track state must have a frame_id greater than
   * the last frame in the history.
```

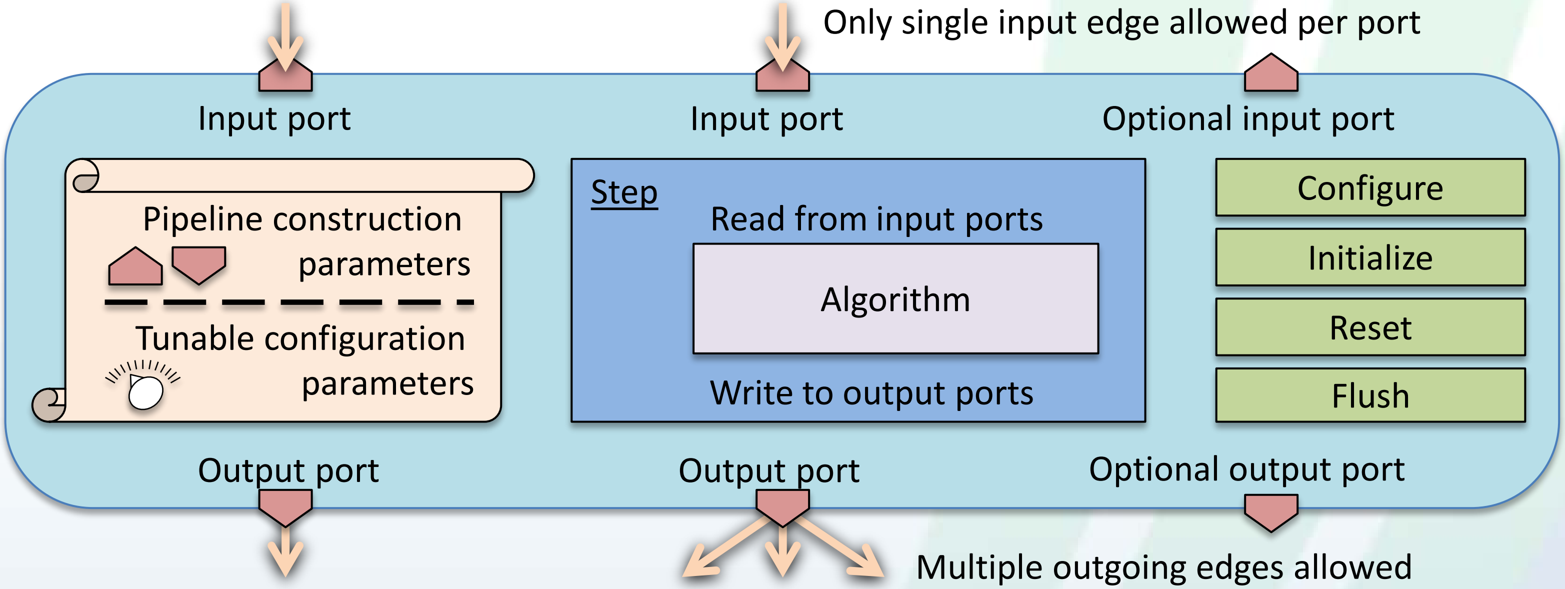
C++ but with python (and limited Matlab) bindings

Simple Pipeline Example

```
# ========================================
process input
  :: frame_list_input
  :image_list_file     input_files.txt
  :frame_time          .3333
  :image_reader:type   ocv

# ========================================
process detector
  :: image_object_detector
  :type                                   scallop_tk_detector
  :scallop_tk_detector:config_file   config_location

# ========================================
process draw
  :: draw_detected_object_boxes
  :default_line_thickness 3

# ========================================
process disp
  :: view_image
  :annotate_image          true
  :pause_time              0  # 1.0
  :title                   NOAA images
```

```
# ========================================
# global pipeline config
#
config _pipeline:_edge
       :capacity 10

# ========================================
# connections
connect from input.image
       to   detector.image

connect from detector.detected_object_set
       to   draw.detected_object_set
connect from input.image
       to draw.image

connect from detector.detected_object_set
       to archive.detected_object_set

connect from input.timestamp
       to   disp.timestamp
connect from draw.image
       to   disp.image
```

Process Definition

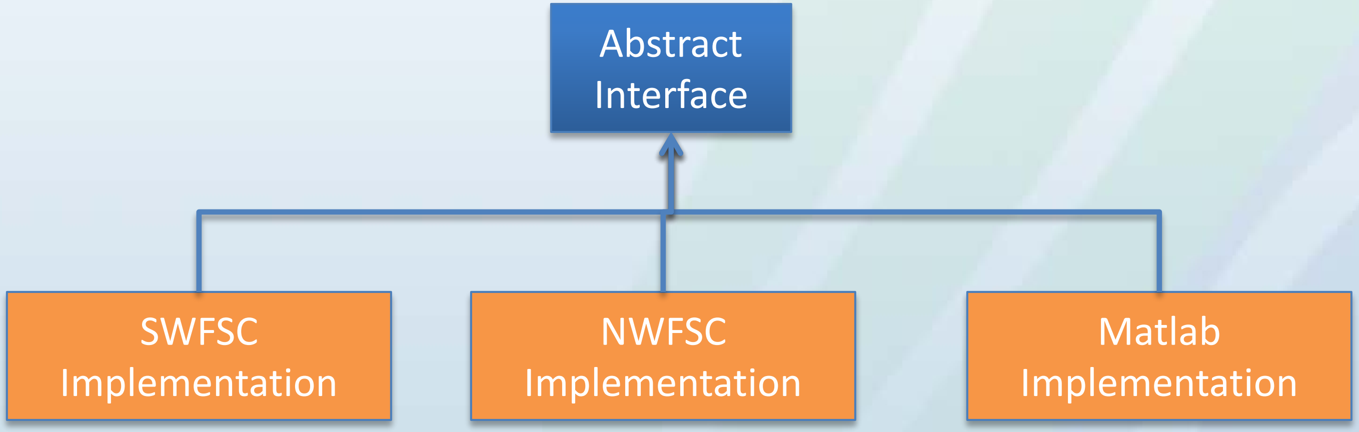


Two separate ways to define:

- Manual: User specifies all input/output ports and each rectangular block in the above (step, configure, initialize, etc…)
- Automatic: Only need to define single function for existing base class APIs with process wrappings
  - For example, object_detector: image in, detections out

Algorithm Concepts

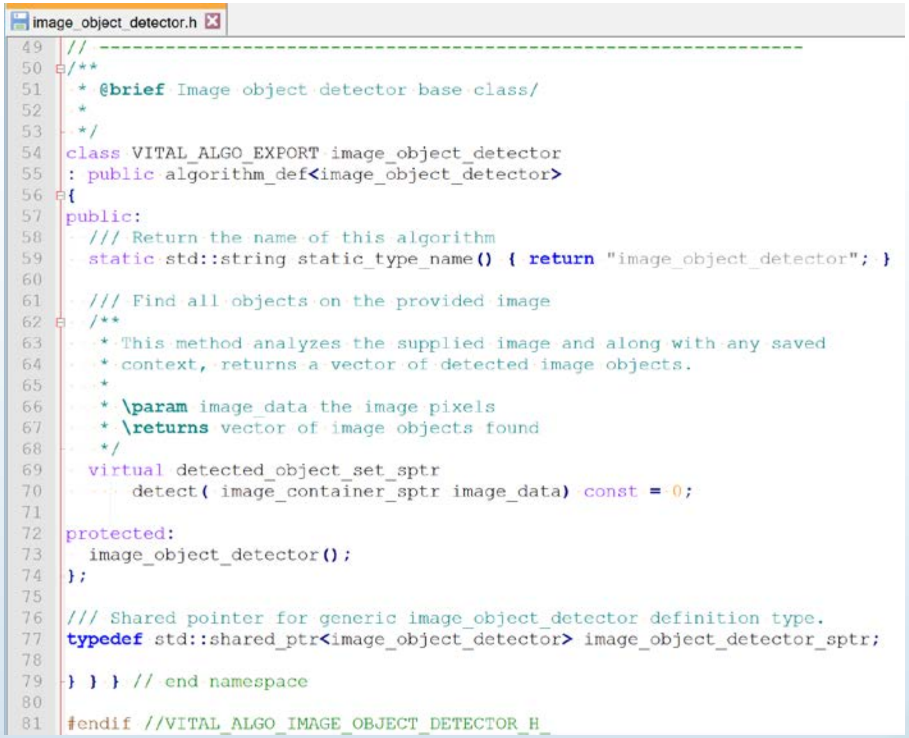
- Application uses abstract algorithm type using a polymorphic model
- Instantiates an implementation based on config
- Implementations are dynamically loadable
- New implementations can be easily added



Automatic Implementation Method:

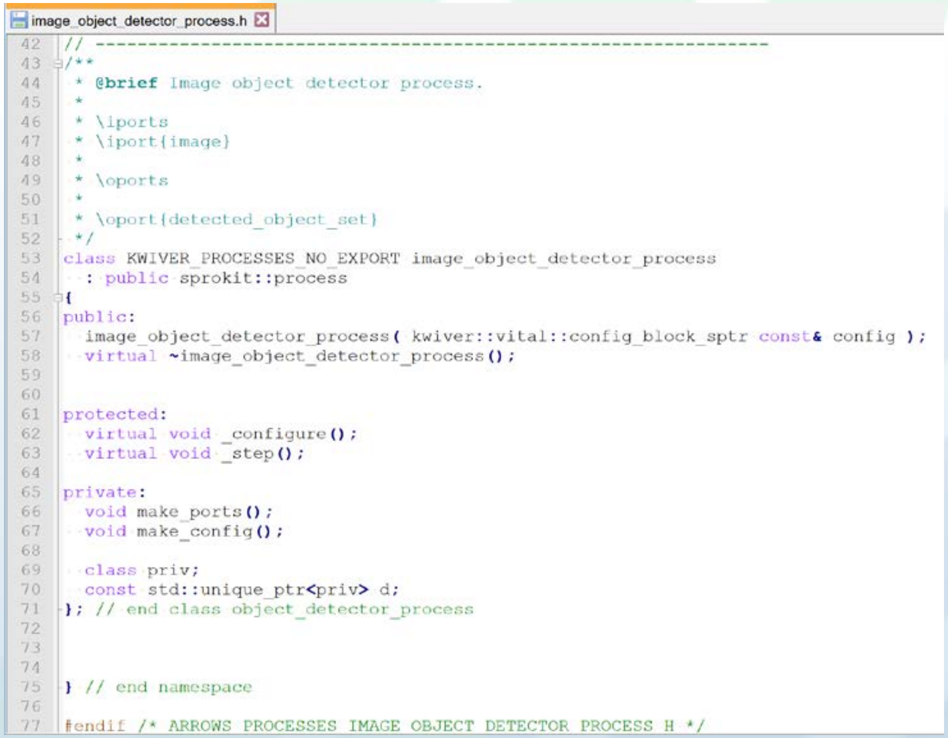
Example Base Classes

kwiver/vital/algos/image_object_detection.h



```
// -----------------------------------------------------------------
/**
 * @brief Image object detector base class/
 *
 */
class VITAL_ALGO_EXPORT image_object_detector
: public algorithm_def<image_object_detector>
{
public:
  /// Return the name of this algorithm
  static std::string static_type_name() { return "image_object_detector"; }

  /// Find all objects on the provided image
  /**
   * This method analyzes the supplied image and along with any saved
   * context, returns a vector of detected image objects.
   *
   * \param image_data the image pixels
   * \returns vector of image objects found
   */
  virtual detected_object_set_sptr
      detect( image_container_sptr image_data) const = 0;

protected:
  image_object_detector();
};

/// Shared pointer for generic image_object_detector definition type.
typedef std::shared_ptr<image_object_detector> image_object_detector_sptr;

} } } // end namespace

#endif //VITAL_ALGO_IMAGE_OBJECT_DETECTOR_H_
```

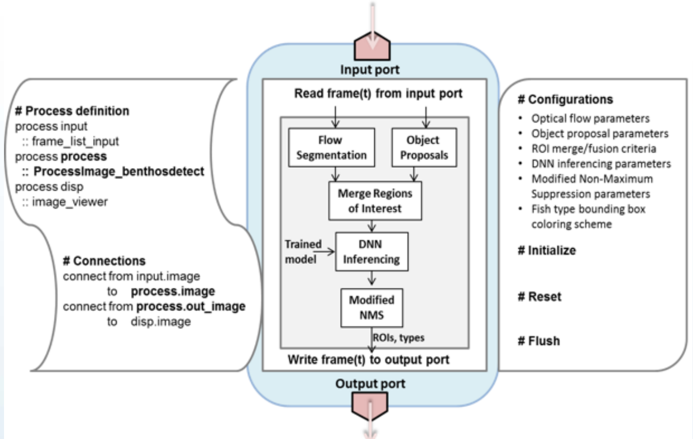
kwiver/sprokit/core/image_object_detector_process.h



```
// ----------------------------------------------------------------
/**
 * @brief Image object detector process.
 *
 * \iports
 * \iport{image}
 *
 * \oports
 *
 * \oport{detected_object_set}
 */
class KWIVER_PROCESSES_NO_EXPORT image_object_detector_process
  : public sprokit::process
{
public:
  image_object_detector_process( kwiver::vital::config_block_sptr const& config );
  virtual ~image_object_detector_process();


protected:
  virtual void _configure();
  virtual void _step();

private:
  void make_ports();
  void make_config();

  class priv;
  const std::unique_ptr<priv> d;
}; // end class object_detector_process



} // end namespace

#endif /* ARROWS_PROCESSES_IMAGE_OBJECT_DETECTOR_PROCESS_H */
```

Example Algorithms

SRI: BenthosDetect



Process definition
process input
 :: frame_list_input
process **process**
 :: **ProcessImage_benthosdetect**
process disp
 :: image_viewer

Connections
connect from input.image
 to **process.image**
connect from **process.out_image**
 to disp.image

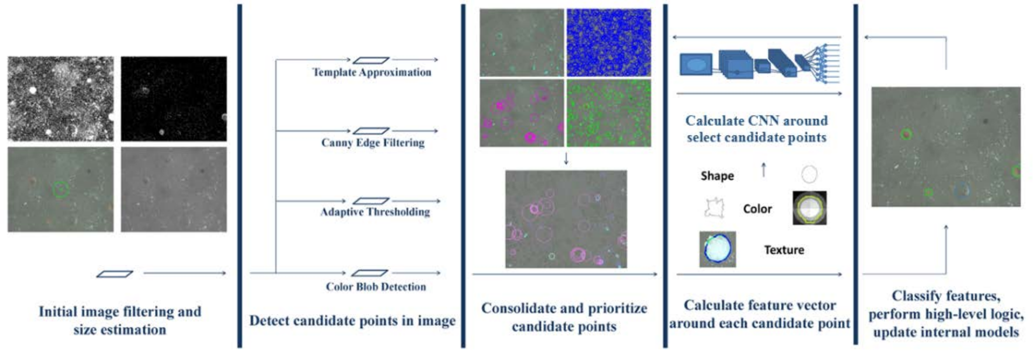
Configurations
- Optical flow parameters
- Object proposal parameters
- ROI merge/fusion criteria
- DNN inferencing parameters
- Modified Non-Maximum Suppression parameters
- Fish type bounding box coloring scheme

Initialize

Reset

Flush

Kitware: ScallopTK



Initial image filtering and size estimation

Detect candidate points in image

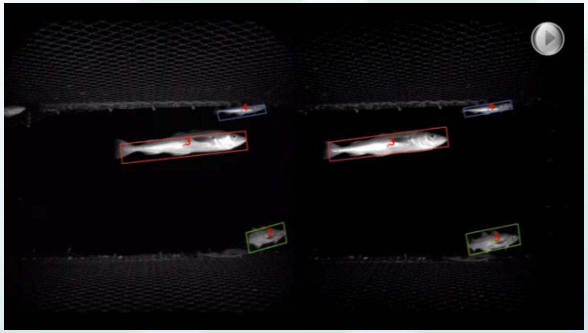
Consolidate and prioritize candidate points

Calculate CNN around select candidate points

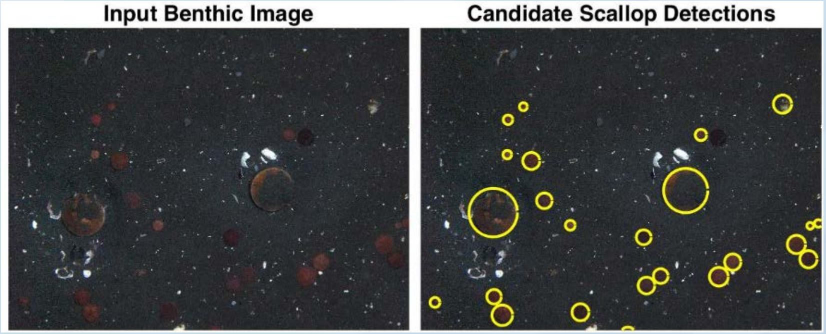
Calculate feature vector around each candidate point

Classify features, perform high-level logic, update internal models

NOAA/UW: FishRuler

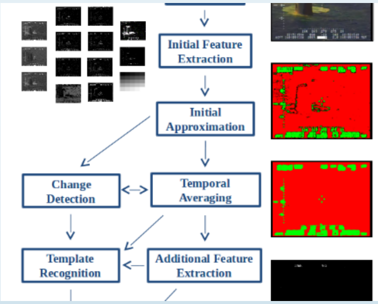


LANL: ScallopFinder

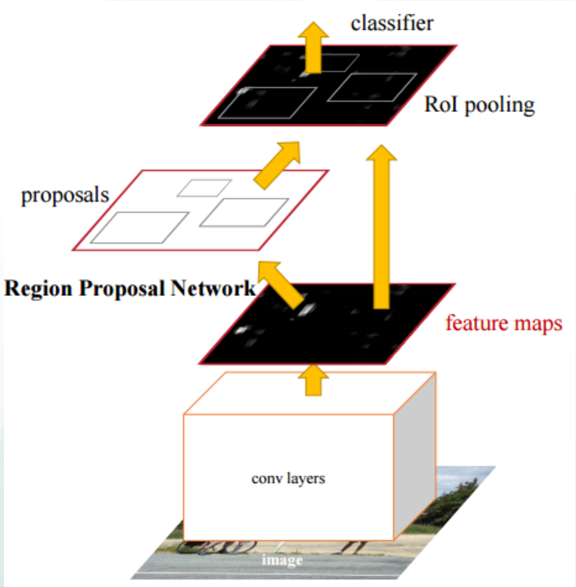


Example Algorithms (cont.)

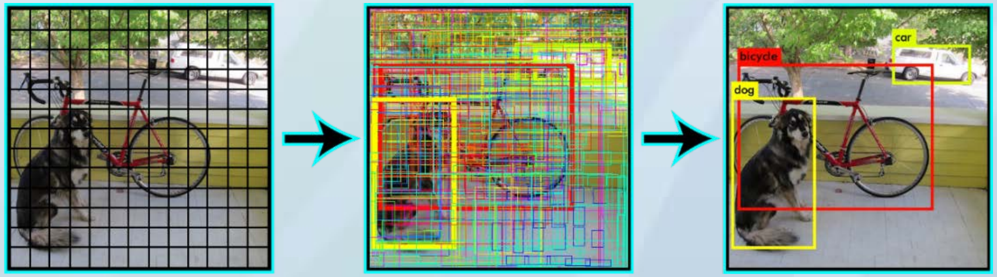
Burn-Out



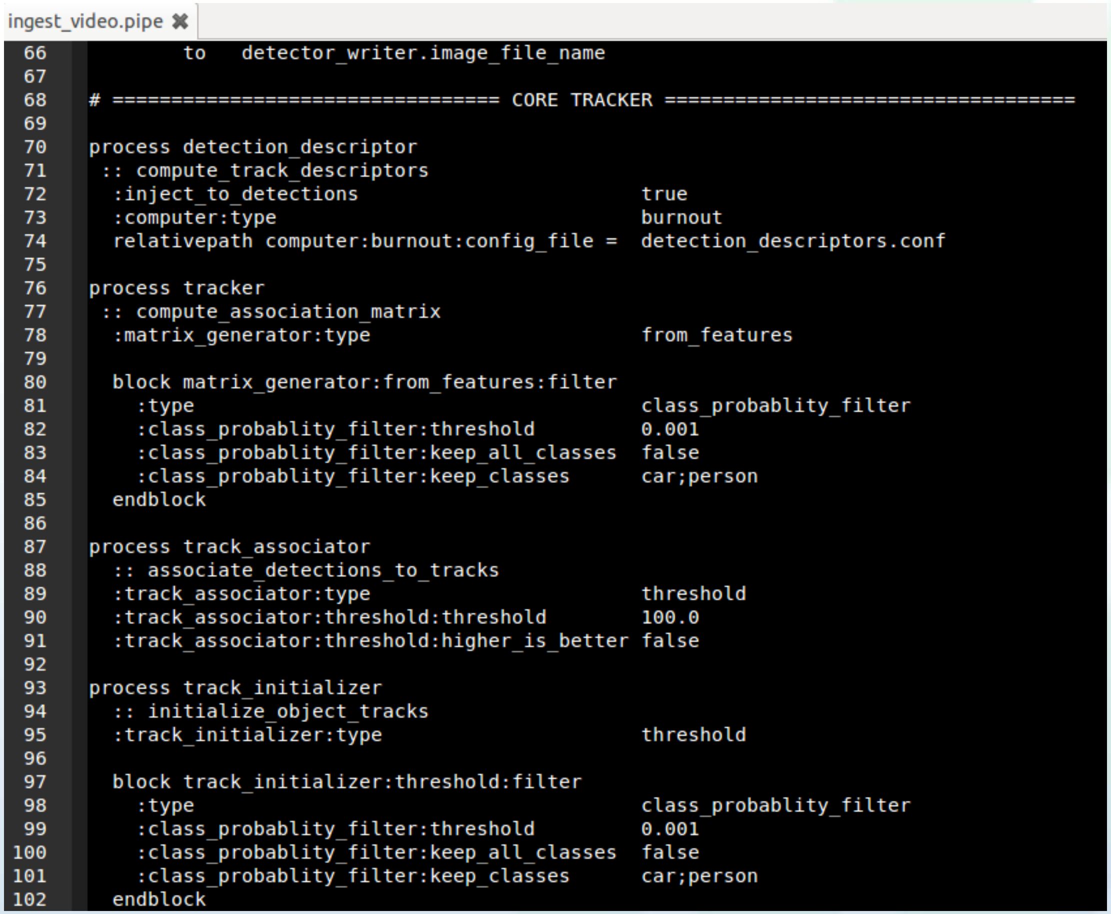
Faster R-CNN



YOLOv2



Baseline Tracker – Simple Tracker

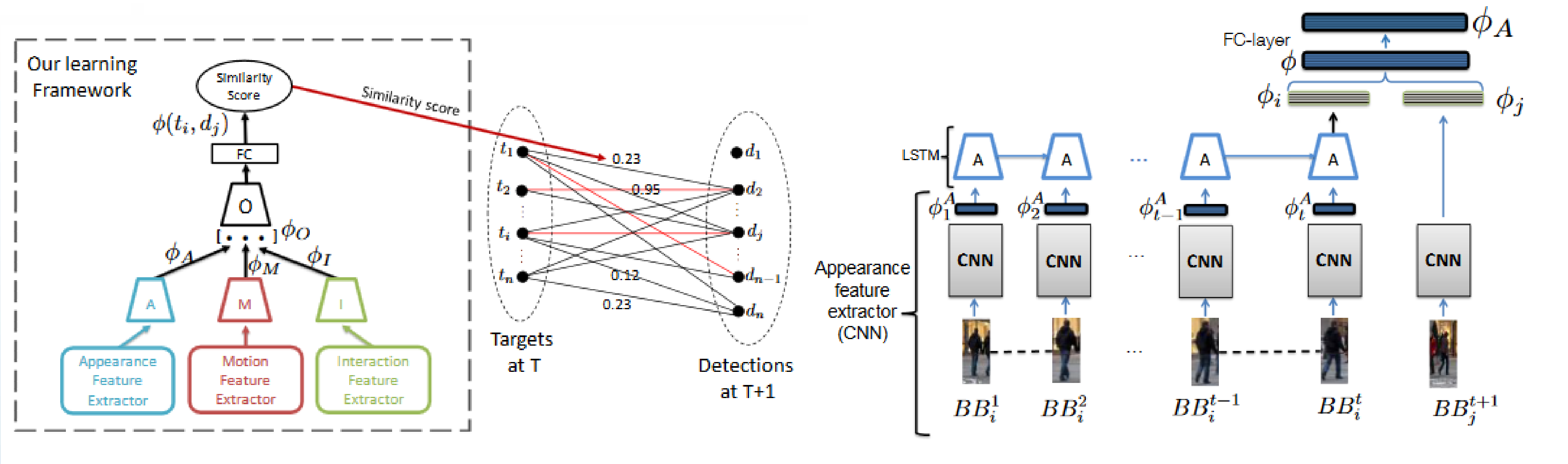


```
        to   detector_writer.image_file_name

# ================================= CORE TRACKER ===================================

process detection_descriptor
 :: compute_track_descriptors
  :inject_to_detections                        true
  :computer:type                               burnout
  relativepath computer:burnout:config_file =  detection_descriptors.conf

process tracker
 :: compute_association_matrix
  :matrix_generator:type                       from_features

  block matrix_generator:from_features:filter
    :type                                      class_probablity_filter
    :class_probablity_filter:threshold         0.001
    :class_probablity_filter:keep_all_classes  false
    :class_probablity_filter:keep_classes      car;person
  endblock

process track_associator
  :: associate_detections_to_tracks
  :track_associator:type                       threshold
  :track_associator:threshold:threshold        100.0
  :track_associator:threshold:higher_is_better false

process track_initializer
  :: initialize_object_tracks
  :track_initializer:type                      threshold

  block track_initializer:threshold:filter
    :type                                      class_probablity_filter
    :class_probablity_filter:threshold         0.001
    :class_probablity_filter:keep_all_classes  false
    :class_probablity_filter:keep_classes      car;person
  endblock
```

Pipeline performs differencing on CNN intermediate features derived from bounding boxes around detections

Baseline Tracker – Tracking the Untrackable

Implemented in PyTorch, Integrated into VIAME



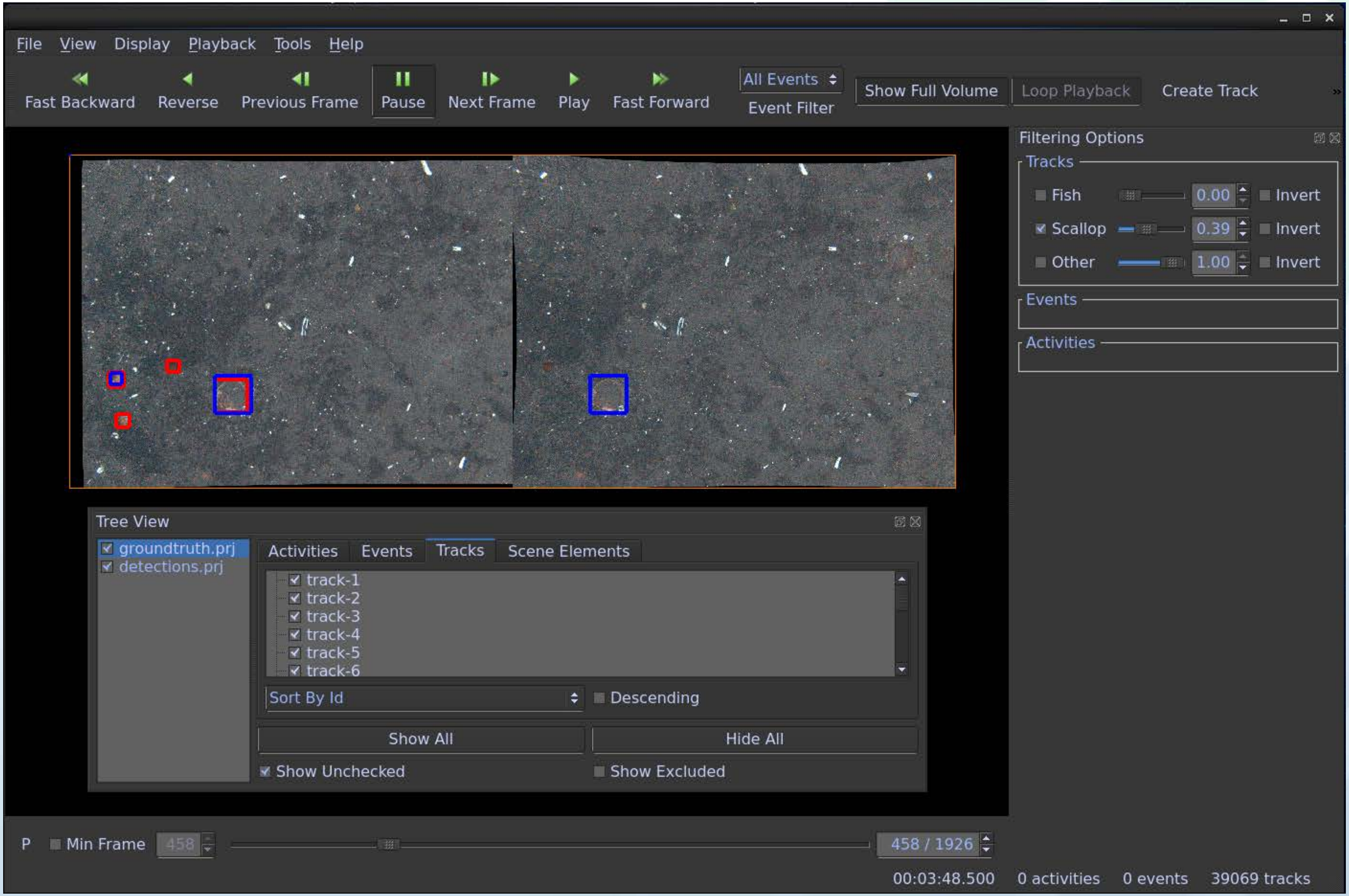
“Tracking the Untrackable: Learning to track multiple cues with long-term dependencies” Sadeghian et al. ICCV 2017

Algorithm Usage

- Application uses abstract algorithm(s) via pointer to base class, can either be used in pipeline files or C++ code. Pipelines can also be embedded in C++ code.
- Configuration info specifies algorithm to use
- High level code is unchanged as different algorithm implementations are used

```
namespace algo = kwiver::vital::algo;
algo::image_io_sptr    m_image_reader;
kwiver::vital::config_block_sptr algo_config = get_config(); // config

// validate config parameters
if ( ! algo::image_io::check_nested_algo_configuration( "image_reader", algo_config ) )
{  // Handle error
}

// instantiate image reader and converter based on config type
algo::image_io::set_nested_algo_configuration( "image_reader", algo_config, m_image_reader);
if ( ! m_image_reader )
{  // Handle error
}

// Read an image
kwiver::vital::image_container_sptr img_c;
img_c = m_image_reader->load( resolved_file_name );
```

Standalone Utilities: GUIs for Imagery



All Events

Filtering Options

Tracks

- □ Fish 0.00 □ Invert
- ☑ Scallop 0.39 □ Invert
- □ Other 1.00 □ Invert

Events

Activities

Tree View

- ☑ groundtruth.prj
- ☑ detections.prj

Activities | Events | Tracks | Scene Elements

- ☑ track-1
- ☑ track-2
- ☑ track-3
- ☑ track-4
- ☑ track-5
- ☑ track-6

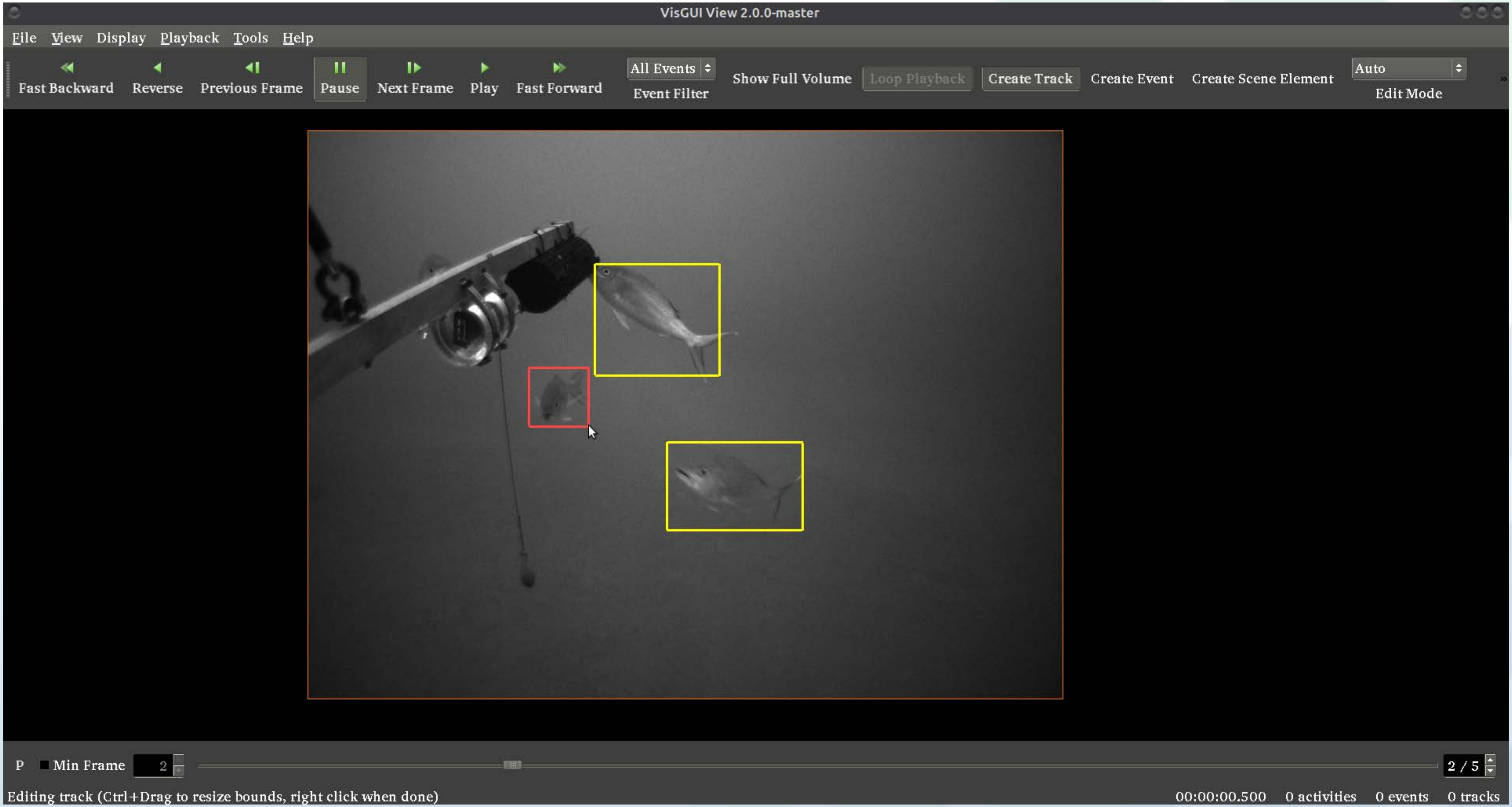
Sort By Id □ Descending

☑ Show Unchecked □ Show Excluded

P □ Min Frame 458 458 / 1926

00:03:48.500 0 activities 0 events 39069 tracks

GUIs for Displaying Videos and Annotations



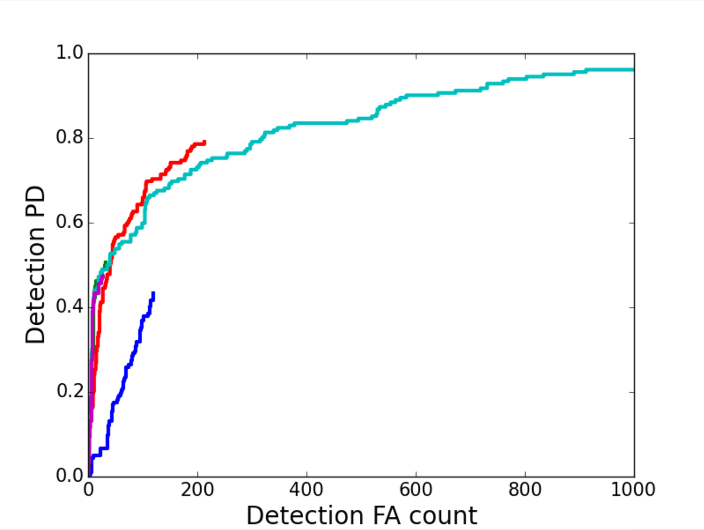
Editing track (Ctrl+Drag to resize bounds, right click when done)

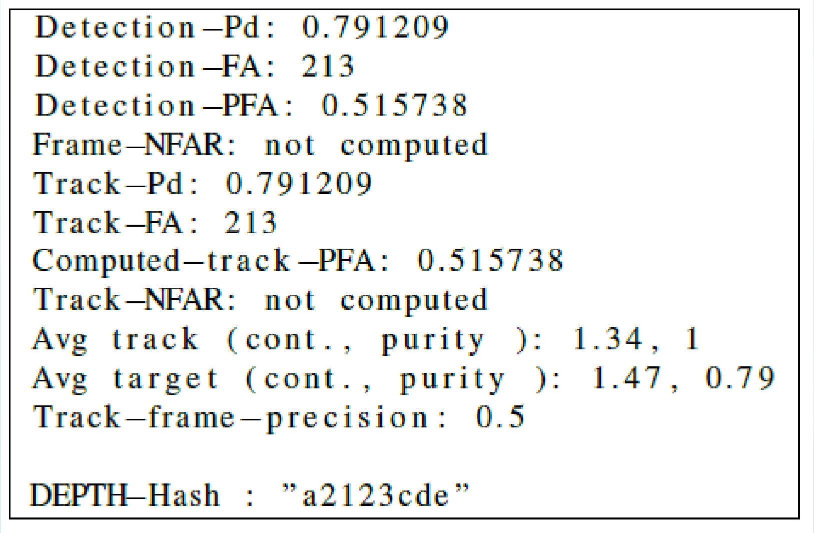
00:00:00.500 0 activities 0 events 0 tracks

Evaluation and Scoring

- VIAME includes an extensive scoring capability for measuring detection, tracking, and classification on images or video

- Existing annotations can be translated to VIAME-compatible formats



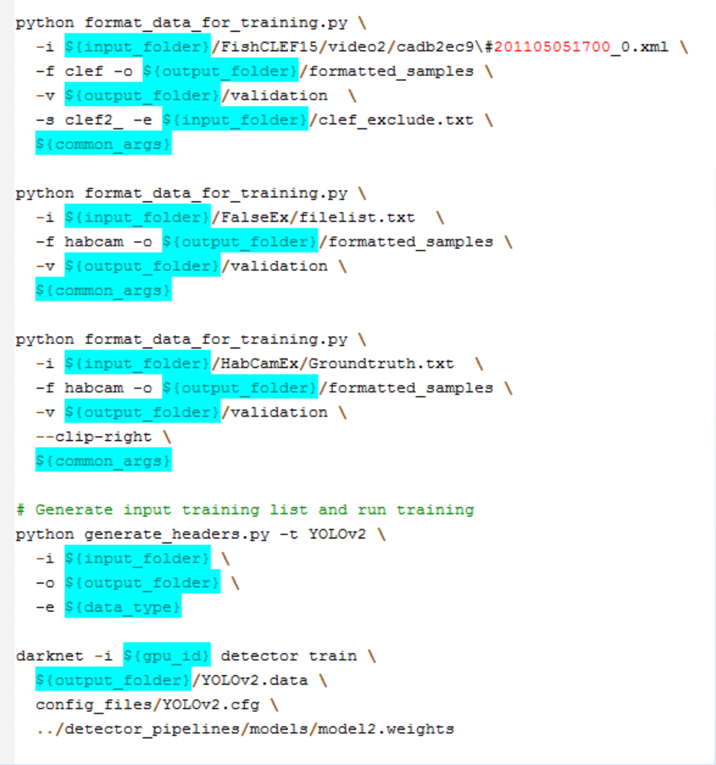


```
Detection-Pd: 0.791209
Detection-FA: 213
Detection-PFA: 0.515738
Frame-NFAR: not computed
Track-Pd: 0.791209
Track-FA: 213
Computed-track-PFA: 0.515738
Track-NFAR: not computed
Avg track (cont., purity ): 1.34, 1
Avg target (cont., purity ): 1.47, 0.79
Track-frame-precision: 0.5

DEPTH-Hash : "a2123cde"
```

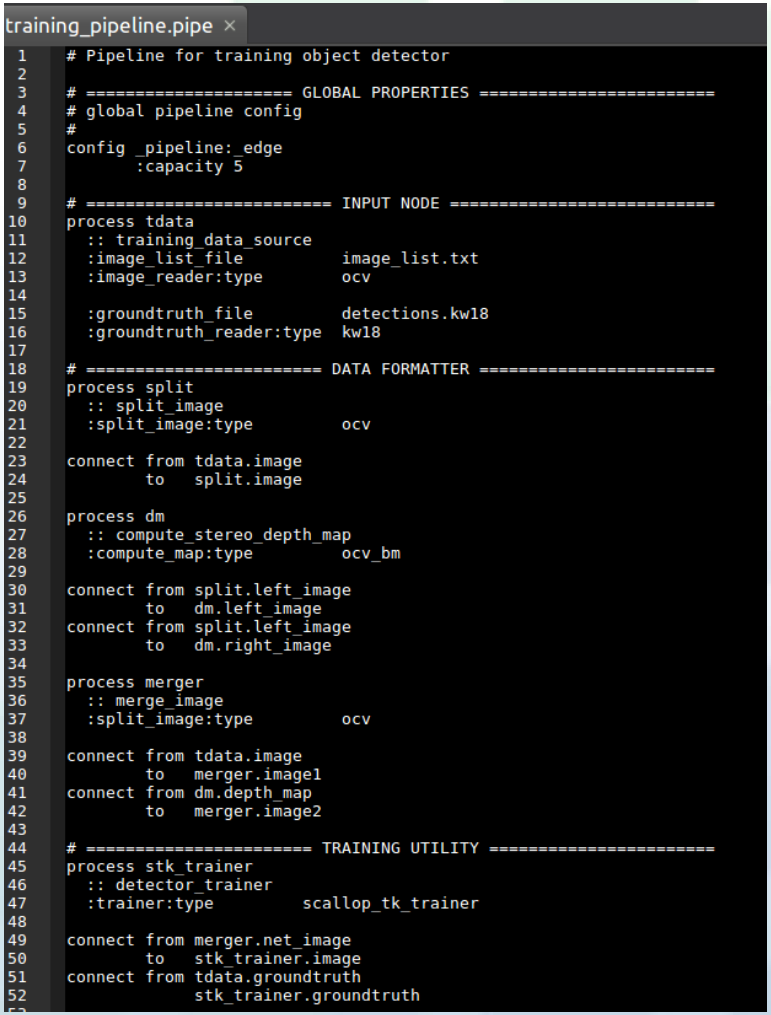
Model Training

Old:

New:



```
python format_data_for_training.py \
  -i ${input_folder}/FishCLEF15/video2/cadb2ec9\#201105051700_0.xml \
  -f clef -o ${output_folder}/formatted_samples \
  -v ${output_folder}/validation  \
  -s clef2_ -e ${input_folder}/clef_exclude.txt \
  ${common_args}

python format_data_for_training.py \
  -i ${input_folder}/FalseEx/filelist.txt  \
  -f habcam -o ${output_folder}/formatted_samples \
  -v ${output_folder}/validation \
  ${common_args}

python format_data_for_training.py \
  -i ${input_folder}/HabCamEx/Groundtruth.txt  \
  -f habcam -o ${output_folder}/formatted_samples \
  -v ${output_folder}/validation \
  --clip-right \
  ${common_args}

# Generate input training list and run training
python generate_headers.py -t YOLOv2 \
  -i ${input_folder} \
  -o ${output_folder} \
  -e ${data_type}

darknet -i ${gpu_id} detector train \
  ${output_folder}/YOLOv2.data \
  config_files/YOLOv2.cfg \
  ../detector_pipelines/models/model2.weights
```

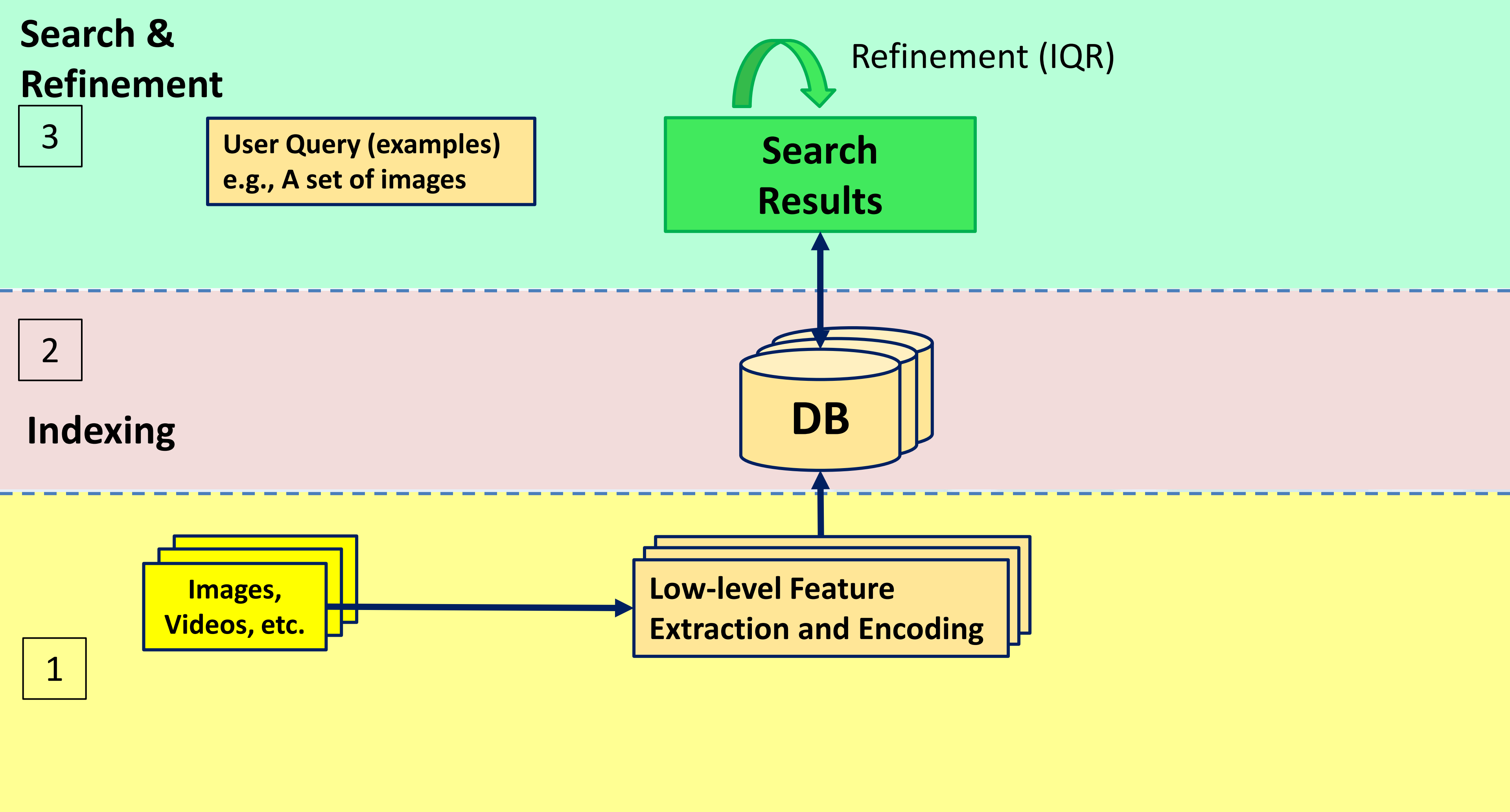


```
# Pipeline for training object detector

# ===================== GLOBAL PROPERTIES ========================
# global pipeline config
#
config _pipeline:_edge
       :capacity 5

# ========================= INPUT NODE ===========================
process tdata
  :: training_data_source
  :image_list_file            image_list.txt
  :image_reader:type          ocv

  :groundtruth_file           detections.kw18
  :groundtruth_reader:type    kw18

# ======================== DATA FORMATTER ========================
process split
  :: split_image
  :split_image:type           ocv

connect from tdata.image
        to   split.image

process dm
  :: compute_stereo_depth_map
  :compute_map:type           ocv_bm

connect from split.left_image
        to   dm.left_image
connect from split.left_image
        to   dm.right_image

process merger
  :: merge_image
  :split_image:type           ocv

connect from tdata.image
        to   merger.image1
connect from dm.depth_map
        to   merger.image2

# ======================= TRAINING UTILITY =======================
process stk_trainer
  :: detector_trainer
  :trainer:type           scallop_tk_trainer

connect from merger.net_image
        to   stk_trainer.image
connect from tdata.groundtruth
             stk_trainer.groundtruth
```

pipeline_runner –p training_pipeline.pipe

Kitware

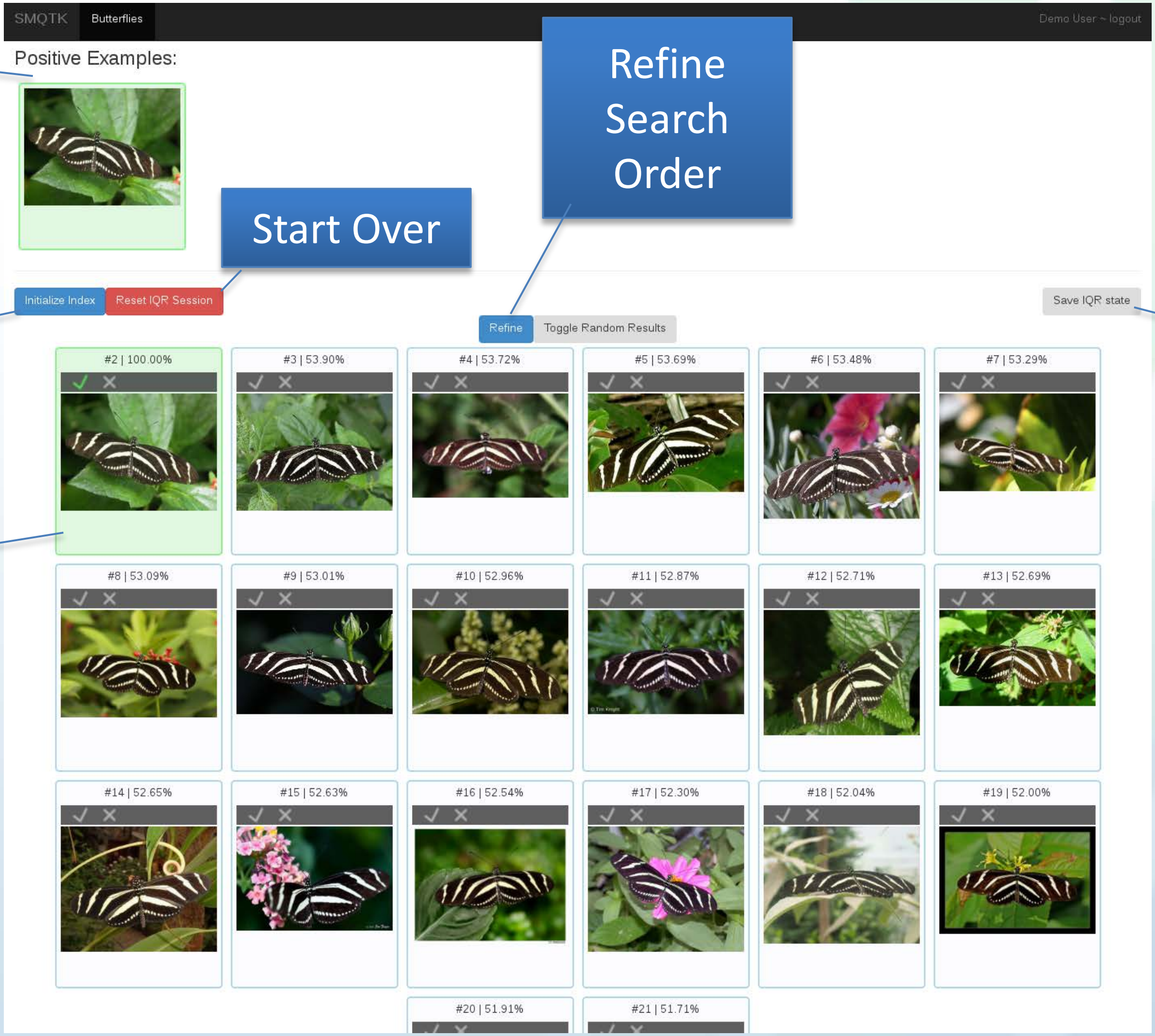
Interactive Query Refinement:
A User-Driven Search Work Flow



+ : good for similarity search

- : no semantic queries

Demo IQR Interface Example



Upload Positive Examples From Disk

Start Over

Refine Search Order

(re)Search Data set

Export Annotations for Classifier Constructio n

Mark Items Relevant or Not Relevant

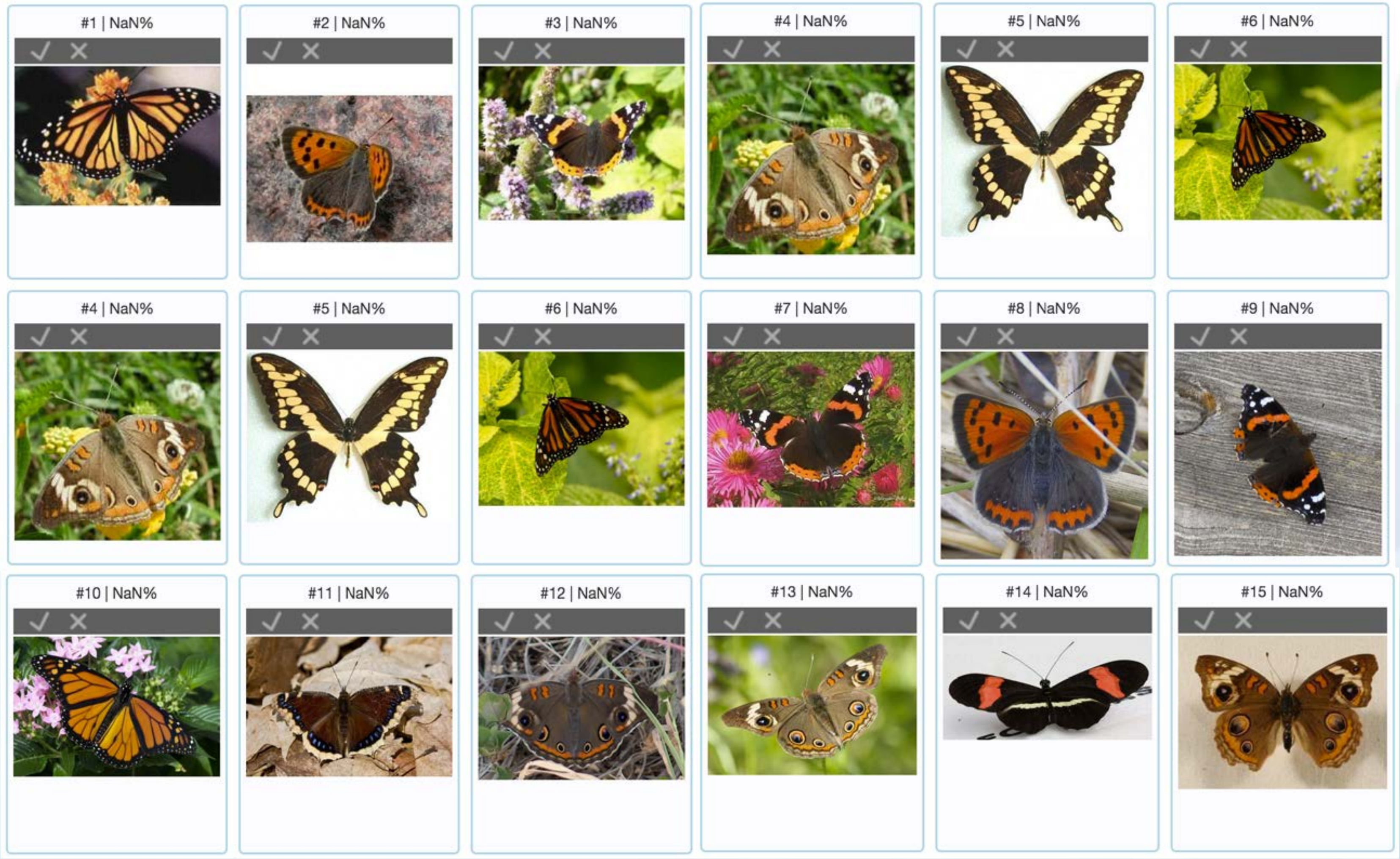
IQR Example

Start IQR with a single positive exemplar

Dataset contains 832 images with 55-100 images per type.

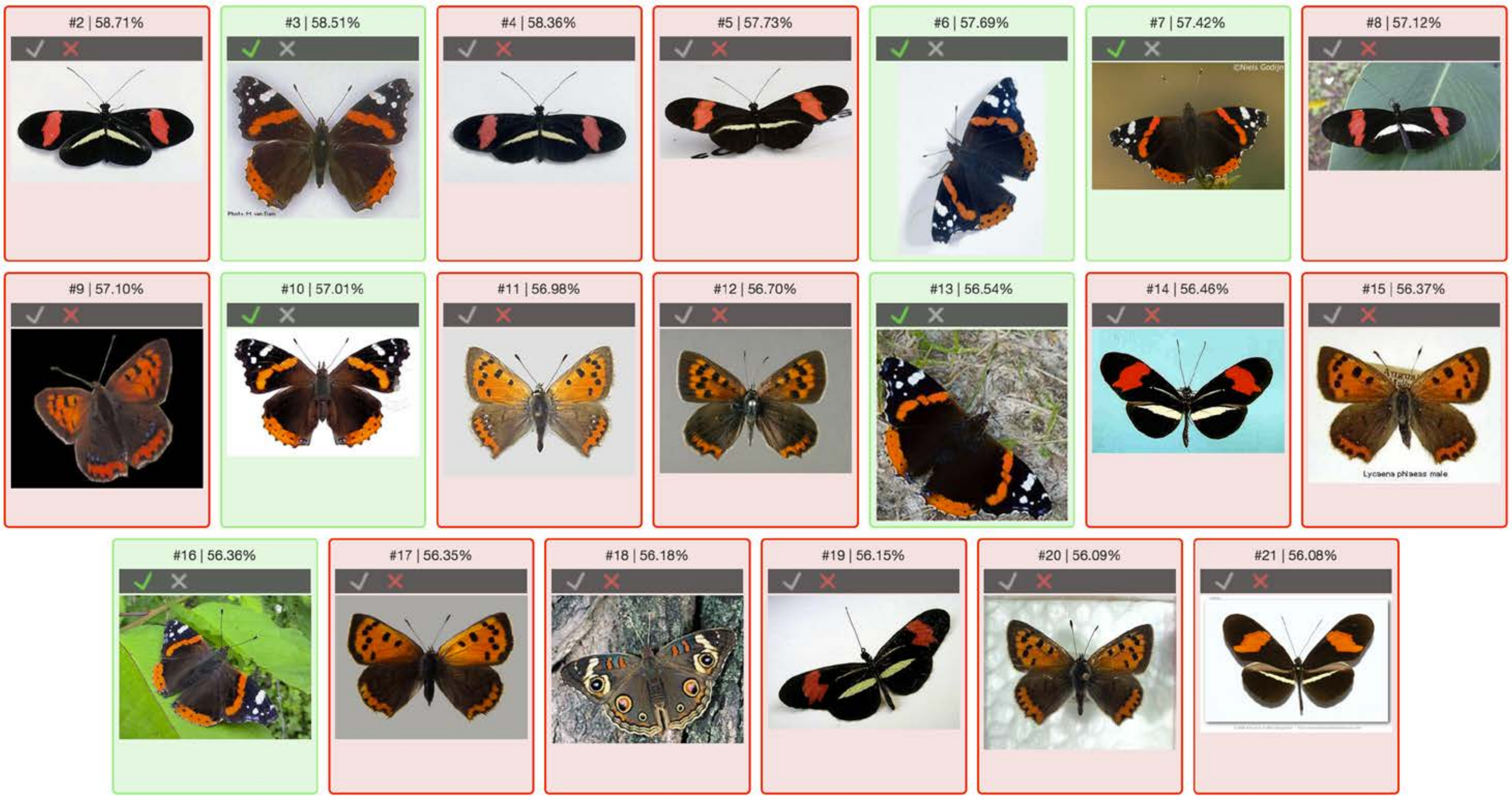
Use CAFFE AlexNet Layer 7 as an image descriptor

Random Selections from Leeds Butterfly Dataset

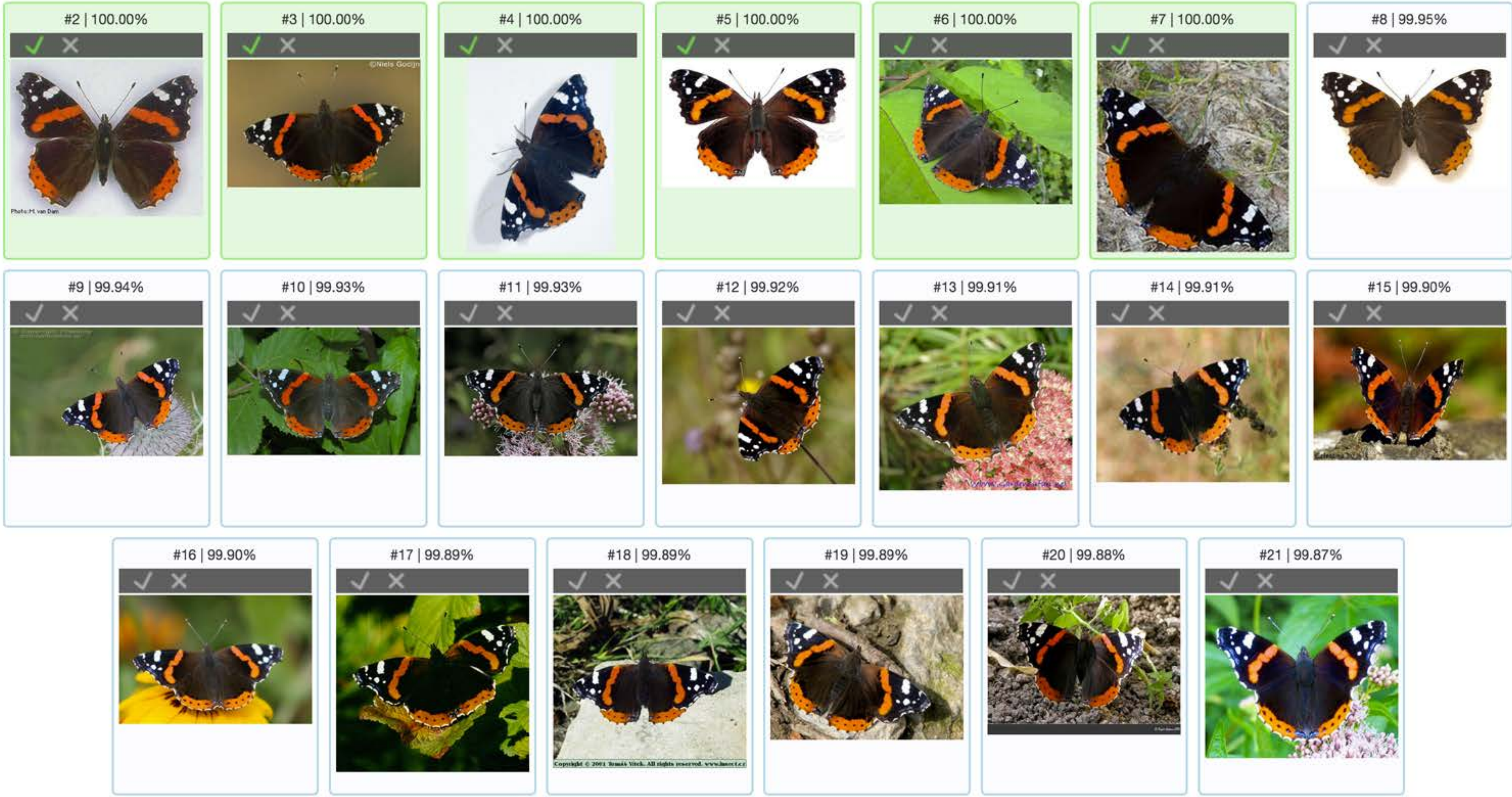


Josiah Wang, Katja Markert, and Mark Everingham
Learning Models for Object Recognition from Natural Language Descriptions
In *Proceedings of the 20th British Machine Vision Conference (BMVC2009)*

Results from Single Exemplar



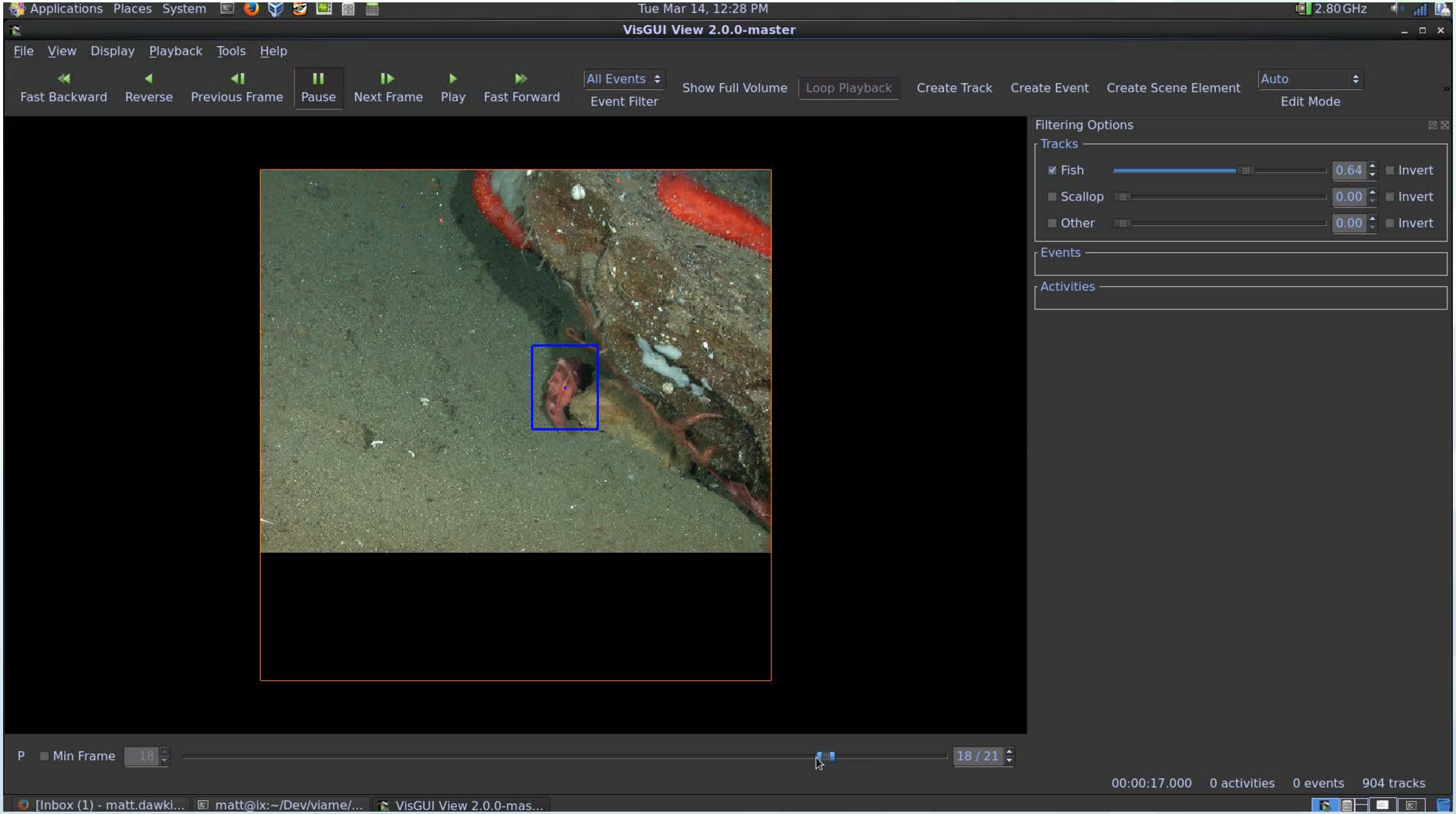
One Refinement Based on Adjudications from Previous Slide



Ongoing and Future Developments

- Improved video handling
- Improved detection and tracking
- Additional stereo processing
  - Calibration
  - Dense 3D reconstruction
- More integrated analytics
  - Anomaly detection and clustering
  - Habitat classification
- Large-scale visualization
- Extend deep learning integration
- Database extensions
- Make system easier to use
- Documentation

Demo



Event Filter: All Events

Edit Mode: Auto

Filtering Options

Tracks

☑ Fish 0.64 □ Invert

□ Scallop 0.00 □ Invert

□ Other 0.00 □ Invert

Events

Activities

□ Min Frame 18

18 / 21

00:00:17.000 0 activities 0 events 904 tracks

VIAME Resources

- VIAME is publicly-available, open-source software
  - viametoolkit.org/
  - Community contributions are highly encouraged, both framework additions and analytics
- Multiple benthic datasets previews are available at marineresearchpartners.com/nmfs_aiasi/Home.html

Kitware

Thank you NOAA!

Code Repository: https://github.com/Kitware/VIAME

Initial development and testing of VIAME was funded by the NOAA Fisheries Strategic Initiative on Automated Image Analysis. The findings and conclusions in the paper are those of the authors and do not necessarily represent the views of the National Marine Fishery Service, NOAA, or the government of the United States. The use of trade, firm, or corporation names in this publication is for the convenience of the reader and does not constitute an official endorsement or approval of any product or service to the exclusion of others that may be suitable.

Kitware