ABSTRACT

SeeCoast extends the US Coast Guard Port Security and Monitoring system by adding capabilities to detect, classify, and track vessels using electro-optic and infrared cameras, and also uses learned normalcy models of vessel activities in order to generate alert cues for the watch-standers when anomalous behaviors occur. SeeCoast fuses the video data with radar detections and Automatic Identification System (AIS) transponder data in order to generate composite fused tracks for vessels approaching the port, as well as for vessels already in the port. Then, SeeCoast applies rule-based and learning-based pattern recognition algorithms to alert the watch-standers to unsafe, illegal, threatening, and other anomalous vessel activities. The prototype SeeCoast system has been deployed to Coast Guard sites in Virginia. This paper provides an overview of the system and outlines the lessons learned to date in applying data fusion and automated pattern recognition technology to the port security domain.

Keywords: Port security, video surveillance, track fusion, alerts, anomaly detection, homeland security

1. GOALS AND ARCHITECTURE

Automated scene understanding is a common issue throughout the Department of Homeland Security, including Coast Guard, Border Patrol, Transportation Security, and Infrastructure Protection. Numerous sensors have been deployed for these Agencies, engendering watch-stander manning as a growing problem. The goal of automated scene understanding is to reduce operator workload while maintaining effectiveness and improving detection of anomalous behaviors.

SeeCoast builds upon the USCG Hawkeye system, which uses a combination of surface surveillance radar and Automatic Identification System (AIS) beacons to track vessels and generate zone-incursion alerts. Hawkeye also includes manually-controlled and radar-slaved video cameras, an integrated command center, and a web-based information client. Hawkeye has been deployed to more than six operational US Coast Guard (USCG) Sector Command Centers. It allows watch-standers to surveil all vessel arrivals, departures, and transits within a port-centered area of interest.

Currently, watch-standers attempt to detect and assess possible anomalous behaviors using a large number of video cameras, radars, and AIS track data. SeeCoast has two goals: to increase the effectiveness of the watch-standers, and to increase their efficiency in surveillance operations. SeeCoast increases watch-stander effectiveness by adding capabilities to process video and infrared camera streams to detect, classify (small, medium, large vessel size), and track vessels; to fuse video tracks with radar & AIS tracks; to maintain seamless 24/7 vigilance on the track scene; to provide consistent alerting using established rules and learned normalcy models; and to free the surveillance system operators to assess behaviors rather than spending their time detecting behaviors. SeeCoast increases the efficiency of maritime surveillance operations by simultaneously examining data and tracks from many sensors, and by simultaneously assessing all tracks for alerting behaviors.

The desired end-state of the system is to automatically and reliably detect anomalies in the stream of maritime scene data while decreasing reliance on operator performance. SeeCoast achieves this by adding a number of new capabilities including: 1) video processing to detect, classify, and track vessels; 2) multi-sensor track correlation of video track data with the radar/AIS tracks; 3) automated camera control for track acquisition, ship size classification, and track
maintenance; 4) more sophisticated rule-based track activity analysis to reduce operator workload; 5) learning-based track activity analysis to increase operator performance; and 6) display enhancements for improved situational awareness and forensic analysis.

Figure 1 presents a block diagram of the architecture of the SeeCoast system. The capabilities listed above (and illustrated by dark boxes in the figure) are described in the following sections. Following component overviews, the final sections describe testing and deployment at the Joint Harbor Operations Center at Portsmouth, Virginia and conclude with lessons learned from development of this prototype automated scene understanding system.

2. CAMERA CONTROL

The camera controller is an integral part of the continuous feedback systems shown in Figure 1 used to extend the capability of the system to monitor several areas of interest simultaneously. Based on continuous reports of the current state of all tracked objects, the controller dynamically assigns the camera systems to observe particular vessels and collect visual information. The video information is processed (as described in more detail in Section 3) for three purposes: to estimate the size of the vessels detected by radar; to update vessel track information; and to detect and initiate tracks on vessels in areas of interest. Without a camera controller, the watch-standers would typically leave the camera aimed at a fixed aim point in order to detect vessels crossing this point. They would also use the cameras under manual control or slaved to a radar track to maintain surveillance on a particular vessel. Hence it is important that integration of automated camera control into an operational command center is operator-aware by deferring control of each camera to the watch-stander.

Camera locations were pre-determined by existing infrastructure at test/deployment facilities. Effective surveillance dictates that automated control should optimize the coverage provided by those sensors via intelligent resource coordination. For each camera to be controlled, a geospatially registered visibility map was created to indicate the visible areas within the field of regard and the areas that were occluded. The map was also coded such that areas near the camera were weighted more highly than more distant areas. This allowed the camera controller to choose the most advantageous camera for a particular track when multiple cameras were available. Each camera also had a list of fixed aim-points (a triple coordinate consisting of latitude, longitude, and a setting for the width of the field of view) that had to be visited on a regular basis. These preset locations are typically access regions or other important zones within a port. By forcing the system to survey these areas periodically, tracks can be initiated for purely visual contacts so that vessels cannot enter the operations area undetected even in the absence of reliable radar signatures or functional AIS transponders.

Figure 1. Dark blocks indicate SeeCoast components which control and process video data, fuse it with radar and AIS data into composite tracks, analyze the tracks to generate alerts, and display alerts for the watch-standers as well as send track information to Hawkeye and the Common Operating Picture (COP).
In periods between preset locations, cameras collect visual information on existing contacts (obtained from the track fusion component). The camera controller for SeeCoast was implemented as a simple rule-based manager to assign cameras to tracks. Cameras are assigned preferentially to tracks with either no visual information or with limited or stale visual information, since the only method for obtaining a vessel size estimate is via a video track component. A short-term list of video tracks visited by the camera controller was also maintained, along with the number of attempts to detect each track with the cameras) in order to prevent the camera controller from revisiting a particular track that may be difficult or impossible to detect via video (e.g., occluded by other vessels, during night-time, or due to poor aspect angle geometry). The reduced frequency of visits to difficult targets allows the system to maximize the amount of useful video information collected. The camera controller sets sensor pan and tilt according to the position of the vessel track, and also sets the zoom-level according to the predicted number of pixels needed at the estimated range to the vessel such that adequate information is available to the video processors after each pan/tilt/zoom operation.

3. VIDEO PROCESSING

Each camera has an associated video processing unit (VPU). Each VPU receives time-stamped video and estimates of camera orientation and field-of-view as inputs. The video processing component was designed to detect, classify, and provide track information for moving vessels in a moving background from a moving camera. The background in a maritime environment contains motion of the water, waves, clouds, cloud shadows, wind effects, birds, raindrops, and many other factors. The camera may be moving intentionally as when under manual or track-slaved control, or unintentionally as when shaken by the wind or nearby vibrations. Video processing must therefore be robust to significant continuous movement, making traditional approaches to automated surveillance inadequate for maritime applications.

Our approach is to estimate the motion of the background and to segment it into components. The motion of each component is estimated in the image plane. The main effect is due to the motion of the camera, which may affect several discrete partitions, and which typically affects the largest component. Once this component of the motion is estimated and isolated, it can be used to “stabilize” the image by removing this component from the other components. Remaining incoherent motion is typically due to weather and environmental effects, while remaining coherent motion is typically due to a foreground object. Thus, the locations of the foreground objects are measured along with their bounding image-plane partition dimensions. To aid in tracking, the detections are correlated across frames, and the waterline for each detection is more precisely measured. All of this information is passed to the track fuser.

Once a vessel is detected, measurements are made of the position and size of the vessel. The track fuser uses the orientation and zoom settings of the camera to convert the image measurements into world-based coordinates, and the pixel sizes into vessel length estimates. The position is used to associate the detection with an appropriate existing track, and to update the track using a sensor fusion algorithm. At this point, the track would be a composite track containing a video component, and so would no longer be at the highest priority for the camera controller on subsequent visits. The size of the vessel detection contributes to vessel length estimation. The video processor estimates the waterline and width of the vessel in the image. The track fuser uses the detected width together with the current zoom level of the camera, the estimated range from the camera to the vessel, and the heading of the vessel (to correct for aspect angle) in order to compute the actual vessel length. The length estimate may contain errors in any of these measurements, and so the length estimate is improved using a Kalman filter.

Track quality can be significantly improved by using an appropriate estimate of the measurement error. The primary error is in range estimation since seaport-based video cameras typically have unfavorable geometries involving cameras on masts or low buildings or towers. Range error increases significantly with range itself and can also be affected by the estimate of the measurement of the vessel waterline, or with even single-pixel errors in camera stabilization. Additionally, since range is used during the computation of vessel size, and vessel size a factor is generating alert cues for the watch-stander, repercussions of range estimation errors have many downstream effects. A careful error model was developed for this problem based on an analysis of the video processor across a variety of operating conditions.

During system testing and performance evaluation, a choice had to be made between generating extra false-positive detections, or potentially missing detections of vessels. For instance, motion-based vessel detection will have difficulty when the boat is moving directly toward the camera since the vessel may appear to be simply slowly growing over time,
leaving many pixels constant from frame to frame. To attain satisfactory performance in this circumstance, the parameters which control the size and magnitude of the coherence necessary to declare motion detection were set to be susceptible to being triggered occasionally by environmental effects such as waves. These extraneous detections were accommodated in the track fuser, which was configured to require several consecutive motion detections in order to initiate a new track.

The result was a video processor which detected vessel motions reliably, but was robust with respect to the background motions which would plague a less sophisticated algorithm. Furthermore, the detections were able to be used to reliably classify the size of the vessel and to generate and maintain position reports of the vessel sufficient for further video tracking or to augment a composite track identified from radar or AIS sensors.

4. TRACK FUSION

The All-Source Track and Identity Fuser (ATIF) uses a multi-source fusion algorithm that processes complementary data sources to produce a self-consistent, compact representation of the region of interest with low kinematic and tracked object classification errors and high track continuity. For input, ATIF can accept tracks output by any fusion engine or tracker and merges individual tracks and associated identification information with source reports enabling improved identification. The open architecture of ATIF is used as the basis of the SeeCoast track fusion component. In SeeCoast, ATIF fuses together track outputs from Hawkeye (based on SSR Engineering’s Radar System Tracker fused with AIS and Blue Force transponders data), and detections from the automated Video Processing component to generate an integrated harbor scene. ATIF simultaneously fuses these reports to generate, correlate, track, and identify all vessels within the sensors’ fields of view by generating position, velocity, and vessel size estimates.

The functional architecture of the generic ATIF system is shown in Figure 2. ATIF is based on multiple hypothesis tracking (MHT) technology developed under many previous tracking programs. The MHT technology enables recursive generation of association hypotheses between incoming tracks, evaluation of correlation scores for different association hypotheses and selection of the best fit association, fusion of state estimates, and prediction of states to future times. In addition, ATIF incorporates advanced algorithms for processing latent data. Complications of fusing data from multiple sources include data latency, high volumes of data input at differing rates, and differing types of data (e.g., raw reports, tracks with accuracy information). Various sensor systems incorporate differing processing and communication time delays and thus provide data at differing rates; video may provide multiple reports on multiple targets in less than a second while radar and AIS updates typically arrive several seconds apart. ATIF has the capability of processing out-of-sequence or latent data at differing input data rates in an efficient manner.

ATIF was developed to handle either Track-to-Track or Contact-to-Track correlation. In both cases, multiple hypotheses are carried to represent the association ambiguities. In the case of Track-to-Track correlation, an upstream tracking/fusion system has already put together multiple sources of information, e.g. SSR’s Radar System Tracker correlates reports from multiple radars, and assigns track numbers to unique tracks. Track information is then passed on to ATIF in the form of a Track ID and the associated track update. ATIF operates under the assumption that these track numbers are unique and strictly obeys the associations that are made by the upstream system. ATIF also has the capability to process contacts directly. ATIF fuses measurements of the bearing

---

**Figure 2.** ATIF has been developed under other programs to fuse a wider variety of sensor and tracker input data types using a multiple-hypothesis track loop.
and elevation angles to targets detected in video scenes by the Video Processing component with radar and AIS tracks to produce an improved estimate of the position, course and speed of the target. In addition to kinematic filtering, ATIF derives a vessel size estimate using measurements of the subtended angle of the visible aspect of targets; these measurements also being provided by the Video Processing component (see Figure 3).

5. RULE-BASED PATTERN RECOGNITION

The rule-based pattern recognition component – part of the situation understanding and alerting toolkit (SUAT) – can monitor a combination of streaming and static data feeds for predefined patterns that are associated with events of interest, generating alerts in real-time to keep operators aware of these events. The component was derived from BAE Systems’ Indications and Warnings Toolkit (IWT), which implements a three-stage approach illustrated in Figure 4. In the Domain Modeling stage, a data ontology is constructed describing the range of data sources and the attributes of data reports (e.g., fused tracks as a data type, with velocity as a data field). In the Pattern Definition stage, operators use GUIs to create patterns based on the data ontology. A simple example of a pattern is “any track whose current location is within a restricted area of the port.” Patterns can also involve comparisons of multiple fields in multiple types of data, and can be constructed modularly to allow for increasing levels of complexity. A GUI allows operators to script patterns based on the data ontology, walking the operator through a series of selections and questions that use information about the data environment to simplify the process. Operators can also create patterns from “templates” that only require specification of key inputs, automatically applying other pattern elements that were defined during template creation. As an example, for the simple pattern “any track whose current location is within a restricted area of the port”, the fact that the data being analyzed is track data and that it is the current location of a track that is relevant will always be true, so the GUI will ask the user to specify only the restricted area. Templates streamline the process of creating patterns that will repeatedly be applied with a small set of varying key attributes. In SeeCoast, the only inputs to the SUAT rule-based (SUAT-RB) component are the track streams from the track fusion component.

In the Pattern Matching stage, operators select a set of patterns to be monitored for, and the system generates alerts for matching instances. Alerts are highly configurable, incorporating free text entered during pattern creation and able to include specific details from matching data itself (e.g., incorporating the track ID in an alert message). The alerts are displayed in an alert window, together with alerts generated by the SUAT learning-based (SUAT-LB)
component. When available, a snapshot of the offending vessel(s) is presented with the alert to assist the operator in determining if the alert warrants a response.

We have shown that the pattern scripting capability can be used to describe a range of events identified by the Coast Guard as relevant for monitoring activity within a port. In evaluations performed on other deployments of the IWT technology, we have demonstrated an ability to maintain real-time alert generation given data throughput rates on the order of 500 new reports per second. In the port monitoring domain, this provides the capacity for effective monitoring of on the order of hundreds of vessels.

6. LEARNING-BASED PATTERN RECOGNITION

The objectives of the learning-based pattern recognition module are to learn normal traffic patterns and to detect deviations from normalcy. What is normal may differ between contexts—such as class of vessel, time of day, or season—so it is crucial that such learning be capable of discovering normalcy for different contexts. The possible combinations of conditions creating different contexts are numerous enough to defy efforts to manually define (and subsequently use in real-time) a complete set of rules to cover all cases. A continuously learning, adaptive system holds promise for dealing with such complexity without unrealistic involvement of expert input on a frequent basis.

To learn context-sensitive models of vessel behavior, SeeCoast employs a significantly modified version of the Fuzzy ARTMAP neural network classifier (as has been described more extensively in [1]). Use of this form of learning has also been applied successfully to a variety of image mining and tracking tasks [2-7]. The speed and performance of this learning algorithm makes it suitable for real-time and interactive situations wherein an operator/analyst can help teach the model via simple point and click actions. These reasons also make this technology suitable for event-level learning in maritime domain awareness (MDA). The challenge for SeeCoast is to develop this learning-based approach further to satisfy specific MDA needs.

Learning of normal traffic patterns can be achieved via training with a set of track reports that are known to reflect routine activity. Ideally the set of track reports would contain sufficient numbers of exemplars from all the contexts in which the system will be required to operate. However, this is not mandatory because the learning system can adapt at a later point in time, either autonomously or via operator input. Such operator interaction is not onerous using our approach since the required input information can be obtained from examples selected on-line either by selecting tracks or by confirming/rejecting alerts. Once a normalcy model is learned, new track reports are compared to the model representation. Those reports that fall outside of the bounds of normal activity can then be flagged as alerts that require human operator attention. Operators may guide learning by confirming or rejecting alerts raised by the system. Operators may also label events as threatening or innocuous to trigger learning in order to refine system performance. As activities or contexts change, learning proceeds in a semi-supervised fashion, benefiting from operator experience without intensive interaction. Moreover, the operator can adjust the number of false alarms by controlling the sensitivity level of system alerting, i.e., the degree to which vessel activity must differ from learned normalcy models before an alert is generated.

Figure 5 illustrates the normalcy model learned from vessel track data recorded in the Miami Harbor vicinity during August 2004. Note that the potential locations of vessels become less constrained as one pans from west to east (left to right) across the figure. In fact, in the east-most section of the region, the learned representation spans the location space. It should also be noted that the great majority of the learned boxes in the east-most area are uniformly pale green, an indication that the pattern of travel within this area is does not follow particular navigation routes. In contrast, the red arrow indicates a heavy traffic route in which normal activity is tightly constrained in both location and velocity.

7. DISPLAY

A map of the area of interest is displayed in a large panel on the left hand side (see Figure 6). There is a smaller zoomed-out map on the top right hand side that enables the operator to see a higher level view of the port, though this did not seem to be used by the operator in our preliminary testing. The watch-stander can select areas on either map display and can zoom in or out on any area. Certain new capabilities were provided including image snapshots of vessels captured at the time alerts are generated.
An operator has the ability to rewind and replay vessel tracks in fast and slow motion and a mouse click allows the operator to select an alert message and the corresponding vessel tracks are highlighted on the track display. An operator can view the alert history for vessels with past alert detections. One panel on the display shows a summary of the alerts that have been activated or deactivated. The display also shows camera placement and a geometrical depiction of the instantaneous field of view for each camera. Finally the operator has the ability to select display of one or all track sources on the maps and can increase or decrease the size of the track symbols, based on operator preference.

Figure 7 is an example of the alert panel displayed in SeeCoast. As new alerts are generated they appear at the top of the alert list. An operator can confirm or reject an alert, and view New Alerts or Acknowledged Alerts. The operator has the ability to sort alerts by date and time, by type or by alert message text. The operator can turn sound “On” or “Off” as well. Finally when more than one vessel is associated with an alert, thumbnails of both vessels are displayed.

The operator can utilize the interface to assist in training the system and the learned models are updated based on the selection of “confirm” or “reject” on a particular alert. The following describes the contribution to the learning model: If an alert is confirmed, this event (and other similar events) is labeled as anomalous and similar future events will generate Anomaly alerts. Otherwise, the event (and other similar events) is labeled as normal and similar future events will not generate Anomaly alerts. Alert history is displayed by selecting a Maritime Mobile Service Identity (MMSI) in the “MMSIs Involved” area within the Alert Pane.

8. JHOC DEPLOYMENT

The SeeCoast prototype system was deployed at the USCG / USN Joint Harbor Operations Center (JHOC) in Portsmouth Virginia in March, 2006. The JHOC uses a combination of medium and long range electro-optical and infrared video cameras, Automatic Identification System receivers, and maritime surface surveillance radars overlooking the lower Chesapeake Bay including the nearby waterways and open ocean area illustrated in Figure 8.

JHOC was chosen primarily because of its proximity to the USCG C2CEN (Command and Control Engineering Center) where Hawkeye port/coastal surveillance systems are developed, tested, and supported with system engineering and operator training.

9. INTEGRATION & TESTING

The implementation of the SeeCoast system involved development and integration of several core software components (video processor, track fusion engine, learning based analysis, rule based analysis, display as described above, as well as COMRIC (COMDAC – Command Display & Control – Radar Information Communication) interface manager, and
messenger communication service). Along with these core software components, a set of non-core helper components were also developed to provide capabilities such as field sensor data capture, internal component message log capture, and general component health evaluation and monitoring.

Initial integration efforts focused on definition and refinement of both external and internal component interfaces. External data sources, provided via Hawkeye, are the radar system tracker (which produces radar tracks, AIS tracks, and composite radar and AIS tracks via COMRIC), the camera manager (which produces camera metadata related to the current pan, tilt, and zoom of a particular camera’s field of view), and the SmartSite video encoder (which produces the video stream itself in a proprietary format). Verification of proper data extraction from these external data sources was the first milestone in the development effort. To fulfill its automated scene understanding role, SeeCoast makes more stringent accuracy and data extraction demands than would be required for human in the loop analysis. Among the many issues that had to be addressed were time synchronization across multiple heterogeneous sensors, accuracies and tolerances of the hardware itself, and accuracies of the software interface between the hardware and downstream systems.

In parallel with the external data stream interface analysis and design, internal component interfaces were analyzed and designed to ensure correct and efficient distribution of all required data elements to appropriate components. Transport between components was implemented either by point-to-point TCP/IP communication or by data exchange through a relational database. The utilization of a relational database as the inter-process communication mechanism between

Figure 6. SeeCoast display illustrating the map display, alert log window, and track detail panel.
internal components reaped multiple benefits over a purely TCP/IP based implementation; these include simple capture of data exchanged between different components, the ability to extract and preserve data produced by components under differing scenarios for later analysis, and the capability to compare data produced by components after defect repair to assess accuracy and verify absence of undesired changes to output results (i.e., regression testing). Data captured within the database include tracker output, alerts generated by both learning- and rule-based alerting, and normalcy models learned over the course of scenario execution.

With external and internal interface definitions available, a set of components that perform basic robust data exchange was developed. This was one of the most important milestones in the entire process because once the interfaces are implemented and verified, a pluggable framework exists, within which algorithmic analysis and assessment of component effectiveness can truly begin (using the entire system as an analysis workbench). Moreover, component interactions and general system effectiveness can also begin to be assessed.

Testing of the SeeCoast system was a very complex and difficult task due to the complex interactions between the external data sources and their data characteristics as well as the impact that this input data has upon the internal algorithmic implementation of the core software components. A layered test plan was established to provide a reasonable level of verification and validation for logical and manageable parts of the SeeCoast system. On-site testing was used to address system issues with live data for video and fusion testing. This was helpful to work through interactions between components.

To provide a set of consistent computing environments for development and testing, a Software Integration Lab (SIL) was assembled. The SIL environment operated under rules tailored to provide appropriate flexibility for the level of maturity being tested. At the lowest ‘unit test’ level, all components were required to have a mechanism enabling self-testing in a standalone mode. ‘Units’ would have to ingest reasonable data inputs, produce reasonable outputs, and not fail for a specified period of time. Success at the unit test level enabled a component to be subjected to an integration test where it would be integrated with both upstream and downstream components to assess the stability and accuracy of that components’ functionality within the context of other system components. The goal of integration testing was to verify that components could interact with each other in a stable and accurate manner over a specified period of time without system failure. It is at this level that a large number of analysis tasks took place to determine the impact of different algorithmic approaches upon system effectiveness as a whole. The integration test level was flexible enough to allow quick integration of individual component changes, enabling execution

Figure 7. SeeCoast alert log.

Figure 8. SeeCoast was deployed to JHOC in Portsmouth, Virginia overlooking the lower Chesapeake Bay.
of different types of impact analysis without having to create a formal system release. The third level of testing – ‘system testing’ – is the last level of internal testing. When a complete set of software components exhibits sufficient stability and functionality to comprise a baseline, these components are packaged together as a formal system release. In formal system tests, a complete set of versioned software components are installed in a controlled environment and subjected to a set of well defined tests outlined within the System Test Plan. System Test Reports are generated indicating the exact test cases executed as well as documenting any observations of the system responses to the defined test cases. Software defects are recorded within a defect tracking database to trigger a formally defined process for defect repair. Satisfactory system test performance results in that set of components being deemed a fieldable release. A ‘field test’ is the final level of testing wherein a fieldable system release is executed in the field on a platform deployed at a Coast Guard facility that is directly connected to the external data sources. This was the most challenging of the levels of testing to execute as typically the deployment environment had several differences that could not be replicated or anticipated within the SIL. Defects discovered during field tests are entered into the defect tracking database together with as much supporting environmental data as possible to aid in the replication of the problem back within the SIL and to be used to validate the defect repair.

10. LESSONS LEARNED & CONCLUSIONS

The prototype SeeCoast system is currently installed at the Joint Harbor Operations Center in Portsmouth Virginia where it is currently learning the scene, undergoing testing by the USCG, and being evaluated. The learning system continues to accumulate experience in refining models of normal activities for various classes of vessels in and around the lower Chesapeake Bay area. Current work is focused on new learning models and developing more advanced video processing algorithms to further improve the robustness of the system to local conditions and to reduce any remaining false alarms.

When deploying a new technology to an existing operational site, one can take an approach of discarding the old and replacing it with the new, in essence a revolutionary approach, or instead incrementally adding capabilities to the existing system, in essence an evolutionary approach. The latter was chosen in this case in order to assess the improvements in operational efficiency and effectiveness with the enhancements. We attempted to adapt the system to the operator rather than adapting the operator to the system. This allowed the USCG to use their experiences to find more efficient and effective ways accomplish their existing tasks.

Another significant lesson concerned the effects of noise or errors in the fusion of information at earlier stages upon the later stages. For instance, if detections were mis-associated during track fusion, then an alert pattern might be erroneously activated, or also the evidence for a pattern could be degraded to the point where an event of interest may go undetected. It was important to keep the performance of the complete system in mind while tuning the components. For example, large vessels would occasionally generate multiple detections from radar or video which would detect the bow and stern as separate vessels. It would be possible to compensate for this in the track association algorithm, but since this does not impede the ability to match against activity patterns of interest (say, penetration of a zone), it was not necessary to reduce the probability of a false alarm in this case and risk losing a positive detection. There are still issues to address, but the SeeCoast system has passed its acceptance test plan.

Capabilities such as developed in SeeCoast are necessary to provide watch-standers to perform their tasks effectively and efficiently. SeeCoast includes video processing, intelligent camera control, multi-sensor track fusion, rule-based alerting, learning of vessel activity normalcy models, and an alert interface.

ACKNOWLEDGEMENTS

The material in this paper is based upon work funded by DHS S&T/HSARPA under US Coast Guard Contract No. HSCG32-05-R00004.
REFERENCES


