A Generalizable Radiography Algorithm Test Environment for NDE Applications

Andrew C. Wantuch, Jaxon M. Gittinger, Ismael Perez, Edward S. Jimenez
Sandia National Laboratories, Software Systems R&D
So What Who Cares?

- **Topic**
  - ATR Algorithm Development

- **Problem 1**: Need for a fast, *open architecture* ATR test environment that doesn’t require access to systems to evaluate different types of 3rd party ATR algorithms
  - **Solution**: Provide 3rd party ATR algorithm developers with a way to develop ATR algorithms using pre-existing scans
    - Inspired by earlier efforts at ALERT for automated scoring (TO4)

- **Problem 2**: No standardized, open architecture method of comparing ATR algorithms
  - **Solution**: Provide a standardized way to benchmark algorithms

- **So What?**
  - Reduced barrier to entry for algorithm developers
  - Potentially speed up certification processes
  - Build confidence for adoption of 3rd party contributions
Open Threat Assessment Platform (OTAP)

- Develop and demonstrate an open architecture baggage screening prototype
- What is “Open”?
  - Standardized across vendors
  - Modular
  - Plug-and-play
- Allow 3rd Party Development of:
  - Hardware
  - Software
  - Algorithms
- Partner with security technology manufacturers
Test Environment Objectives

- Evaluate the performance of algorithms developed by third-parties using a common image database
  - Standardized metrics
  - Standardized timing
  - Programming language agnostic

- Be simple and easy for algorithm developers to use
  - No complex emulators
    - Emulators often need every component of the screening system implemented
  - Be highly flexible to support all conceivable algorithms
    - Variable input/output methods
    - Nontraditional approaches

- Enable iterative algorithm development
Design

Image Data
- Image 1
- Image 2
- Image N

Ground Truth DATA
- Signal Locations 1
- Signal Locations 2
- Signal Locations K

Task Execution
- Raw Image
- Standard Image
- Comparator

Metrics
- False Positive
- False Negative
- % Matching Threat

Data Visualization

Library of Algorithms
- Standard Converter
- Algorithm
Example

- **Algorithms: SIFT and SURF**
  - Popular computer vision algorithms
  - Identify features in images such as corners and changes in contrast
  - Only feature locations used for this example

- **Database: Radiographs of various COTS components**

- **Ground Truth: Features extracted by Matlab SIFT**

![SIFT Features](image1)

![SURF Features](image2)
Example – Continued

Ground Truth: 1948 points

Algorithm 1: Use 5 Points

Algorithm 2: Use 25 Points

Algorithm 3: Use All Points (349)
Results
Results
Conclusions

- Implemented a functional Prototype in Matlab
  - Likely supports algorithms written in any programming language
    - Tested with Python and Matlab
  - Generates standardized metrics for algorithms
  - Compares multiple algorithms or multiple versions of the same algorithm
  - Helps with rapid and iterative development of new algorithms with lower barrier to entry
Next Steps

- Support DICOS files as input
- Support CT datasets
- Determine method of deployment
  - Web app?
  - Distribute to 3rd parties?
  - Keep in-house at TSA/SNL?
- Investigate security concerns
  - How can we securely execute someone else’s executables?
- Work with vendors to provide what they want/need
Questions?
Backup Slides
OTAP Enables Plug-and-Play

Examples are Notional

Sensor 1, 2, …
Algorithm 1, 2, …
GUI

Component A
Component B
Component C
Component D

Open Platform Software Library (Middleware)
Operating System
CPU

TSE (AT-2, EDS, CAT, AIT, ETD, etc.)

OEM Control s/w
API (Function 1, 2, …)

OEM or 3rd Party
OEM-Developed
TSA-Developed

STIP I/O