GStreamer for Engineering Video Processing Applications

Night Vision

Night Vision & Electronic Senso

TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

Joshua M. Doe, Stephen D. Burks

Overview

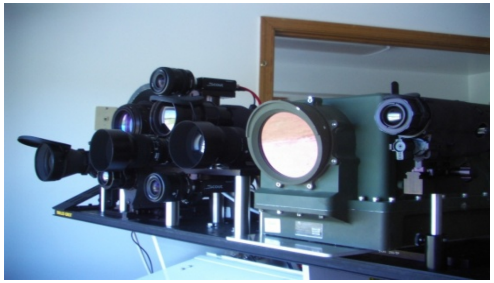
- Our organization
- Our relationship with GStreamer
- Our relationship with local GStreamer users
- Video capture
- Video processing and analysis
- KLV/Metadata
- Our vision for GStreamer
- OSS future
- Development Timeline

Our Organization-NVESD

- ***“CERDEC NVESD's mission is to conduct research and development in order to provide U.S. Warfighters with advanced sensor technology to dominate the 21st Century digital battlefield.”***
- We are interested in different imaging and sensing technologies, image processing algorithms, and physical packages in which these imaging technologies are kept.
- We are a large organization that has varied interests, including a wide variety of image capture and image processing platforms.

Our Organization-NVESD Modeling and Simulation

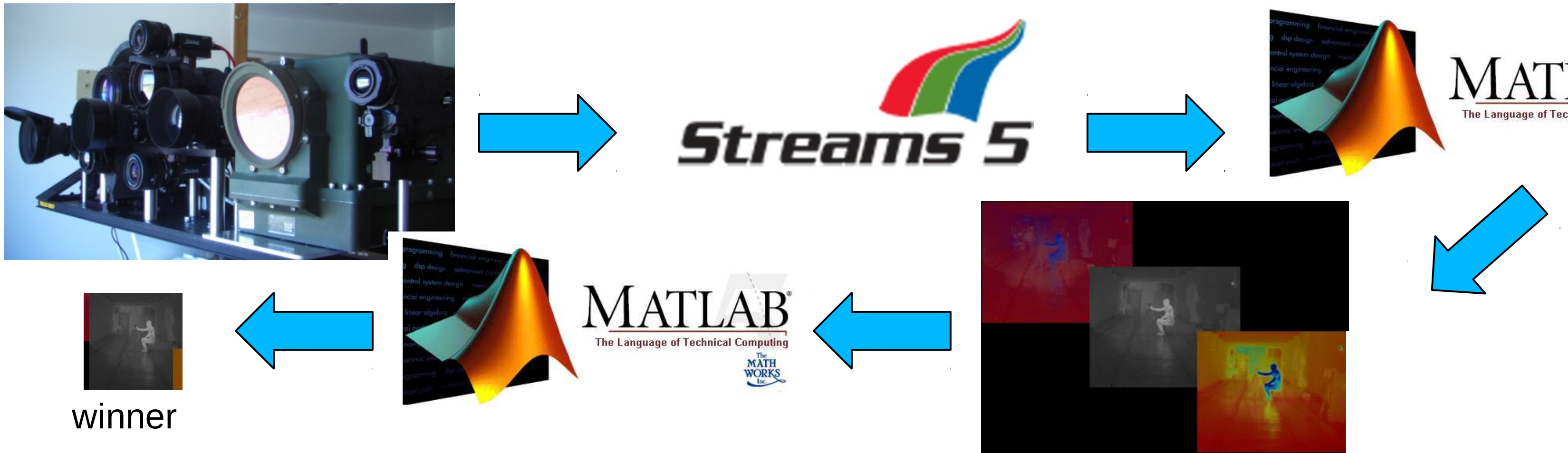
- SWAP-CP (Size, Weight, Power, Cost, and Performance) is a common trade space within the military, and our division's primary concern is imager performance
- We either collect imagery from different candidate sensors or we collect imagery from standard systems to apply image processing algorithms to either determine absolute sensor performance or relative sensor performance in the presence of those algorithms.



- We make perception tests using trained human observers to determine performance of those various sensors and image processing algorithms.

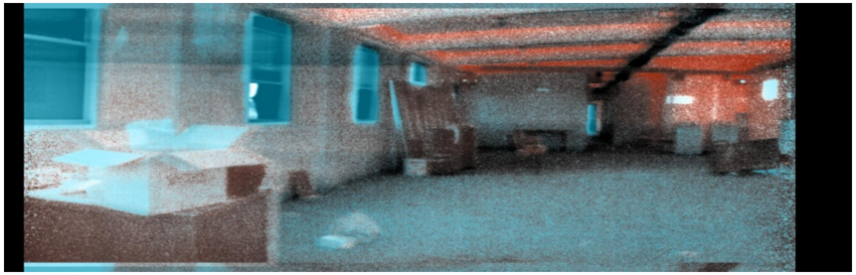
Our Organization-Challenges

- Because our interests are so varied, we have many different needs:
  - Ruggedness vs. performance in data collection
  - Power consumption vs. image processing
  - Data collection performance vs. size, weight, and power
  - Certain vendor capture cards vs. other vendor capture cards (form factor, inputs)
- Balancing requirements typically leads to larger than necessary effort in post processing
  - The total cost of a data collection does not stop once the imagery or video is collected
  - It is the aggregate sum of all of the costs associated with a collection, including data reduction, image segmenting, video sequencing, et cetera


Streams 5
MATLAB
The Language of Technical Computing
The MATH WORKS Inc.
MATLAB
The Language of Technical Computing
The MATH WORKS Inc.
winner

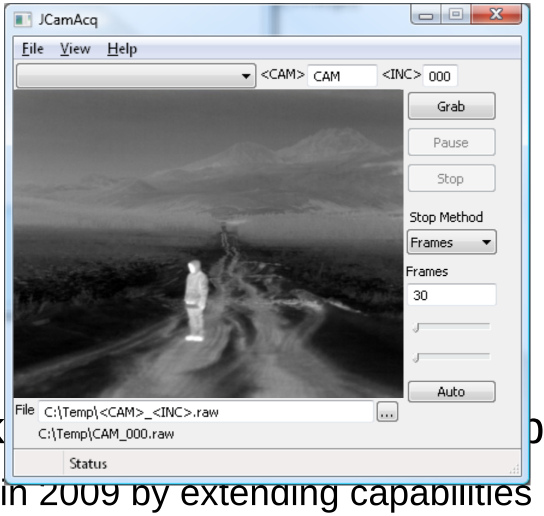
Our Organization-Wants

- We want capabilities for easily inserting new algorithms into data processing systems to see their performance enhancements/limitations.
- We want to be able to simultaneously compare two or more sets of processing algorithms while also saving the un-processed source video.
- We want to have software that is agnostic to different hardware platforms and vendor capture interfaces.
- We want to be able to create deployable stand-alone software packages based upon this work.



Our Involvement with GStreamer

- In 2006, had a need for capturing long (~30min) videos from Camera Link (an industrial machine vision camera interface)
  - No application existed that was affordable or easily adaptable
  - Created JCamAcq, simple capture application using National Instruments IMAQ



<CAM> CAM <INC> 000

Stop Method Frames

Frames 30

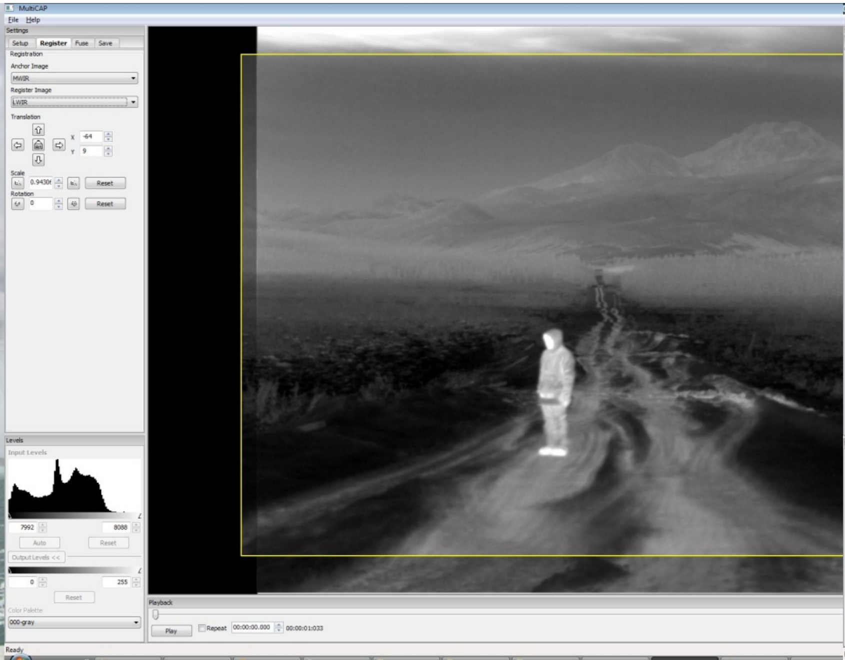
File C:\Temp\<CAM>_<INC>.raw

C:\Temp\CAM_000.raw

- In 2008, we were ask bed
  - Created NVMultiCAP in 2009 by extending capabilities of JCamAcq
  - Captured from up to five NI-IMAQ supported cameras
  - Could register and fuse any two source images

Our Involvement with GStreamer

- NVMultiCAP met its performance specifications for the customer. However, we wanted to progress further
  - Need modular code
  - User need to see intermediate steps



Anchor Image: MWIR

Register Image: LWIR

X: -64

Y: 9

Scale: 0.9430€

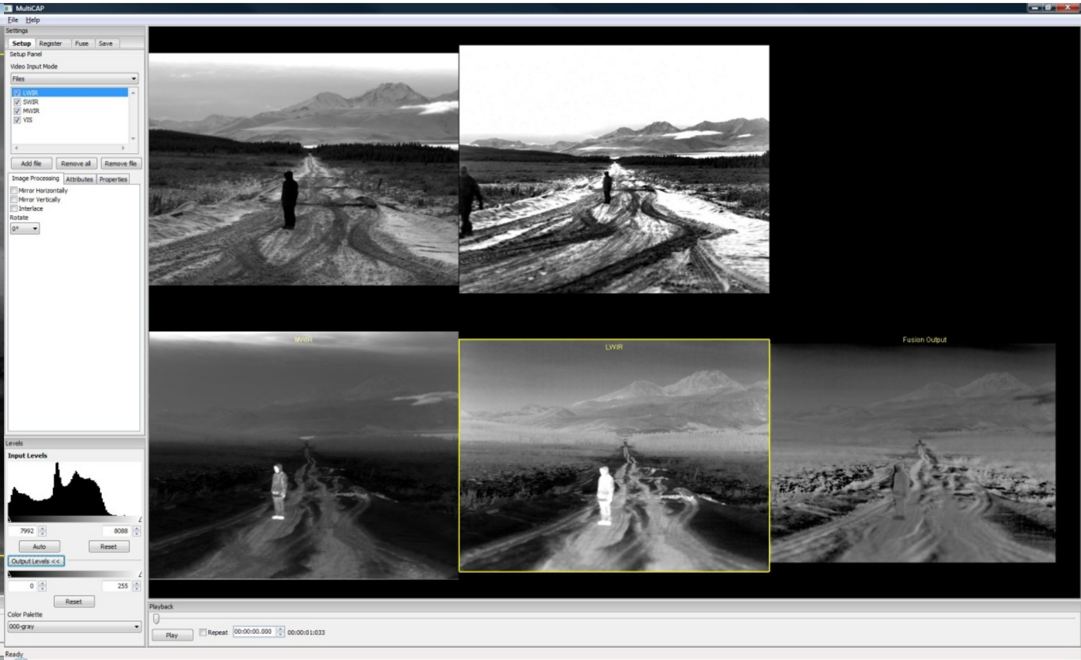
Rotation: 0

Input Levels: 7992 8088

Output Levels: 0 255

Color Palette: 000-gray

☐ Repeat 00:00:00.000 00:00:01:033



Video Input Mode: Files

☑ LWIR
☑ SWIR
☑ MWIR
☑ VIS

☐ Mirror Horizontally
☐ Mirror Vertically
☐ Interlace

Rotate: 0°

Input Levels: 7992 8088

Output Levels: 0 255

Color Palette: 000-gray

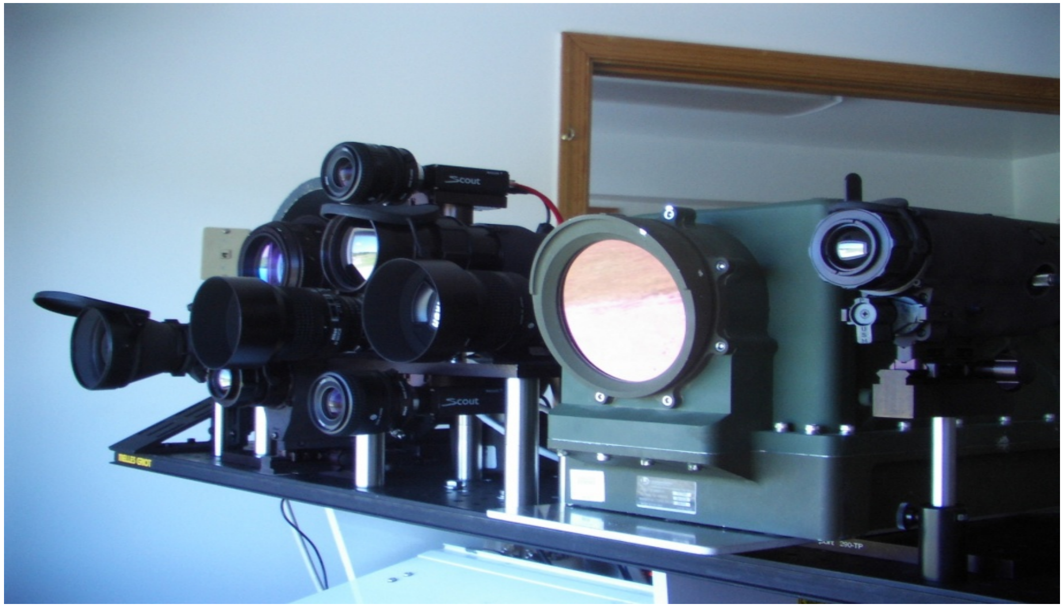
☐ Repeat 00:00:00.000 00:00:01:033

Our Relationship with GStreamer

- Continual enhancement of capabilities required us to move to a modular framework
  - We considered VLC, DirectShow, proprietary solutions, GStreamer, etc.
- We chose GStreamer because it met our requirements
  - Project vitality
  - Large plugin collection
  - Time synchronization
  - Rapid prototyping
  - Python support

Video Capture-Overview

- Want multi-spectral imagery that is spatially and temporally registered
- Researchers also like to have multiple variants of a certain type of sensor, such as a smaller and larger pixel pitch
- Some data collections have as many as 10 sensors going into a single capture computer
- Need a means to display and save all this imagery without any loss



Video Capture-Vendor Card Support

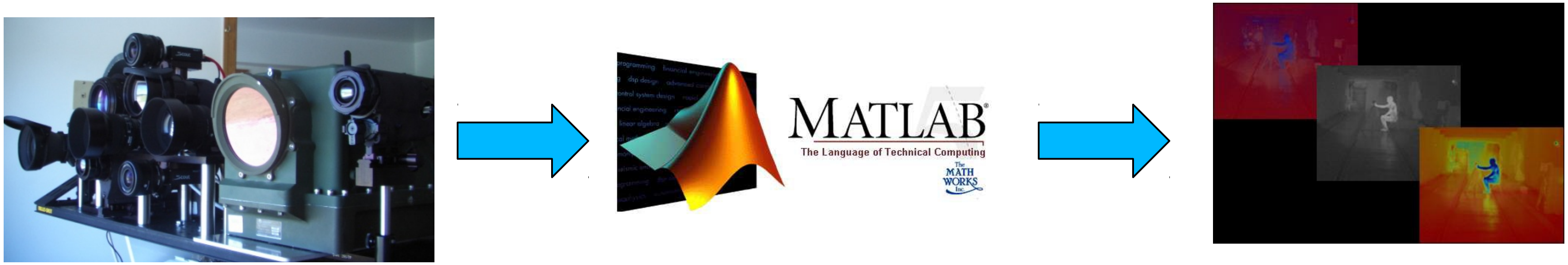
- No publically available GStreamer plugins for Camera Link, LVDS, CoaXPress, but some machine vision plugins exist
  - Aravis library for GigE Vision (Linux only)
  - IIDC plugin exists but not many imagers NVESD uses supports IIDC
- NVESD has GStreamer elements for some vendors
  - Blackmagic DeckLink (Windows patch accepted upstream)
  - Active Silicon (patch on Bugzilla)
  - National Instruments
  - Euresys
- NVESD is planning on adding support for the following capture cards:
  - DALSA/Coreco, BitFlow, Matrox, Foresight Imaging, EDT

Video Capture-Cost Savings of GStreamer Approach

- Software
  - Capture Software from two major vendors cost 5K and 10K per license
  - Captured data usually has to be re-processed once collected
- Hardware
  - Since GStreamer will support multiple vendors, users can choose less expensive options
  - Example: Need 4 RS-170 signals captured simultaneously. 4 NI 1410 cards cost approximately 4K, while one Euresys Picolo Pro card costs 0.6K. They will give identical imagery in GStreamer.
- Personnel
  - Less time re-processing imagery
  - Less individuals needed manning computers
  - Less need for data-loggers since that data can be incorporated into imagery

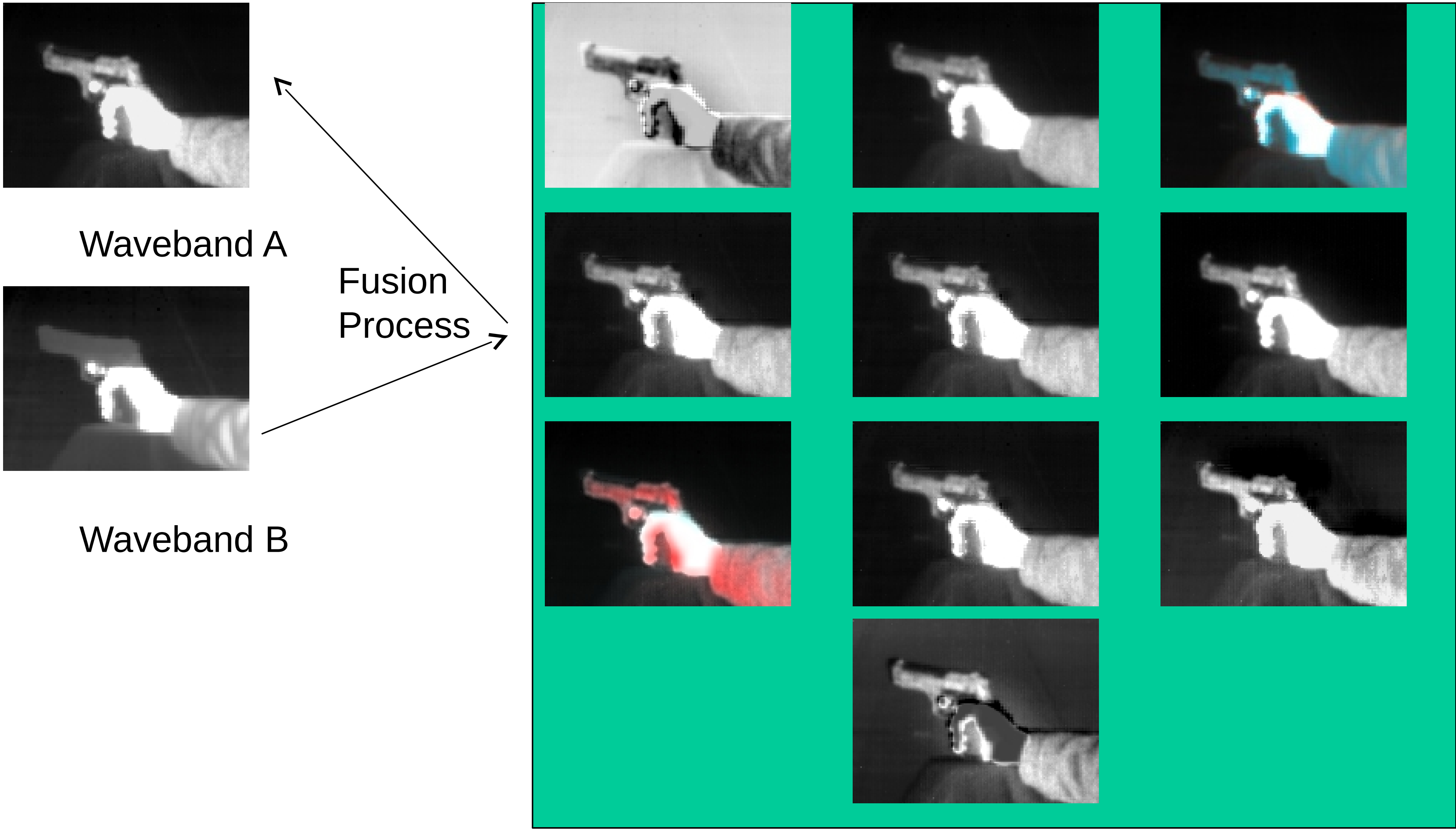
Video Processing and Analysis

- Most data is collected at one site, processed at another site, and then analyzed using additional software



- It would be of our best interests to combine all of these steps so long as:
- Data is not lost due to processing or analysis
- The interface is simple
- It is simple to add and modify processing or analysis steps

Video Processing and Analysis -Image Fusion



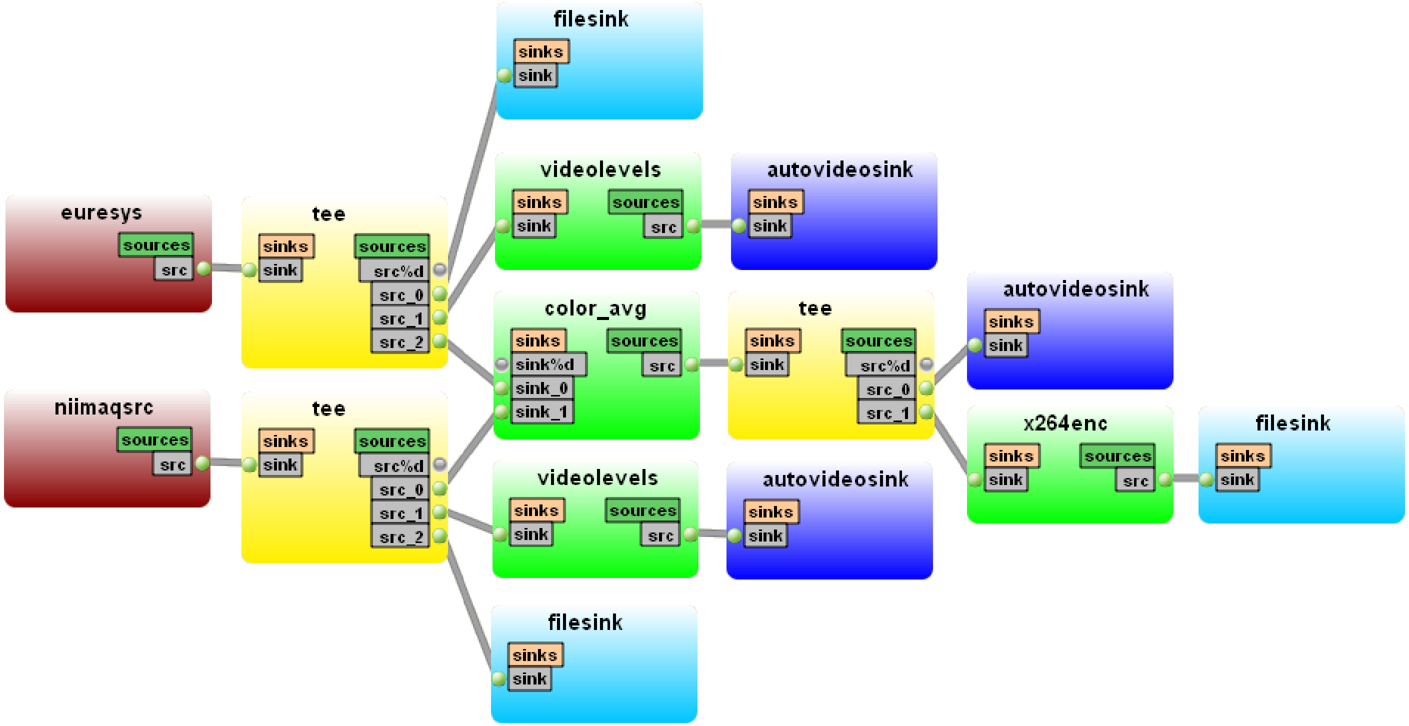
Waveband A

Fusion Process

Waveband B

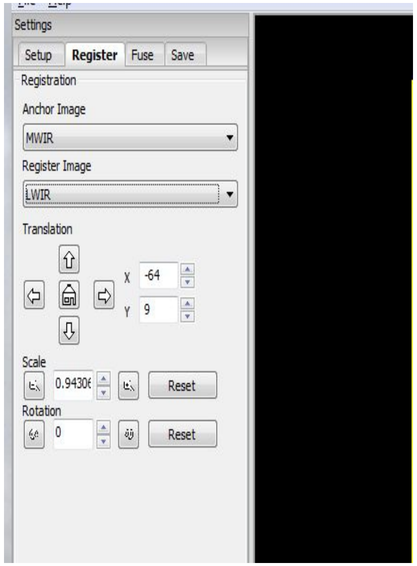
Video Processing and Analysis -Sample Created Graphical Pipelines

- Image capture source pipeline
- Timing negotiated by Gstreamer
- This pipeline has 2 video inputs, 4 processing elements, 3 video outputs, and 3 file outputs



Video Processing and Analysis - Image Registration

- Affine Transformations (shift, scale, rotate)
- Once we find algorithms for more advanced image registration, we will add them as elements



Settings

Setup | Register | Fuse | Save

Registration

Anchor Image: MWIR

Register Image: LWIR

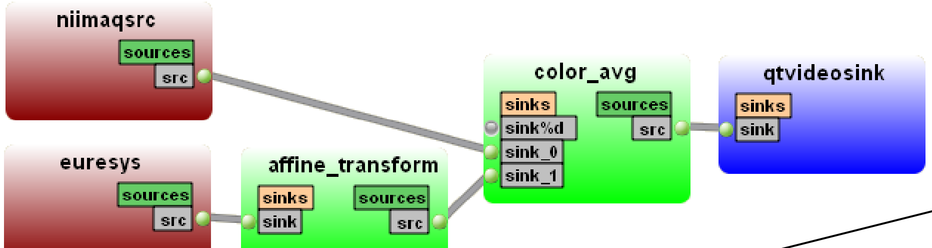
Translation

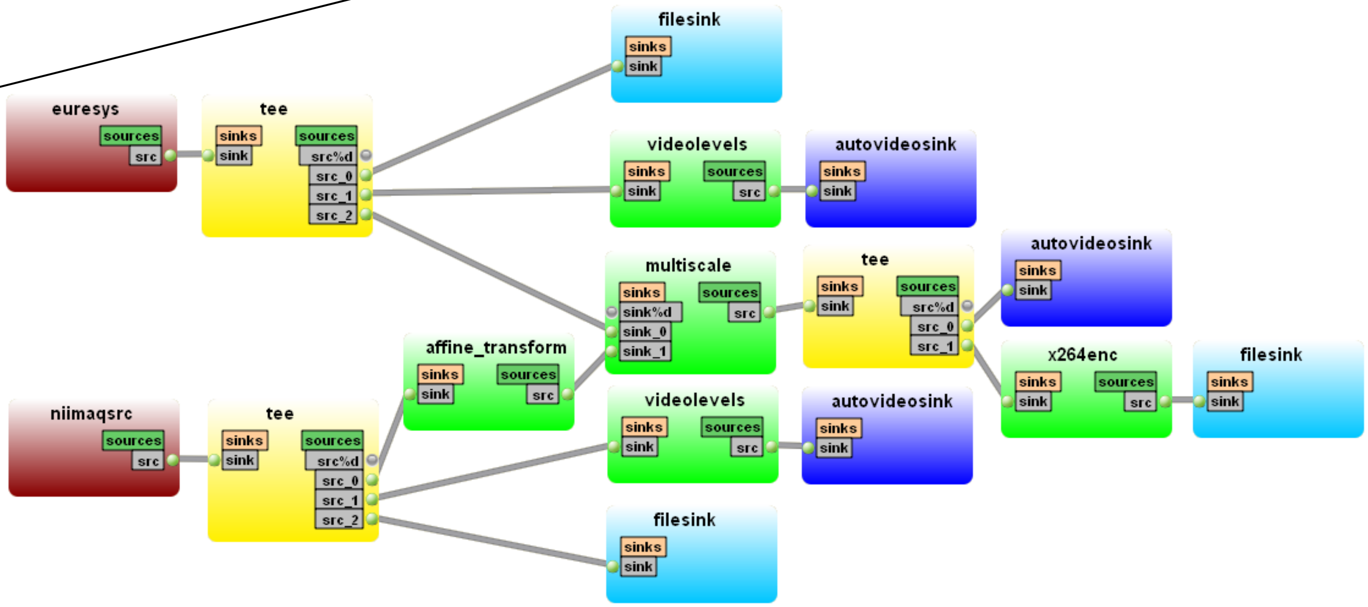
X -64

Y 9

Scale 0.9430€ Reset

Rotation 0 Reset



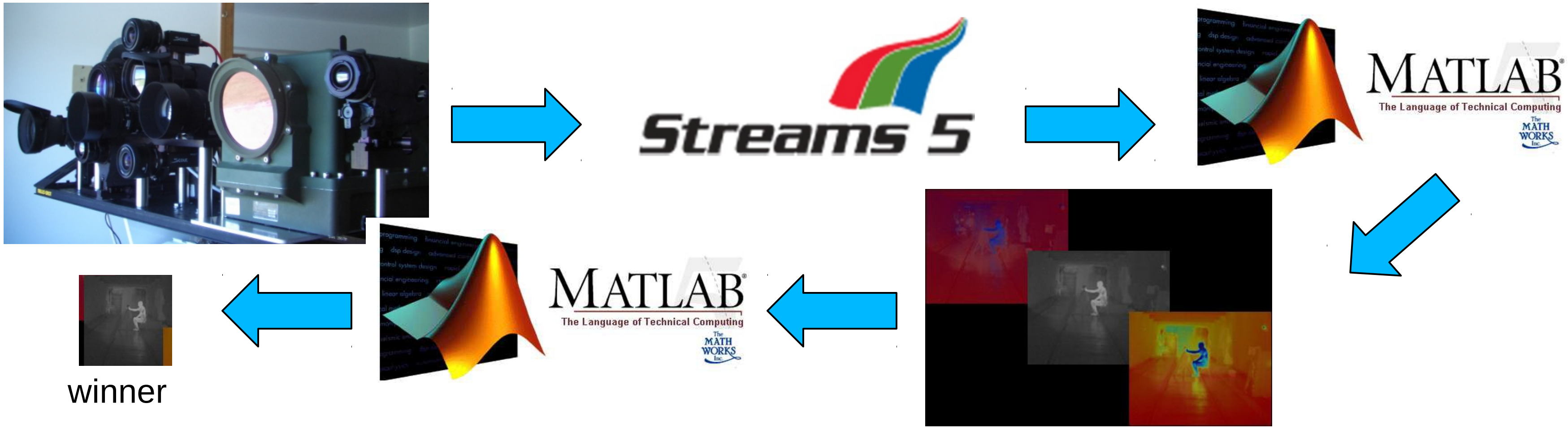


Video Processing and Analysis -Cost Savings of GStreamer Approach

- Rapid Prototyping
  - Can add, change, and remove GStreamer elements with ease
  - Image quality can be strongly affected by some image processing elements, while others will be benign to image quality
  - Faster time from prototype to engineering test unit
- Re-Use of Elements
  - Most image processing tasks are now standardized, so there is no need to re-code them multiple times.
  - It is the relative weights of these elements that make them unique for a particular program.

Video Processing and Analysis -Image quality

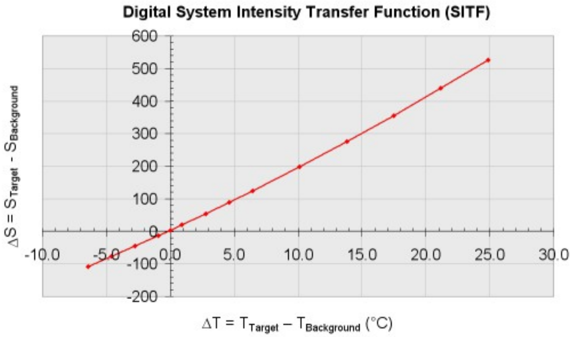
- To complete the task of determining what is the best image process, one needs to "score" imagery based upon certain metrics.

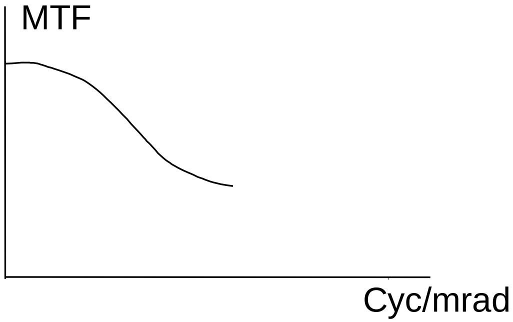

Streams 5
MATLAB
The Language of Technical Computing
The MATH WORKS Inc
MATLAB
The Language of Technical Computing
The MATH WORKS Inc
winner

- The Image quality tasks could be added as discrete elements into the pipeline for real time scoring of imagery
- If a certain image quality metric is determined to correlate well to an observer task (like navigation or identification), then a user can make image processing changes to raise that score for a particular task.

Video Processing and Analysis -Objective measurements

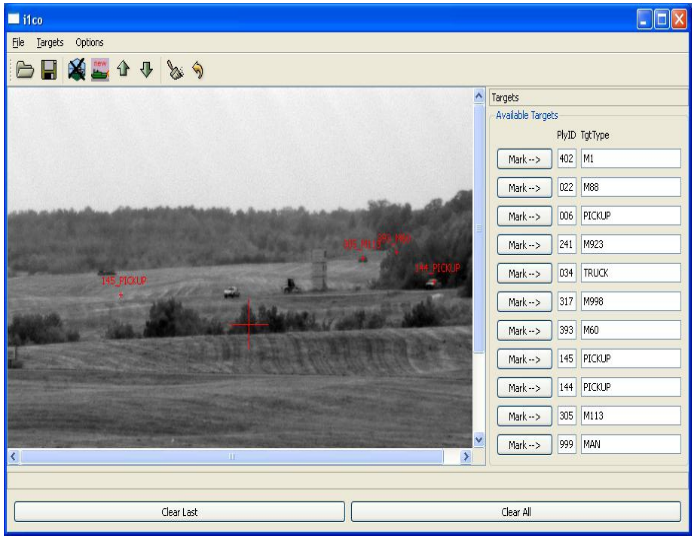
- Want to have ability to do “on the fly” measurements of objective image quality measurements
  - Resolution (MTF)
  - Noise (3D Noise)
  - Responsivity (SITF)
- These measurements can determine if your sensor is focused properly and if its electronics are working properly



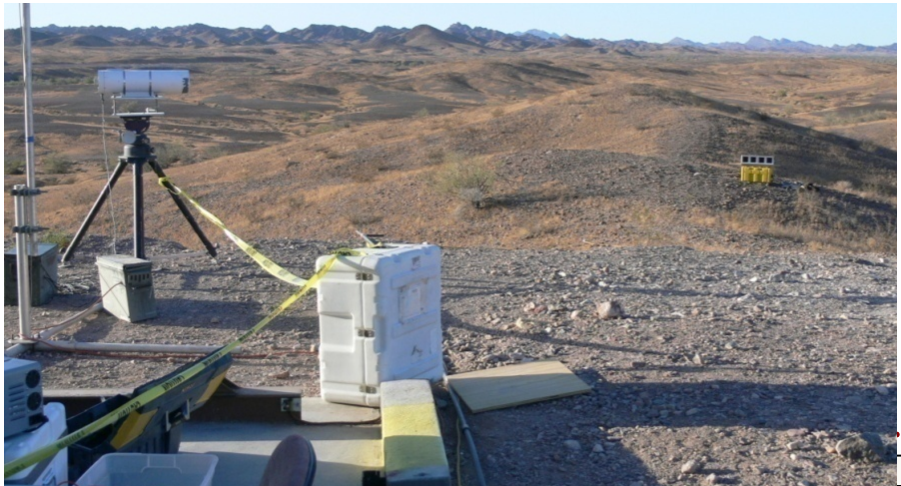


KLV/MetaData

- For data collections, it is beneficial to know all of the following:
  - Target, location, range, aspect
  - Weather, atmospheric transmission
  - Time of collection, sensor mode, sensor parameters
- Current file format provides for very few metadata items, and difficult or impossible to add more



	PlyID	TgtType
Mark-->	402	M1
Mark-->	022	M88
Mark-->	006	PICKUP
Mark-->	241	M923
Mark-->	034	TRUCK
Mark-->	317	M998
Mark-->	393	M60
Mark-->	145	PICKUP
Mark-->	144	PICKUP
Mark-->	305	M113
Mark-->	999	MAN



KLV/MetaData

- Need a format that can:
  - Store metadata items sampled both synchronously and asynchronously with video
  - Easily accommodate additional items
- Motion Imagery Standards Board (MISB) adopted SMPTE standards that can meet our needs:
  - Material eXchange Format (MXF)
  - Broadcast standard
  - Can store compressed and uncompressed audio/video
  - Plugin exists
  - Key-Length-Value (KLV)
  - Can be embedded in MXF, MPEG-TS, and others
  - SMPTE and MISB dictionaries provide for interoperability

KLV/MetaData

- Some have handled KLV in GStreamer by sending the information downstream in events
- Could also handle KLV as a separate stream, as many items have an associated timestamp
  - Participating elements could include combiner, muxer/demuxer (e.g. mxfmux/mxfdemux, MPEG-TS)

Making GStreamer Accessible to Non-C Programmers

- Most algorithm development occurs in commercial software such as:
  - MATLAB by MathWorks
  - Mathematica by Wolfram Research
- Algorithm testing usually limited to recorded stills or video
- Need to ease insertion of algorithms into GStreamer pipeline to test against live video

MATLAB

Wolfram Mathematica 8

MATLAB to GStreamer wrapper

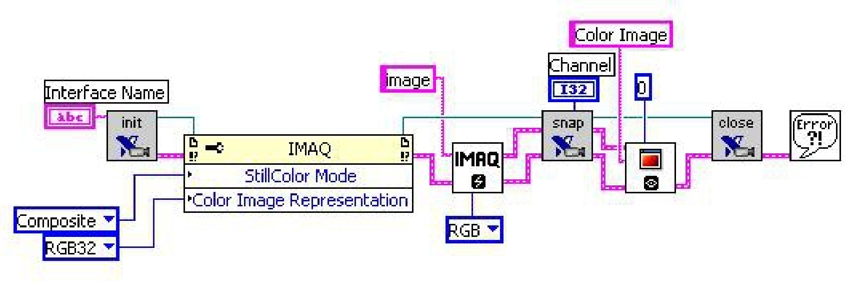
- Integrating existing MATLAB code into GStreamer
  - Reuses concept from frei0r plugins
  - User creates a few simple MATLAB functions
  - init, deinit, set_property, get_property, update, etc.
  - User runs build script to create plugin
- Advantages:
  - Quickly test MATLAB algorithms with GStreamer
  - Exploit large base of existing code
- Drawbacks:
  - Still requires use of MATLAB runtime
  - No significant speed improvements
  - Still tied to expensive proprietary software

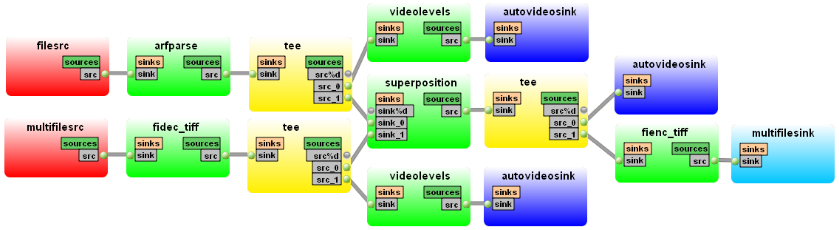
Python as a Transition Step

- Educate on the possibility of porting MATLAB code to Python
  - Utilize numpy, scipy, matplotlib, etc. through Python(x,y)/Spyder
  - Barriers to adoption at large organizations:
  - Availability (MATLAB site licenses)
  - Familiarity (MATLAB taught by most universities)
- NVESD created blackbody test in Python
  - Read and set blackbody temperature over GPIB
  - Capture video using GStreamer
  - Calculate metrics on frame using GStreamer
  - Adjust temperature to reach target metric

Our Pipedream for a GStreamer Application Builder

- Allow non-programmers to easily create custom GUIs for data collections
- Create GUI plugins (widgets) that can be connected to:
  - Properties (e.g. sliders, file choosers)
  - Elements (e.g. histogram display/levels controller)
  - Bins
- User can drag and drop to build pipeline and GUI
  - Similar to LabVIEW, but much smaller in scope





How to go OSS/OTD

- NVESD has limited experience releasing Open Source Software
  - Mil-OSS community provides support
- Contributions to GStreamer starting small
  - Small bug fix for DeckLink capture card (#655362)
  - Source element for Active Silicon Phoenix (Camera Link, #666385)
- Not all code can be released
  - Open Technology Development [1] – be as open as you can
  - Build community of interest within DoD and in scientific community
  - Put all non-OSS-releasable code in limited-access site for open development within the DoD

[1] http://mil-oss.org/resources/books/open-technology-development

Contributing Upstream

- Enhancements/bug fixes to existing plugins
- Plugins for commercially available hardware
  - Mostly capture cards
- Plugins implementing publicly available algorithms
  - Contrast Enhancement
  - Local Area Processing
  - Image Registration

Development Timeline

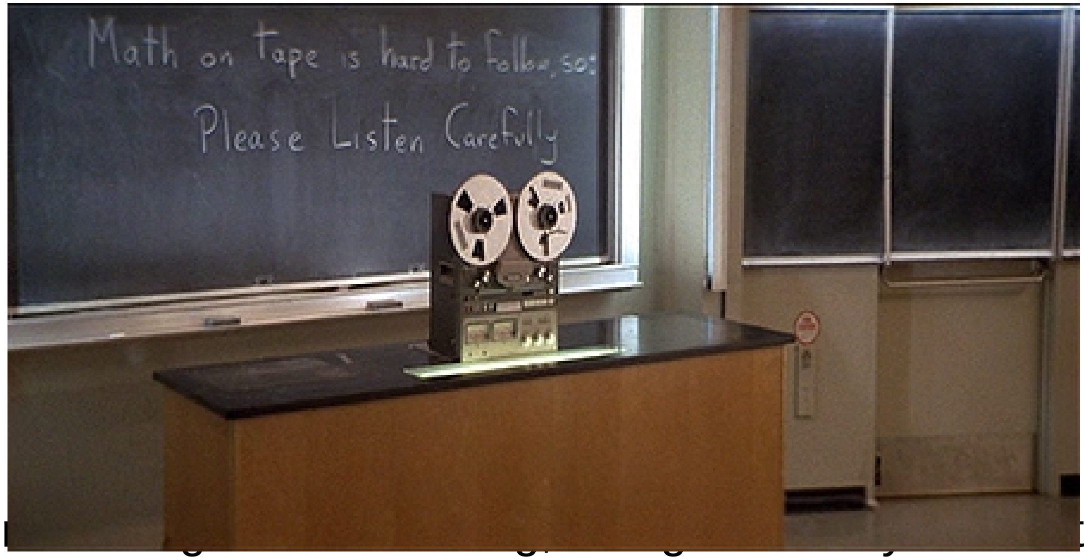
- Near Term
  - Single image/video capture through all supported vendors
  - Active Silicon
  - National Instruments
  - Euresys
  - Blackmagic Decklink
  - Basic image processing
  - Histogram stretch
  - Data counts on live imagery
  - Saturation points noted on imagery (through blue/black overlay)
  - Video Analytics
  - Super-Resolved Modulation Transfer Function
  - 3D Noise
  - Signal Intensity Transfer Function
  - Automatic Minimum Resolvable Temperature

Development Timeline

- Long Term
  - Multiple image/video capture with additional support for vendor hw
  - Matrox
  - Dalsa/Coreco
  - Bitflow
  - Additional image processing capabilities
  - Image fusion
  - Imager registration
  - Local Area Contrast Enhancement/Local Area Processing
  - Geo-rectification of imagery

Summary

- GStreamer is integral for our new developments in video capture and video processing.
- Please talk to conference planners for our contact information if you wish to contact us directly.



- Thank you fo ttle *Real Genius*-ey