ADSA 06
Fusion Development and Deployment

November 8, 2011

Dr. Matthew Merzbacher

Manager, Machine Vision & Innovation

SAFRAN
Morpho Detection

Conclusions

- Devising a clear and simple systematic fusion framework is prerequisite to meaningful sharing of data and results, and is necessary for smooth operations involving fused systems
- Having a good framework for talking about performance does not mean that one can talk (openly) about performance
  - But it helps (and is required)
- The more complex the fusion…
  - The trickier the testing
  - The less certain the conclusions that can be drawn
  - The more likely that a corner case will arise
- Fusion frameworks must be scalable and allow systemic and component testing & evaluation
- Fusion should enhance ConOps, not cripple it

3 Questions

- How do we share strengths & weaknesses of systems to allow (better) fusion?
- How do we test fused systems?
- How does fusion affect concept of operation?

MDI Lessons Learned from Data Fusion

Prerequisites for Data Fusion Development:

1. Core sensor knowledge for both systems
   - Full cooperation from sensor experts and algorithm people
2. Access to threat and false alarm data
   - Joint data collection desirable for test & validation

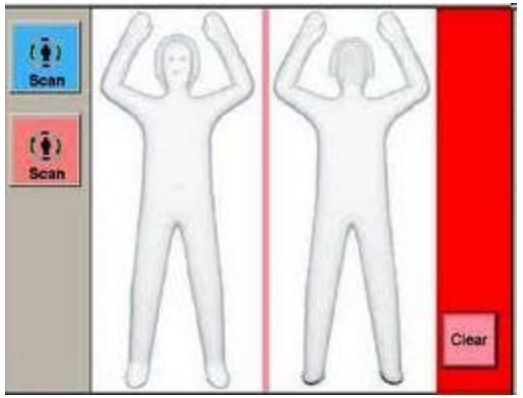
- **Both conditions requires tapping into IP**
  - **Difficult playing field between vendors (or vendor & academia)**
- **Could two entities make contributions without sharing IP?**
  - **With a shared framework, sure**

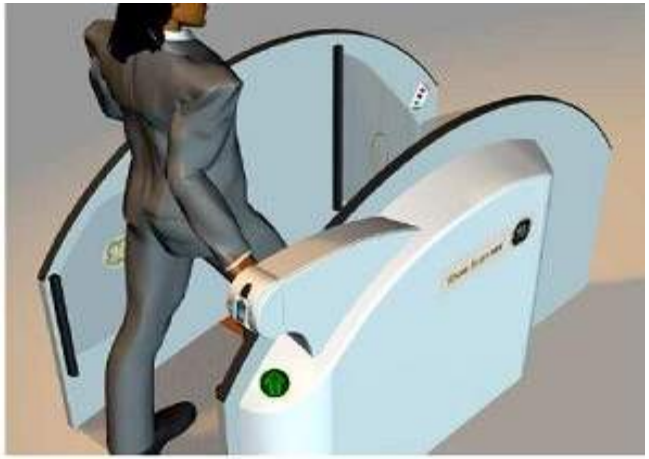
Sharing of Performance

- How can the government share performance with researchers and potential new vendors?
- How can vendors share performance with one another without giving up “secret sauce”?
- What is the performance information that must be shared?
- Two examples:
  - AIT + Shoe Scanner
  - CT + XRD

AIT + Shoe Scanner: Shared Responsibility

- Let's suppose AIT doesn't perform well on shoes
- A separate shoe scanner seems the ideal solution
  - Already a fused system
- Can we speak meaningfully and honestly about how well (or badly) each of these perform and where the limitations are?
  - Avoid gaps
  - Avoid redundancy
  - Drive performance





Shoe scanner

CT + XRD: Perspective on Performance

- Intended for false alarm reduction in checked baggage systems
- "Uptuned" one system to compensate for the other
- Strengths of one system allowed desensitization of the other for speed or detection performance
- Sometimes "meeting halfway" is the best approach
  - But how?
- Public method for sharing information (DSFP), but the data therein is still sensitive

Having a scheme for talking about performance does not mean you can talk about performance… but it helps

Testing

- **Adding systems adds corner cases**
  - **More degrees of freedom**
  - **Need to account for all components and the fusion**
  - **CT-XRD**
    - ✓ CT corner cases
    - ✓ XRD corner cases
    - ✓ Fusion corner cases
    - ✓ Other systemic corner cases (bag registration)
- **Testers sometimes apply selective memory or develop biased hypotheses – especially for fused systems**
- **Need to gather system data (threat & FA) that can also decompose into component data**
  - **Very hard across institutional boundaries**

Testing

- **Single-box testing is much easier than fused system testing**
- **So, why not treat a fused system like a single box?**
  - **Can't test pieces at different facilities (or on different timelines)**
  - **Hard to evaluate potential combinations, going back to example**
    - ✓ How do **N** AIT systems combine with **M** shoe scanners (each already fused)?
  - **Need to understand the source of failures**
    - ✓ Traceability for evaluation, improvement, and blame
  - **Once a system is qualified, want a fast upgrade path**
    - ✓ Test one component without retesting entire system

The more complex the fusion, the trickier the testing and the less certain the conclusions that can be drawn from testing

Concept of Operations

- ConOps is already complex
  - Detection/FA/Speed/Reliability requirements
  - Space
  - Cost
  - Ergonomics
  - Safety
- Fusion should be seamless – cannot add new requirements to an overtaxed system
- How is the data passed between fused systems? Framework!
- What happens to a fused system when one component fails or becomes overwhelmed? How do they communicate?
- Methodology should scale to evaluate “whole airport” fusion

Fusion should enhance ConOps, not cripple it

Conclusions

- Devising a clear and simple systematic fusion framework is prerequisite to meaningful sharing of data and results, and is necessary for smooth operations involving fused systems
- Having a good framework for talking about performance does not mean that one can talk (openly) about performance
  - But it helps (and is required)
- The more complex the fusion…
  - The trickier the testing
  - The less certain the conclusions that can be drawn
  - The more likely that a corner case will arise
- Fusion frameworks must be scalable and allow systemic and component testing & evaluation
- Fusion should enhance ConOps, not cripple it