SEAK Specification

Editors:
Nitin A. Gawande, Seunghwa Kang, Joseph B. Manzano
Nathan R. Tallent, Darren J. Kerbyson, Adolfy Hoisie

Pacific Northwest National Lab

May 2016

Contents

Introduction to SEAK

Many applications of high performance embedded computing are limited by performance or power bottlenecks. Consider a mobile imaging system that recognizes faces from an array of cameras. Because face recognition is a computationally intensive task, potential solutions may not fit within the mobile system's power and size envelope. Suppose the system's designer desires the most power efficient face recognition *solution* that satisfies a given real-time constraint. That is, the solution may use *any* algorithm on *any* architecture that meets the given correctness, time, and power constraints. To solicit the best solutions, how should the designer capture the key input and output requirements without biasing toward specific algorithms or architectures?

The SEAK benchmark suite generalizes this question. The benchmark suite is a collection of *constraining problems* for common embedded computing challenges. A constraining problem is a goal-oriented problem specification that separate problem-domain constraints from solution implementations so as to encourage creative solutions that meet goals but that may deviate from standard implementations. Further, a constraining problem is defined so as to facilitate rigorous, objective, end-user evaluation for their solutions.

To avoid biasing solutions toward existing algorithms, SEAK constraining problems use a *mission-centric* (abstracted from a particular algorithm) and goal-oriented (functional) specification. To encourage solutions that are any combination of software or hardware, we use an end-user black-box evaluation that can capture tradeoffs between performance, power, accuracy, size, and weight. The tradeoffs are especially informative for procurement decisions. We call our benchmarks *future proof* because each mission-centric interface and evaluation remains useful despite shifting algorithmic preferences.

It is challenging to create both concise and precise goal-oriented specifications for mission-centric problems. An IPDPS '16 paper [107] describes the SEAK benchmark suite and discusses these challenges.

A constraining problem consists of a specification document and a source code distribution. The specification document (a) justifies each problem, (b) describes input and output requirements, and (c) details evaluation criteria for correctness, performance, and power. The source code repository contains input generators and correctness checkers. Although the specification does not require reference implementations, when available we include representative solutions to use as reference implementations.

A SEAK evaluation ranks results according to criteria such as a solution's correctness, power, cost, size, and weight. The evaluation results establish a parial ordering among solutions. An *inefficient* solution is any solution where one performance metric can improve without degrading another. An *efficient* solution is one where no improvement is possible without degrading another metric. Efficient solutions form an *efficiency frontier*. One can use the efficiency frontier to identify the best solutions given a set of constraints.

Constraining Problem 1

Acoustics: Automatic Speech Recognition

Seunghwa Kang (PNNL), Joseph B. Manzano (PNNL), Nathan R. Tallent (PNNL)

1.1 Justification

This constraining problem asks participating solutions to perform automatic speech recognition: transcribing spoken texts using a computer system. Early speech recognition research had focused on recognizing an isolated word or a short phrase from a small set of known words (small vocabulary) assuming that words or phrases are spoken by one or few speakers with known vocal characteristics (speaker dependent speech recognition) under low background noise [4, 40]. The advances over the past few decades have shifted the focus to recognizing long continuous speech (*e.g.* broadcast news transcription) or spontaneous speech (common in conversational speech) composed of a large or even unlimited number of words (large or unlimited vocabulary) uttered by arbitrary speakers (speaker independent speech recognition) in noisy acoustic environments [4, 40, 100]. The speech recognition performance has significantly improved in the past several years largely owing to the adoption of deep neural networks [48].

This constraining problem asks participants to develop a speaker independent large vocabulary continuous speech recognition system for English under various noisy conditions. The Defense Advanced Research Projects Agency (DARPA) has invested on speech recognition for decades over multiple projects such as the 1000 word resource management database project [93], Effective Affordable Reusable Speech-to-Text (EARS) [23], Global Autonomous Language Exploitation (GALE, focusing on transcribing foreign languages such as Mandarin Chinese and Arabic) [24, 105], and Robust Automatic Transcription of Speech (RATS, focusing on language and speaker identification and keyword spotting under adverse acoustic environments) [31]. Speech recognition techniques can be utilized in multiple military applications; for example, low bit rate communication, voice control of military devices, military intelligence systems, and language translation. Low bit rate communication of speech is necessary to provide more channels, enhance security, and improve robustness [119]; speech recognition technique is used to transform speech signals to intermediate codes (*e.g.* phonetic units) which require less bandwidth to transmit [119]. Voice control of computers, weapons, and sensor systems improves the productivity of military officers but this application requires high speech recognition performance under adverse conditions (such as moving military vehicles in noisy environments) [119, 88, 117]. Speech recognition is also necessary in searching, sorting, and filtering a large volume of speech data (*e.g.* a large volume of broadcast news) [119, 88]. Speech

recognition is the first step in language translation between military personnel in multinational forces as well [88].

This constraining problem adopts two existing databases as proxy data to represent speech recognition challenges in military applications. Databases are selected based on their relevance to military application challenges and public availability.

The CMU PDA database [82, 81] contains transcripts and single-channel and multi-channel recorded speech audio data. Participating speakers read transcripts displayed on a hand-held PDA in varying noisy environments. For the single-channel data, 11 speakers participated in the recording. Each speaker read about 140 sentences under noisy environments. The first 40-43 sentences are selected from the Wall Street Journal 5K test database and the remaining sentences are selected from the Wall Street Journal training database [85]. The built-in microphone on a PDA is used for recording speech in noisy environments. An additional close-talk microphone is used to record speech with higher signal-to-noise ratio (SNR). For the multi-channel data, 16 speakers participated in the recording. The first 8 speakers read transcripts displayed on a hand-held PDA under relatively low noise conditions. The remaining 8 speakers read transcripts under noisier environments. Each speaker read approximately 50 sentences. The first 40-43 sentences are from the Wall Street Journal 5K test database and the remaining sentences are numbers having multiple digits. Four microphones attached around a PDA are used for recording speech in noisy environments, and an additional close-talk microphone is used to obtain clearer speech data similar to the single-channel case. The single-channel dataset provides 11 KHz speech audio data. For the multi-channel dataset, 11 KHz and 16 KHz speech data are provided. This database is challenging due to varying noisy environments and continuously changing positions of a speaker and microphones on a hand-held PDA. Speech recognition systems are expected to exploit speech data from multiple channels to overcome the challenges. We adopt this database due to its availability to the public and the relevance of the challenges to various military applications—*e.g.* speech recognition in military vehicles operating in noisy conditions.

The 3rd CHiME Speech Separation and Recognition Challenge [9] provides a database of recorded speech utterances in challenging noisy environments. Participants read transcripts from the Wall Street Journal corpus displayed on a tablet in multiple acoustic environments: sound proof booth, bus, cafe, pedestrian area, and street junction. Speech utterances were recorded using six microphones attached on a tablet and a separate close-talk microphone. The database provides both real data (speech utterances recorded in noisy environments) and simulated data (speech utterances generated by mixing clean speech data with separately recorded noise data). The audio data were initially recorded in 24 bits at 48 KHz and downsampled to 16bit 16 KHz for distribution. The database is composed of training, development, and final test datasets. The training dataset includes 1600 real utterances (from four speakers) and 7138 simulated utterances (from 83 speakers). The simulated utterances are generated by mixing the Wall Street Journal data (clean speech) with background noise. The development dataset

has 410 real utterances per noisy environment and 410 simulated utterances per noisy environment for the four different environments (bus, cafe, pedestrian area, and street junction). The development dataset data are uttered by four new speakers. The final datatest set includes 310 real utterances and 310 simulated utterances per noisy environment. Four new speakers participated in generating the test dataset. Speech utterances recorded in sound proof booth are mixed with background noise to generate simulated data for the development and test datasets. The test dataset was disclosed later (June 3rd, 2015) than the training and development datasets close to the final challenge submission deadline (July 10th, 2015); the test dataset will be used for final evaluation of the submissions. This database is challenging due to noise coming from different environments similar to the PDA database. This database is larger and newer than the PDA database.

1.2 Description

1.2.1 Input

The CMU PDA database is available to download from the following url: `http://www.speech.cs.cmu.edu/databases/pda`. The compressed *PDA.tar.gz* file includes the following four directories.

- *doc*: This directory includes documents and images describing the database.
- *lists*: This directory includes transcripts and other auxiliary files related to speech utterances in the database.
- *PDAs*: This directory includes the 11 KHz single-channel speech audio data from 11 speakers (one sub-directory per speaker).
- *PDAm*: This directory includes the 11 KHz and 16 KHz multi-channel speech audio data from 16 speakers (one sub-directory per speaker).

Refer to the CMU PDA database for further details.

This constraining problem uses the 16 KHz multi-channel speech audio data for testing. Participating systems can use data from all four channels but cannot use speech audio data from the close-talk microphone. Speech recognition needs to be performed using speech audio data only; speaker id, utterance number, and channel number in the file name cannot be used for speech recognition. Participating systems can be trained using any data (*e.g.* the Wall Street Journal training data) excluding the test data (both 11 KHz and 16 KHz data cannot be used for training) and their postprocessed derivatives.

Accessing the 3rd CHiME challenge database requires registering at the following url: `http://spandh.dcs.shef.ac.uk/chime_challenge/data.html`. Registrants will receive an email providing a link to the download page. The download page has multiple datasets and baseline software tools. The following explains several relevant items in the database.

- *CHiME3_isolated_tr05_real*: This is a training dataset having real data recorded in soundproof booth and noisy environments. Each utterance is isolated from the entire recording and saved in a separate file.
- *CHiME3_isolated_dt05_real*: This is a development dataset having real data recorded in soundproof booth and noisy environments. Each utterance is isolated from the entire recording and saved in a separate file.
- *CHiME3_isolated_et05_real*: This is a test dataset having real data recorded in soundproof booth and noisy environments. Each utterance is isolated from the entire recording and saved in a separate file.
- *CHiME3_isolated_tr05_simu*: This is a training dataset having simulated data generated by mixing clean speech data with background noise. Each utterance is isolated from the entire recording and saved in a separate file. Downloading this dataset requires a license to access the Wall Street Journal data.
- *CHiME3_isolated_dt05_simu*: This is a development dataset having simulated data generated by mixing clean speech data with background noise. Each utterance is isolated from the entire recording and saved in a separate file.
- *CHiME3_isolated_et05_simu*: This is a test dataset having simulated data generated by mixing clean speech data with background noise. Each utterance is isolated from the entire recording and saved in a separate file.
- *CHiME3_backgrounds*: This has background noise data. New simulated speech data with varying signal-to-noise ratio (SNR) can be generated by mixing speech data recorded in soundproof booth with background noise data.

Refer to the CHiME homepage for additional details.

This constraining problem tests participating systems using the test dataset having real data (*CHiME3_isolated_et05_real*). Participating systems can use data from all six channels but cannot use speech audio data from the close-talk microphone. Speech recognition needs to be performed using speech audio data only; speaker id, utterance number, location id, and channel number in the file name cannot be used for speech recognition. Participating systems can be trained using any data excluding the test data (*CHiME3_isolated_et05_real*) and their postprocessed derivatives.

1.2.2 Output

A trained speech recognition system should output one transcript per each utterance in the test datasets. One may separately train a speech recognition system for the PDA test dataset and the CHiME test dataset, but only a single trained speech recognition system with a single set of parameters should be used for each test dataset.

1.3 Evaluation

1.3.1 Correctness

Word error rate (WER) [4, 40] is used to measure speech recognition accuracy. The word error rate is computed by dividing the number of word substitutions, deletions, and insertions by the total number of words in the reference transcript; $WER = \frac{S+D+I}{N}$, where S is the number of substitutions, D is the number of deletions, I is the number of insertions, and N is the total number of words in the reference transcript.

1.3.2 Performance and Power

Execution time is separately measured for training and testing. Power consumption is separately measured for training and testing as well. The overall performance of a speech recognition system will be evaluated based on the word error rate (lower is better), execution time for training, power consumption for training, execution time for testing, and power consumption for testing. Different weights will be assigned to different metrics based on varying requirements in military sites; in general, execution time and power consumption for training are less important as training can be performed off-site using cluster computers while testing (and actual speech recognition) will more likely to be performed using embedded systems in military sites.

Constraining Problem 2

Radio: Synthetic Aperture Radar Image Formation

Nitin A. Gawande (PNNL), Nathan R. Tallent (PNNL), Joseph B. Manzano (PNNL)

2.1 Justification

Simple Radar system has the potential to achieve fine range resolution, but it is constrained to relatively poor azimuthal resolution [18]. The synthetic aperture Radar (SAR) imaging technique can achieve high azimuthal resolution as well [18]. SAR synthesizes a large aperture using Radar platform motion and then forms an image using data corresponding to the pulses acquired over the large synthetic aperture. SAR technique has wide applications in imaging through aerial borne platforms and satellites.

The purpose of SAR is to make repeated observations of a scene of interest over a large range of angles of observation. These observations are then post-processed to resolve an image of the scene whose resolution is limited by the angular range of the observations rather than by the size of the physical receiver. The amount of data needed for image formation is a function of the radar system parameters, desired image resolution, and coverage area.

SAR radars require a moving transmitter, and long range moving target identification must be done at high elevations that are not achievable from a ground station. The raw data rates for radars are much higher than the fastest available military communications link can support, requiring processing and data reduction on the mobile platform when the latency required is less than the mission endurance.

There are many algorithms to implement SAR image formation. These algorithms are broadly classified in two categories: (1) frequency domain methods, and (2) time domain methods. One of the most flexible and general approach currently in use is the backprojection approach which falls in the category of time domain methods. Novel algorithms that offer high performance with improved image quality have evlolved over time. However, high resolution SAR imaging in real-time require significant computing resources. In addition, many platforms of interest for radar are limited in size, weight, and ability to dissipate energy. As a result, many SAR missions are constrained by arithmetic throughput, internal data movement throughput, and total sensor input data rates. We therefore define image formation as one of the most constraining problems in the SAR domain.

Table 2.1: Input parameters for SAR image formation.

Parameter	**Variable (Dimensions)**
Carrier frequency	f_c
Bandwidth, transmitted signal Baseband bandwidth is $2 \times f_0$	f_0
Pulse duration	$\mathrm{T_p}$
Range distance to center of target area	$\mathrm{X_c}$
Cross-range distance to center of target area	$\mathrm{Y_c}$
Target area (range $\times$ cross-range)	$(2\mathrm{X_c} \times 2\mathrm{Y_c})$
Minimum half length synthetic aperture	$L_{\min}$
Output image (complex valued)	I $(N_x \times N_y)$

2.2 Description

Although SAR can utilize a three-dimensional radar data cube, we consider the two-dimensional case. In the two-dimensional case there is one platform sensor (channel) with an aperture of length $2L$, $\{-L \text{ to } +L\}$ in the coordinate system used in the input generator. This results in a Radar signal data matrix of N_p radar pulses and K range bins. A pulse is one transmission of radar energy; given the duration of a pulse and the relative speeds of pulse propagation and sensor movement, the return pulse relates to both time and location. The range bins correspond to K complex-valued samples per pulse. Further description of input and output for this problem is described in following sub-sections.

2.2.1 Input

The input parameters for SAR image formation problem are given in Table 2.1. In this problem we make use of two-dimensional coordinates. However, the complexity of the image formation algorithm remains unchanged from that of a three-dimensional image scene. We use template images of targets to construct Radar signal response. The input parameters including the template images are different for a small, medium, and large size problem. The computational load of image formation increases with the size of the problem. We provide input generator and reference implementations for stripmap and spotlight modes of SAR.

2.2.2 Output

2.2.3 Signal Model

A signal model for monostatic SAR is described here. A more detailed description of signal model is described else where [97], [44]. The sensor platform moves nomally in the Y-direction which is perpendicular to the range, X-direction. The

antenna (a) phase center has a three-dimensional spatial location denoted by $\mathbf{r}_a(\tau) = (x_a(\tau), y_a(\tau), z_a(\tau))$. Where, τ denotes the synthetic aperture, or slow time, domain. A target is specified at a location $\mathbf{r}(\tau) = (x(\tau), y(\tau), z(\tau))$. For a stationary target, the dependence of $\mathbf{r}$ on τ can be dropped. The distance of the target from the antenna phase center, $d_{a_0}(\tau)$ can be computed by equation 2.1.

$$d_{a_0}(\tau) = \sqrt{(x_a(\tau))^2 + (y_a(\tau))^2 + (z_a(\tau))^2} \tag{2.1}$$

The antenna periodically transmits pulses of energy in the direction of the target that reflects off scatterers in the scene. Some of the energy of the transmitted pulses is received by the radar. There are N_p number of pulse sampled in a given synthetic aperture. The sequence of transmission time of each pulse is given by $\{\tau_n = 1, 2, \ldots N_p\}$. There are K frequency samples, per pulse. The sequence of frequency values is given by $\{f_k = 1, 2, \ldots K\}$. The complex signal from the target is given by Equation 2.2.

$$S(f_k, \tau_n) = A(f_k, \tau_n) exp\left(\frac{-j4\pi f_k \Delta R(\tau_n)}{c}\right) \tag{2.2}$$

The amplitube, $A(f_k, \tau_n)$, is related to the radar cross section of the target. The phase is dependent on the frequency of each sample and on the differential range, $\Delta R(\tau_n)$ which is given by Equation 2.3.

$$\Delta R(\tau_n) = d_{a_0}(\tau) - d_a(\tau) \tag{2.3}$$

The frequency samples, $\{f_k\}$ have a median value denoted by f_c and a step size denoted by Δf. The free range extent of SAR image, W_x is given by Equation 2.4.

$$W_x = \frac{c}{2\Delta f} \tag{2.4}$$

Considering a total bandwidth of the received pulse equal to $2f_0$, the range resolution is given by Equation 2.5.

$$\Delta x = \frac{c}{4f_0} = \frac{c}{2(K-1)\Delta f} \tag{2.5}$$

The cross-range extent of SAR image, W_y can be computed based on the azimuth step size $\Delta\theta$, and the minimum wavelength $\lambda_m in$ as per Equation 2.6.

$$W_y = \frac{\lambda_{min}}{2\Delta\theta} \tag{2.6}$$

The total azimuth angle traversed in the synthetic aperture is $2\theta_a$. The center wavelength is λ_c such that $\lambda_c = c/f_c$, the cross-range resolution, Δy is given by Equation 2.7.

$$\Delta y = \frac{\lambda_c}{4\theta_a} = \frac{\lambda_c}{2(N_p - 1)\Delta\theta} \tag{2.7}$$

2.2.4 Matched Filter Algorithm

Matched filter is a time-reversed complex-conjugate form of the received signal on itself. Considering an isotropic point scatterer which will have a constant amplitude. Therefore $A(f_k, \tau_n)$ can be considered to have a constant value. The matched-filter denoted by $I(r)$, at location r is given by Equation 2.8.

$$I(r) = \frac{1}{N_p K} \sum_{n=1}^{N_p} \sum_{k=1}^{K} S(f_k, \tau_n) exp\left(\frac{+j4\pi f_k \Delta R(\tau_n)}{c}\right) \tag{2.8}$$

Equation 2.8 is applied for each pixel to form a reconstructed SAR image. Application of Equation 2.8 requires computation of differential range, $\Delta R(\tau_n)$ for all pixels for every pulse. In order to reconstruct a two dimensional SAR image of size $N \times N$, the matched filer algorithm has a computational complexity of $O(N^4)$.

2.2.5 Backprojection Algorithm

The matched filter response as given by Equation 2.8 can be used to compute the target response at a discrete range bin, m. Equation 2.8 forms the basis for the derivation of the backprojection algorithm. The backprojection algorithm is described in many papers and textbooks [44, 78, 33].

The range profile at range bin m for a pulse received at slow time τ_n is given by Equation 2.9.

$$s(m, \tau_n) = \sum_{k=1}^{K} S(f_k, \tau_n) exp\left(\frac{+j4\pi f_k \Delta R(\tau_n)}{c}\right) \tag{2.9}$$

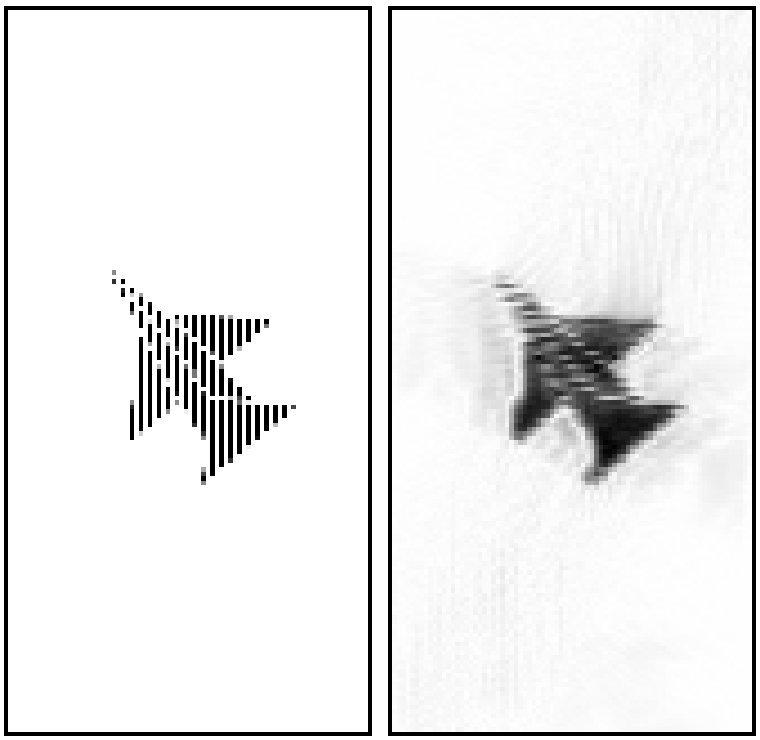
In order to compute the contribution to a pixel at location r for a given pulse n, $\Delta R(\tau_n)$ is calculated and used to compute interpolated value of $s(m, \tau_n)$. This interpolated value $s_{inter}(r, \tau_n)$ is then obtained for every pulse and summed up to reconstruct an image $I(r)$ as given by Equation 2.10.

$$I(r) = \sum_{n=1}^{N_p} s_{inter}(r, \tau_n) \tag{2.10}$$

The output of backprojection is a complex-valued image of dimension $N_x \times N_y$. To form the output image, a contribution from each pulse is integrated into each pixel. Thus, the computational complexity for backprojection is $\mathcal{O}(N_p \cdot N_x \cdot N_y)$. For the nominal case where the number of pulses and pixels-per-side of the formed image are comparable (e.g., N), the computational complexity of backprojection is $\mathcal{O}(N^3)$. To date several variants of backprojection algorithm have been implemented with differences in computational complexity and image quality.

Here, we briefly described two implementations of SAR image formation, however any novel implementation can be used to reconstructed the SAR output image. Figure 2.1 shows the input targets in SAR imaging scene and backprojection reconstruction for spotlight mode. Figure 2.1a is used as an input target

imaging scene. Using the parameters as per input specification, the signal from the input image is constructed. With the application of suitable SAR image formation algorithm, the target signal is used to reconstruct image of the scene. Figure 2.1b shows the reconstruction of the scene using the backprojection algorithm. Here, Figures 2.1a and 2.1b are images that are cropped out of the entire SAR imaging scene.



(a) (b)

Figure 2.1: SAR input and output images: (a) Input targets (b) Backprojection reconstruction.

2.3 Evaluation

2.3.1 Correctness

Correctness can be measured by comparison of reconstructed SAR output image/s to that with the input images that were used to construct the target signal. An exact match to the input image is not anticipated, so a peak signal to noise ratio (PSNR) will be computed between the input image and the SAR reconstructed

image. The PSNR values for the output images will be used as a metric to evaluate any novel implementation of image formation algorithm.

2.3.2 Performance and Power

The unit of work for relative comparisons is a pixel in the output image. There are $N_x \times N_y$ units of work.

Performance is evaluated with time per units of work.

Power is evaluated with Watts per unit of work.

Constraining Problem 3

Radio: Synthetic Aperture Radar Target Detection

Nitin A. Gawande (PNNL), Nathan R. Tallent (PNNL), Joseph B. Manzano (PNNL)

3.1 Justification

Simple Radar system has the potential to achieve fine range resolution, but it is constrained to relatively poor azimuthal resolution [18]. The synthetic aperture Radar (SAR) imaging technique can achieve high azimuthal resolution as well [18]. SAR synthesizes a large aperture using Radar platform motion and then forms an image using data corresponding to the pulses acquired over the large synthetic aperture. SAR technique has wide applications in imaging through aerial borne platforms and satellites.

The purpose of SAR is to make repeated observations of a scene of interest over a large range of angles of observation. These observations are then post-processed to resolve an image of the scene whose resolution is limited by the angular range of the observations rather than by the size of the physical receiver. The amount of data needed for image formation is a function of the radar system parameters, desired image resolution, and coverage area.

SAR radars require a moving transmitter, and long range moving target identification must be done at high elevations that are not achievable from a ground station. The raw data rates for radars are much higher than the fastest available military communications link can support, requiring processing and data reduction on the mobile platform when the latency required is less than the mission endurance.

There are many algorithms to implement SAR image formation. These algorithms are broadly classified in two categories: (1) frequency domain methods, and (2) time domain methods. One of the most flexible and general approach currently in use is the backprojection approach which falls in the category of time domain methods. Novel algorithms that offer high performance with improved image quality have evlolved over time. However, high resolution SAR imaging in real-time require significant computing resources. In addition, many platforms of interest for radar are limited in size, weight, and ability to dissipate energy. As a result, many SAR missions are constrained by arithmetic throughput, internal data movement throughput, and total sensor input data rates. The SAR target-detection problem differs from the SAR image-formation problem such that the targets in the imaging scene need to be detected for the success of the mission problem. Therefore, targets located in noisy background or appear camouflaged in imaging scene will require rigourous processing or a novel algorithm use. We therefore define target detection as one of the most constraining problems in the

Table 3.1: Input parameters for SAR image formation.

Parameter	**Variable**	**(Dimensions)**
Carrier frequency	f_c	
Bandwidth, transmitted signal Baseband bandwidth is $2 \times f_0$	f_0	
Pulse duration	$\mathrm{T_p}$	
Range distance to center of target area	$\mathrm{X_c}$	
Cross-range distance to center of target area	$\mathrm{Y_c}$	
Target area (range $\times$ cross-range)		$(2\mathrm{X_c} \times 2\mathrm{Y_c})$
Minimum half length synthetic aperture	$L_{\min}$	
Output image (complex valued)	I	$(N_x \times N_y)$

SAR domain.

3.2 Description

Although SAR can utilize a three-dimensional radar data cube, we consider the two-dimensional case. In the two-dimensional case there is one platform sensor (channel) with an aperture of length $2L$, $\{-L$ to $+L\}$ in the coordinate system used in the input generator. This results in a Radar signal data matrix of N_p radar pulses and K range bins. A pulse is one transmission of radar energy; given the duration of a pulse and the relative speeds of pulse propagation and sensor movement, the return pulse relates to both time and location. The range bins correspond to K complex-valued samples per pulse. Further description of input and output for this problem is described in following sub-sections.

3.2.1 Input

The input parameters for SAR image formation problem are given in Table 3.1. In this problem we make use of two-dimensional coordinates. However, the complexity of the image formation algorithm remains unchanged from that of a three-dimensional image scene. We use template images of targets to construct Radar signal response. The input parameters including the template images with targets injected in the scene. The inputs are different for a small, medium, and large size problem. The computational load of image formation increases with the size of the problem. We provide input generator and reference implementations for stripmap and spotlight modes of SAR.

3.2.2 Output

3.2.3 Signal Model

A signal model for monostatic SAR is described here. A more detailed description of signal model is described else where [97], [44]. The sensor platform moves nomally in the Y-direction which is perpendicular to the range, X-direction. The antenna (a) phase center has a three-dimensional spatial location denoted by $\mathbf{r}_a(\tau) = (x_a(\tau), y_a(\tau), z_a(\tau))$. Where, τ denotes the synthetic aperture, or slow time, domain. A target is specified at a location $\mathbf{r}(\tau) = (x(\tau), y(\tau), z(\tau))$. For a stationary target, the dependence of $\mathbf{r}$ on τ can be dropped. The distance of the target from the antenna phase center, $d_{a_0}(\tau)$ can be computed by equation 3.1.

$$d_{a_0}(\tau) = \sqrt{(x_a(\tau))^2 + (y_a(\tau))^2 + (z_a(\tau))^2} \tag{3.1}$$

The antenna periodically transmits pulses of energy in the direction of the target that reflects off scatterers in the scene. Some of the energy of the transmitted pulses is received by the radar. There are N_p number of pulse sampled in a given synthetic aperture. The sequence of transmission time of each pulse is given by $\{\tau_n = 1, 2, \ldots N_p\}$. There are K frequency samples, per pulse. The sequence of frequency values is given by $\{f_k = 1, 2, \ldots K\}$. The complex signal from the target is given by Equation 3.2.

$$S(f_k, \tau_n) = A(f_k, \tau_n) exp\left(\frac{-j4\pi f_k \Delta R(\tau_n)}{c}\right) \tag{3.2}$$

The amplitube, $A(f_k, \tau_n)$, is related to the radar cross section of the target. The phase is dependent on the frequency of each sample and on the differential range, $\Delta R(\tau_n)$ which is given by Equation 3.3.

$$\Delta R(\tau_n) = d_{a_0}(\tau) - d_a(\tau) \tag{3.3}$$

The frequency samples, $\{f_k\}$ have a median value denoted by f_c and a step size denoted by Δf. The free range extent of SAR image, W_x is given by Equation 3.4.

$$W_x = \frac{c}{2\Delta f} \tag{3.4}$$

Considering a total bandwidth of the received pulse equal to $2f_0$, the range resolution is given by Equation 3.5.

$$\Delta x = \frac{c}{4f_0} = \frac{c}{2(K-1)\Delta f} \tag{3.5}$$

The cross-range extent of SAR image, W_y can be computed based on the azimuth step size $\Delta\theta$, and the minimum wavelength $\lambda_m in$ as per Equation 3.6.

$$W_y = \frac{\lambda_{min}}{2\Delta\theta} \tag{3.6}$$

The total azimuth angle traversed in the synthetic aperture is $2\theta_a$. The center wavelength is λ_c such that $\lambda_c = c/f_c$, the cross-range resolution, Δy is given by Equation 3.7.

$$\Delta y = \frac{\lambda_c}{4\theta_a} = \frac{\lambda_c}{2(N_p - 1)\Delta\theta} \tag{3.7}$$

3.2.4 Matched Filter Algorithm

Matched filter is a time-reversed complex-conjugate form of the received signal on itself. Considering an isotropic point scatterer which will have a constant amplitude. Therefore $A(f_k, \tau_n)$ can be considered to have a constant value. The matched-filter denoted by $I(r)$, at location r is given by Equation 3.8.

$$I(r) = \frac{1}{N_p K} \sum_{n=1}^{N_p} \sum_{k=1}^{K} S(f_k, \tau_n) exp\left(\frac{+j4\pi f_k \Delta R(\tau_n)}{c}\right) \tag{3.8}$$

Equation 3.8 is applied for each pixel to form a reconstructed SAR image. Application of Equation 3.8 requires computation of differential range, $\Delta R(\tau_n)$ for all pixels for every pulse. In order to reconstruct a two dimensional SAR image of size $N \times N$, the matched filer algorithm has a computational complexity of $O(N^4)$.

3.2.5 Backprojection Algorithm

The matched filter response as given by Equation 3.8 can be used to compute the target response at a discrete range bin, m. Equation 3.8 forms the basis for the derivation of the backprojection algorithm. The backprojection algorithm is described in many papers and textbooks [44, 78, 33].

The range profile at range bin m for a pulse received at slow time τ_n is given by Equation 3.9.

$$s(m, \tau_n) = \sum_{k=1}^{K} S(f_k, \tau_n) exp\left(\frac{+j4\pi f_k \Delta R(\tau_n)}{c}\right) \tag{3.9}$$

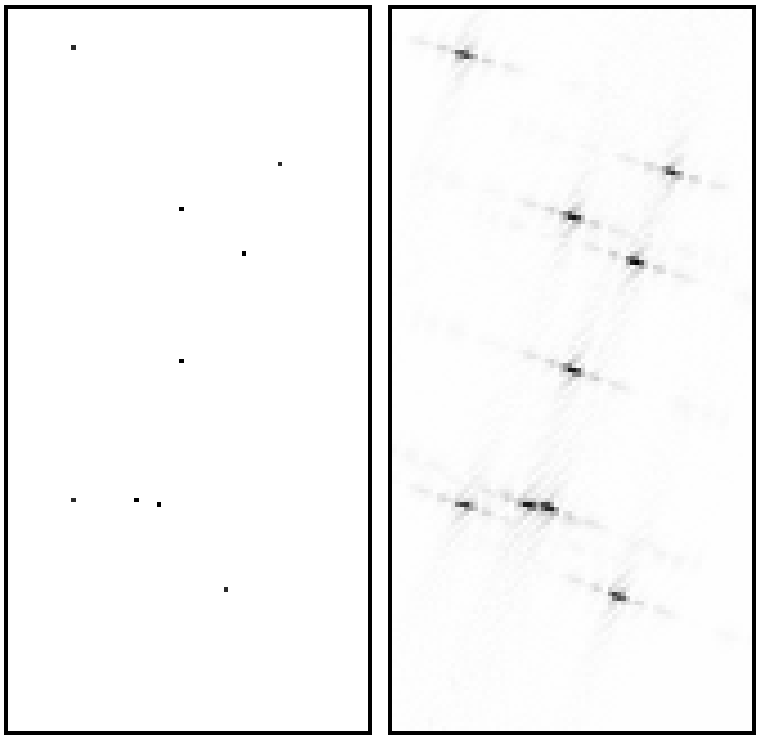
In order to compute the contribution to a pixel at location r for a given pulse n, $\Delta R(\tau_n)$ is calculated and used to compute interpolated value of $s(m, \tau_n)$. This interpolated value $s_{inter}(r, \tau_n)$ is then obtained for every pulse and summed up to reconstruct an image $I(r)$ as given by Equation 3.10.

$$I(r) = \sum_{n=1}^{N_p} s_{inter}(r, \tau_n) \tag{3.10}$$

The output of backprojection is a complex-valued image of dimension $N_x \times N_y$. To form the output image, a contribution from each pulse is integrated into each pixel. Thus, the computational complexity for backprojection is $\mathcal{O}(N_p \cdot N_x \cdot N_y)$. For the nominal case where the number of pulses and pixels-per-side of the formed

image are comparable (e.g., N), the computational complexity of backprojection is $\mathcal{O}(N^3)$. To date several variants of backprojection algorithm have been implemented with differences in computational complexity and image quality.

Here, we briefly described two implementations of SAR image formation, however any novel implementation can be used to reconstructed the SAR output image. Figure 3.1 shows the input targets in SAR imaging scene and backprojection reconstruction for spotlight mode. Using the input generator for this problem targets are injected in the range and cross-range after as shown in Figure 3.1a. Using the parameters as per input specification, the signal from the targets is constructed. With the application of suitable SAR image formation algorithm, the target signal is used to reconstruct image of the scene. Figure 3.1b shows the reconstruction of the scene using the backprojection algorithm. Here, Figures 3.1a and 3.1b are images that are cropped out of the entire SAR imaging scene.



(a) (b)

Figure 3.1: SAR input and output images: (a) Input targets (b) Backprojection reconstruction.

3.3 Evaluation

3.3.1 Correctness

Correctness can be measured by comparison of reconstructed SAR output image/s to that with the input images that were used to construct the target signal. It is essential for the reconstructed image to detect all the targets that were injected as per input and that lie inside the boundary of padding regions of range and cross-range. In addition, a peak signal to noise ratio (PSNR) will be computed between the input image and the SAR reconstructed image. The PSNR values for the output images will be used as a metric to evaluate any novel implementation of image formation algorithm.

3.3.2 Performance and Power

The unit of work for relative comparisons is a pixel in the output image. There are $N_x \times N_y$ units of work.

Performance is evaluated with time per units of work.

Power is evaluated with Watts per unit of work.

Constraining Problem 4

Radio: Space-Time Adaptive Processing Signal Formation

Nitin A. Gawande (PNNL), Joseph B. Manzano (PNNL), Nathan R. Tallent (PNNL)

4.1 Justification

Compared to using a single sensor, signal detection using an array of sensors leads to improvement in the Signal-to-Noise-Ratio (SNR) and has offered significant benefits in applications such as radar, sonar, satellite communications, and seismic systems [96]. Electromagnetic wave signals recorded by an array of antenna receivers as voltage signals over spatial and temporal domain constitutes the radar receiver signal data. Detecting a desired target-signal against background clutter and interference is an important challenge because the spectral characteristics of the interference and the target amplitude are often unknown. The background clutter has a greater impact on target detection for moving platforms such as advanced airborne surveillance radar systems [34]. Space-Time Adaptive Processing (STAP) algorithms employ spatial and temporal adaptive processing to suppress the interference in radar signal data and thus find targets which otherwise may not be detected. The STAP procedure achieves this by computing suitable weights that are applied to the radar signal to obtain desired target signal. These weights are said to be computed 'adaptively' due to the fact they reflect the actual clutter and interference in the environment.

The input to STAP is a three-dimensional radar data cube consisting of L channels, or the phase centers on radar that are displaced along the velocity vector of the platform; P pulses; and N samples per pulse. A pulse is one transmission of radar energy. The N samples per pulse are commonly termed range cells or range bins because they sample at range (or time) intervals. Because of the time scales involved, the range dimension is referred to as fast-time and the pulse dimension is referred to as slow-time. The L channels and P pulses add spatial and temporal diversity, respectively, to the acquired data; these dimensions correspond to the 'space' and 'time' in space-time adaptive processing. As a first step of a typical STAP algorithm like the extended Doppler-factored algorithm (EFA) [34], the pulse (time) domain is converted to Doppler spectrum domain. This is achieved by applying a discrete Fourier transform (DFT) to the P pulses for each range cell and channel pair resulting in K Doppler bins. We assume that DFT is already applied to the data set and we have a data cube structure as shown in Figure 4.1. A 2-D space-time snapshot corresponding to a particular range cell is also shown in Figure 4.1. Optimum adaptive weights are desired such that their application to the received signal vector enhances the signal to noise ratio.

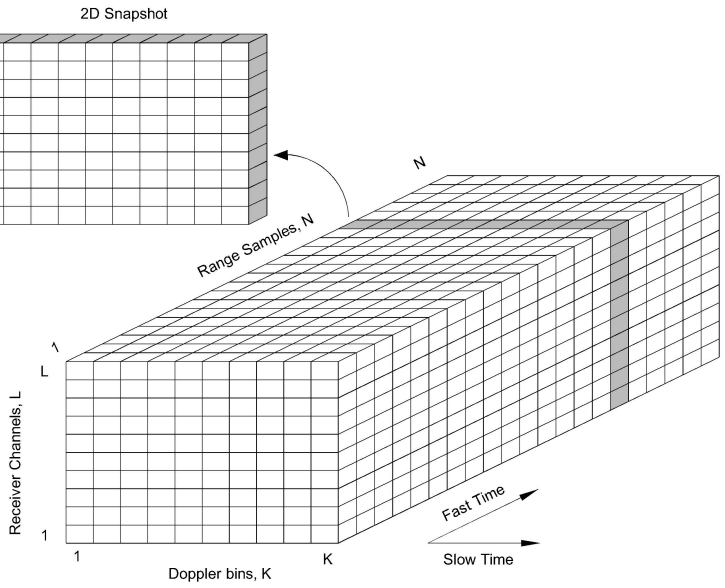


Figure 4.1: Data cube from a single coherent processing Interval (CPI)

The several STAP algorithms approach the problem in a different ways and have different levels of computational complexities. The key output requirement is an estimate of weights that obtains an output signal with desired characteristics. The real-time application of STAP brings performance constraints. Airborne platforms have power constraints. We therefore describe weights estimation as one of the most constraining problems in the STAP procedure.

Statistical STAP techniques use the interference covariance matrix to obtain adaptive weights [114]. Training data samples from range bins are used to compute the covariance matrix. These training data samples are chosen such that they are representative of the stationary clutter present in the cell under test. The adaptive weights are computed for a particular range cell and Doppler bin using statistics of the space-time snapshot vector consisting of data samples from the L array channels and T_{dof} adjacent Doppler bins. Here, T_{dof} is the temporal degree-of-freedom in a given STAP procedure. This requires composing $LT_{\text{dof}} \times LT_{\text{dof}}$-sized interference covariance matrices and applying subsequent operations to these matrices to get weights. The resulting weights are applied to the target steering vector using inner-product to obtain the desired signal. A target steering vector here is defined by its look-spatial-angle and look-Doppler-frequency in space and time, respectively. Therefore, steering vector represents a set of phase delays in space and time.

Most STAP algorithms perform the following steps (cf. [116]):

1. Starting with the data cube, identify the cell under test (corresponding to data vector $\mathbf{s}$ of length LT_{dof} and form the target steering vector $\mathbf{v}$ for every Doppler bin of interest.

2. Select N_{T} representative training data from range bins on both sides of the cell under test keeping the guard cells.

3. Compose the estimate of interference covariance matrix $\tilde{\mathbf{R}}$ using the training data for methods that require use of covariance matrix.

4. Calculate the weight vector $\mathbf{w}$ and apply the weight vector to test cell data to obtain a test statistics, $\Lambda \propto \mathbf{w}^H\mathbf{v}$. Here, $\mathbf{w}^H$ represents the Hermetian transpose of weight vector $\mathbf{w}$. For STAP techniques using covariance matrix inverse, the weight vector is computed as $\mathbf{w} \propto \tilde{\mathbf{R}}^{-1}\mathbf{v}$.

5. Compare the test statistic Λ to a threshold value λ corresponding to a specified probability of false alarm and declare target presence when the test statistic exceeds the threshold.

4.2 Description

The input to STAP consists of a data cube with complex number entries corresponding to data of a coherent processing interval (CPI). A data cube is of size equal to the number of channels L times the number of Doppler bins K times the number of range bins N. The solution to the STAP constraining problem of weights estimation can be obtained using different methods that can make use one or more algorithms. A critical challenge in STAP procedure is to address the heterogeneous and non-Gaussian interference encountered in radar data. Therefore it is important to evaluate different solutions to STAP weights-estimation while processing both homogeneous and non-homogeneous interference problems that are encountered in real STAP applications. This constraining problem of STAP weights-estimation uses two sets of data, one with homogeneous and another with heterogeneous radar data. The output is an interference suppressed signal. Since there are D steering vectors in a given STAP problem, the output signal vector will have $N \times L \times D$ elements.

4.2.1 Input

The parameters for computation of STAP output signal are given in Table 4.1. The input consist of a STAP data cube with complex number entries, complex steering vector elements corresponding to data of a coherent processing interval (CPI). The data cube is of size equal to $N \times K \times L$ complex elements corresponding to the number of channels L, number of Doppler bins K, and number of range bins N. The temporal degrees of freedom T_{dof} is provided. The

Table 4.1: STAP inner-product with data extraction parameters.

Parameter	Variable (Dimension)
Spatial channels/elements	L
Doppler bins	K
Range bins	N
Temporal degrees of freedom	T_{dof}
Data cube (complex valued)	$(N \times K \times L)$

complex steering vector elements $\mathbf{v}$ has D complex vectors, each of size equal to LT_{dof}.

The input data set is provided for a simulated monostatic sidelooking radar on a moving or stationary platform at a given height and velocity. The simulated data uses specified spacing for antennas with uniform transmit pattern at a given carrier frequency and pulse repetition frequency (PRF). The mainbeam azimuth is centered at 0°. Uniformly distributed clutter and a jammer location is provided between azimuth $-\pi/2$ to $+\pi/2$. A homogeneous and non-homogeneous clutter-to-noise ratio (CNR) per pulse, per channel is provided while a variable jammer to noise ratio at different jammer location/s is used. At least 2 targets with locations centered around 0° azimuth and given Doppler frequencies with variable signal-to-noise-ratio form the input data. In order to validate the mathematical robustness of a given STAP algorithm used for weights estimation and compute the output signal, suitable metric is used.

4.2.2 Output

The output signal vector has $N \times L \times D$ elements. The output signal is computed via estimates of adaptive weighting elements. The complex array of weighting vector elements $\mathbf{w}$ has $N \times K \times D$ complex vectors corresponding to the number of range bins N, number of Doppler bins K, and number of steering vectors, D. Each complex element of the weighting vector is of size equal to LT_{dof} corresponding to the number of channels L and the temporal degree of freedom T_{dof}. The scalar output signal y is obtained by performing inner-product computations on the snapshot vector with the weighting vector array $\mathbf{w}$. The complex output signal y is computed over $N \times K \times D$ elements as follows:

$$y(n,k,d) = \gamma(n,k,d)\langle \mathbf{w}(n,k,d), \mathbf{s}(n,k)\rangle \tag{4.1}$$

where $d = 1 \ldots D$ are indices to steering vectors. The scalar weights array multiplier γ represents a scalar multiplier. Thq quality of solution is equaluated using the output signal to noise ratio (SNR) or signal to interferrence plus noise ratio (SINR) as the case may be.

The output SINR is computed as per Equation 4.2. The input SINR is equal

to the SINR at a given spatial-temporal element as per given radar input data.

$$\mathrm{SINR}_{\mathrm{out}} = \frac{E\left[\mathbf{w}^H \mathbf{t}\mathbf{t}^H \mathbf{w}\right]}{E\left[\mathbf{w}^H \mathbf{s}_{\mathrm{H}_0} \mathbf{s}_{\mathrm{H}_0}^H \mathbf{w}\right]} \tag{4.2}$$

where $E[\cdot]$ denotes the expectation operator and $E[\mathbf{s}_{\mathrm{H}_0}\mathbf{s}_{\mathrm{H}_0}^H]$ is the interference covariance matrix. Vectors $\mathbf{s}$ and $\mathbf{t}$ are the data-snapshot and target-snapshot vectors, respectively. The suffix H_0 represent the condition indicating target absence.

4.3 Evaluation

4.3.1 Correctness

The computations are expected to follow the IEEE standard for floating-point arithmetic (IEEE 754-2008) on a standard x86 system. The SINR metric is averaged over a large number of Monte Carlo trials using the input data. A given STAP algorithm will make use of certain total number of spatio-temporal degrees of freedom and a number of snapshots from the range bin which will also affect the performance. Therefore these two performance metrics will be compared based on identical numbers for spatio-temporal degrees of freedom and number of snapshots used in different STAP solutions.

4.3.2 Performance and Power

The unit of work is an element in the weighting vector. Consequently, there are $N \times K \times D$ units of work; corresponding to number of range bins N, number of Doppler bins K, and number of steering vectors D.

Performance is evaluated with time per units of work.

Power is evaluated with Watts per unit of work.

Constraining Problem 5

Radio: Space-Time Adaptive Processing Target Detection

Nitin A. Gawande (PNNL), Joseph B. Manzano (PNNL), Nathan R. Tallent (PNNL)

5.1 Justification

Compared to using a single sensor, signal detection using an array of sensors leads to improvement in the Signal-to-Noise-Ratio (SNR) and has offered significant benefits in applications such as radar, sonar, satellite communications, and seismic systems [96]. Electromagnetic wave signals recorded by an array of antenna receivers as voltage signals over spatial and temporal domain constitutes the radar receiver signal data. Detecting a desired target-signal against background clutter and interference is an important challenge because the spectral characteristics of the interference and the target amplitude are often unknown. The background clutter has a greater impact on target detection for moving platforms such as advanced airborne surveillance radar systems [34]. Space-Time Adaptive Processing (STAP) algorithms employ spatial and temporal adaptive processing to suppress the interference in radar signal data and thus find targets which otherwise may not be detected. The STAP procedure achieves this by computing suitable weights that are applied to the radar signal to obtain desired target signal. These weights are said to be computed 'adaptively' due to the fact they reflect the actual clutter and interference in the environment.

The input to STAP is a three-dimensional radar data cube consisting of L channels, or the phase centers on radar that are displaced along the velocity vector of the platform; P pulses; and N samples per pulse. A pulse is one transmission of radar energy. The N samples per pulse are commonly termed range cells or range bins because they sample at range (or time) intervals. Because of the time scales involved, the range dimension is referred to as fast-time and the pulse dimension is referred to as slow-time. The L channels and P pulses add spatial and temporal diversity, respectively, to the acquired data; these dimensions correspond to the 'space' and 'time' in space-time adaptive processing. As a first step of a typical STAP algorithm like the extended Doppler-factored algorithm (EFA) [34], the pulse (time) domain is converted to Doppler spectrum domain. This is achieved by applying a discrete Fourier transform (DFT) to the P pulses for each range cell and channel pair resulting in K Doppler bins. We assume that DFT is already applied to the data set and we have a data cube structure as shown in Figure 5.1. A 2-D space-time snapshot corresponding to a particular range cell is also shown in Figure 5.1. Optimum adaptive weights are desired such that their application to the received signal vector enhances the signal to noise ratio.

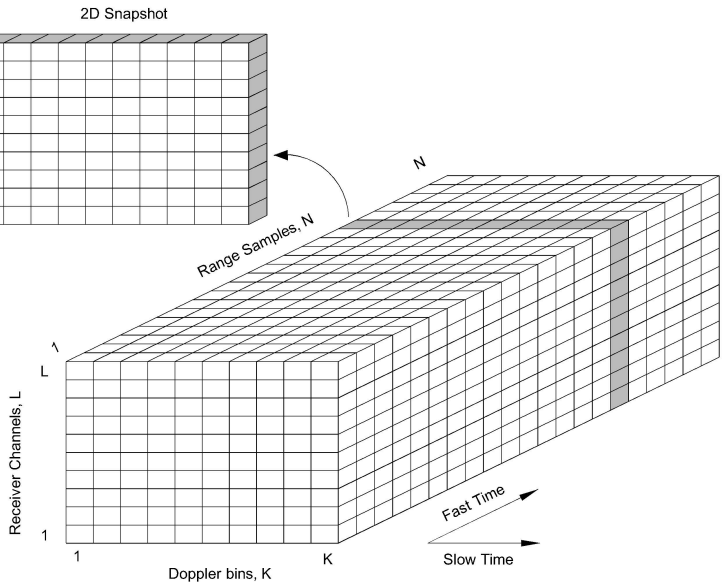


Figure 5.1: Data cube from a single coherent processing Interval (CPI)

The several STAP algorithms approach the problem in a different ways and have different levels of computational complexities. The key output requirement is an estimate of weights that obtains an output signal with desired characteristics. The real-time application of STAP brings performance constraints. Airborne platforms have power constraints. We therefore describe weights estimation as one of the most constraining problems in the STAP procedure.

Statistical STAP techniques use the interference covariance matrix to obtain adaptive weights [114]. Training data samples from range bins are used to compute the covariance matrix. These training data samples are chosen such that they are representative of the stationary clutter present in the cell under test. The adaptive weights are computed for a particular range cell and Doppler bin using statistics of the space-time snapshot vector consisting of data samples from the L array channels and T_{dof} adjacent Doppler bins. Here, T_{dof} is the temporal degree-of-freedom in a given STAP procedure. This requires composing $LT_{\text{dof}} \times LT_{\text{dof}}$-sized interference covariance matrices and applying subsequent operations to these matrices to get weights. The resulting weights are applied to the target steering vector using inner-product to obtain the desired signal. A target steering vector here is defined by its look-spatial-angle and look-Doppler-frequency in space and time, respectively. Therefore, steering vector represents a set of phase delays in space and time.

Most STAP algorithms perform the following steps (cf. [116]):

1. Starting with the data cube, identify the cell under test (corresponding to data vector $\mathbf{s}$ of length LT_{dof} and form the target steering vector $\mathbf{v}$ for every Doppler bin of interest.

2. Select N_{T} representative training data from range bins on both sides of the cell under test keeping the guard cells.

3. Compose the estimate of interference covariance matrix $\tilde{\mathbf{R}}$ using the training data for methods that require use of covariance matrix.

4. Calculate the weight vector $\mathbf{w}$ and apply the weight vector to test cell data to obtain a test statistics, $\Lambda \propto \mathbf{w}^H\mathbf{v}$. Here, $\mathbf{w}^H$ represents the Hermetian transpose of weight vector $\mathbf{w}$. For STAP techniques using covariance matrix inverse, the weight vector is computed as $\mathbf{w} \propto \tilde{\mathbf{R}}^{-1}\mathbf{v}$.

5. Compare the test statistic Λ to a threshold value λ corresponding to a specified probability of false alarm and declare target presence when the test statistic exceeds the threshold.

The computational load in STAP is a function of the spatial-temporal degree of freedom [116]. Moreover, the sample covariance matrix based methods require adequate training samples over range bins to compute the covariance matrix for a given spatial-temporal degree of freedom. The large training data requirement also increases the computational load for STAP. The fully adaptive covariance matrix based methods perform poorly on radar data with heterogeneous and non-Gaussian interference. In order to overcome the two-fold problem of large training data support requirement and heterogeneity of training data, partially adaptive methods requiring a reduced spatial-temporal degree of freedom were developed [116]. Some of these algorithms include the joint domain localized (JDL) processing algorithm [113], the parametric adaptive matched filter (PAMF) [98], the multistage Weiner filter (MWF) [43], extended-factored STAP [114], normalized fractionally-lower order moment (NFLOM) [126], Bayesian covariance estimation methods [45], the low-rank clutter covariance matrix (CCM) methods [41]. Some methods follow the deterministic least-square approach like the direct data domain (D^3) method [101] which does not require statistical training. Another approach is to make use of hybrid stap procedure [1] that uses more than one method including some of the above listed methods and address the problem of heterogeneous and non-Gaussian interference.

5.2 Description

The input to STAP consists of a data cube with complex number entries corresponding to data of a coherent processing interval (CPI). A data cube is of size equal to the number of channels L times the number of Doppler bins K times the number of range bins N. The solution to the STAP constraining

problem of weights estimation can be obtained using different methods that can make use one or more algorithms. A critical challenge in STAP procedure is to address the heterogeneous and non-Gaussian interference encountered in radar data. Therefore it is important to evaluate different solutions to STAP weights-estimation while processing both homogeneous and non-homogeneous interference problems that are encountered in real STAP applications. This constraining problem of STAP weights-estimation uses two sets of data, one with homogeneous and another with heterogeneous radar data. The output is an interference suppressed signal. Since there are D steering vectors in a given STAP problem, the output signal vector will have $N \times L \times D$ elements.

5.2.1 Input

Table 5.1: STAP inner-product with data extraction parameters.

Parameter	Variable (Dimension)
Spatial channels/elements	L
Doppler bins	K
Range bins	N
Temporal degrees of freedom	T_{dof}
Data cube (complex valued)	$(N \times K \times L)$

The parameters for computation of STAP output signal are given in Table 5.1. The input consist of a STAP data cube with complex number entries, complex steering vector elements corresponding to data of a coherent processing interval (CPI). The data cube is of size equal to $N \times K \times L$ complex elements corresponding to the number of channels L, number of Doppler bins K, and number of range bins N. The temporal degrees of freedom T_{dof} is provided. The complex steering vector elements $\mathbf{v}$ has D complex vectors, each of size equal to LT_{dof}.

The input data set is provided for a simulated monostatic sidelooking radar on a moving or stationary platform at a given height and velocity. The simulated data uses specified spacing for antennas with uniform transmit pattern at a given carrier frequency and pulse repetition frequency (PRF). The mainbeam azimuth is centered at 0°. Uniformly distributed clutter and a jammer location is provided between azimuth $-\pi/2$ to $+\pi/2$. A homogeneous and non-homogeneous clutter-to-noise ratio (CNR) per pulse, per channel is provided while a variable jammer to noise ratio at different jammer location/s is used. At least 2 targets with locations centered around 0° azimuth and given Doppler frequencies with variable signal-to-noise-ratio form the input data. In order to validate the mathematical robustness of a given STAP algorithm used for weights estimation and compute the output signal, suitable metrics are used. Probability of detection (P_{D}) is one such metric and is calculated using another parameter, probability of false alarm (P_{FA}). The probability of false alarm is provided for evaluation purpose.

5.2.2 Output

The output signal vector has $N \times L \times D$ elements. The output signal is computed via estimates of adaptive weighting elements. The complex array of weighting vector elements $\mathbf{w}$ has $N \times K \times D$ complex vectors corresponding to the number of range bins N, number of Doppler bins K, and number of steering vectors, D. Each complex element of the weighting vector is of size equal to LT_{dof} corresponding to the number of channels L and the temporal degree of freedom T_{dof}. The scalar output signal y is obtained by performing inner-product computations on the snapshot vector with the weighting vector array $\mathbf{w}$. The complex output signal y is computed over $N \times K \times D$ elements as follows:

$$y(n,k,d) = \gamma(n,k,d)\langle \mathbf{w}(n,k,d), \mathbf{s}(n,k)\rangle \tag{5.1}$$

where $d = 1 \ldots D$ are indices to steering vectors. The scalar weights array multiplier γ represents a scalar multiplier.

Using the output signal y, a test statistic $\Lambda(y)$ is obtained. This value of Λ is compared with a threshold value λ to detect presence of target. This comparison is equivalent to comparing the output signal power to a value equal to fixed threshold times a multiplier representative of noise power. The mathematical robustness of various solutions to STAP problems is evaluated here using two performance metrics, probability of detection (P_{D}) and improvement factor (IF).

The probability of detection P_{D} performance is computed as a function of signal-to-noise ratio (SNR) for a given probability of false alarm P_{FA}. The expressions for P_{FA} and P_{D} are given by Equations 5.2 and 5.3, respectively [75].

$$P_{\text{FA}} = \exp\left(\frac{-\beta_T^2}{2}\right) \tag{5.2}$$

$$P_{\text{D}} = \int_{\beta_T}^{\infty} u \exp\left(\frac{-\left(u^2+\alpha^2\right)}{2}\right) I_0\left(\alpha u\right) du \tag{5.3}$$

where β_T is a normalized detection threshold, $I_0(\cdot)$ is the modified zero-order Bessel function of the first kind, and α is equal to the square-root of the peak signal-to-interference-plus-noise ratio (SINR). The computation of P_{D} using the above equation for both Gaussian and non-Gaussian clutter scenarios can be mathematically difficult. Therefore finite sum expression corresponding to a specific STAP algorithm is used for computation of P_{D}. These computations are performed over a large number of simulations using Monte Carlo Technique [77].

Improvement factor (IF) is a ratio of output SINR to input SINR (Equation 5.4). The improvement factor is computed over given space-time elements. The output SINR is computed as per Equation 5.5. The input SINR is equal to the SINR at a given spatial-temporal element as per given radar input data.

$$\text{IF} = \frac{\text{SINR}_{\text{out}}}{\text{SINR}_{\text{in}}} \tag{5.4}$$

$$\text{SINR}_{\text{out}} = \frac{E\left[\mathbf{w}^H \mathbf{t}\mathbf{t}^H \mathbf{w}\right]}{E\left[\mathbf{w}^H \mathbf{s}_{\text{H}_0} \mathbf{s}_{\text{H}_0}^H \mathbf{w}\right]} \tag{5.5}$$

where $E[\cdot]$ denotes the expectation operator and $E[\mathbf{s}_{\mathrm{H}_0}\mathbf{s}_{\mathrm{H}_0}^H]$ is the interference covariance matrix. Vectors $\mathbf{s}$ and $\mathbf{t}$ are the data-snapshot and target-snapshot vectors, respectively. The suffix H_0 represent the condition indicating target absence.

5.3 Evaluation

5.3.1 Correctness

The computations are expected to follow the IEEE standard for floating-point arithmetic (IEEE 754-2008) on a standard x86 system. The two metrics probability of detection (P_{D}) and improvement factor (IF) are averaged over a large number of Monte Carlo trials using the input data. A given STAP algorithm will make use of certain total number of spatio-temporal degrees of freedom and a number of snapshots from the range bin which will also affect the performance. Therefore these two performance metrics will be compared based on identical numbers for spatio-temporal degrees of freedom and number of snapshots used in different STAP solutions.

5.3.2 Performance and Power

The unit of work is an element in the weighting vector. Consequently, there are $N \times K \times D$ units of work; corresponding to number of range bins N, number of Doppler bins K, and number of steering vectors D.

Performance is evaluated with time per units of work.

Power is evaluated with Watts per unit of work.

Constraining Problem 6

Image: Image Registration

Nitin A. Gawande (PNNL), Nathan R. Tallent (PNNL), Joseph B. Manzano (PNNL)

6.1 Justification

Image registration is a fundamental task used in image processing. Registration is the process of matching or overlaying two or more images taken at different times, from different sensors, or from different viewpoints [17]. Image registration is used in important applications of Wide Area Motion Imaging (WAMI). Optical moving target indication (MTI) is one of the critical operations of WAMI surveillance systems such as ARGUS [63]. MTI is expected to execute over the entire field of view at a minimum of 2 Hz, but higher frequencies are often needed. Image registration is computationally very intensive. The need to process large images at a high frequency makes image-registration a constraining problem for MTI. Typically, an MTI pipeline uses frame-to-frame registration followed by change detection. Other systems of importance where image registration is a significant component include monitoring global land use using satellite images, matching stereo images for autonomous airborne vehicle navigation, and medical imaging [17]. Image registration is also critical to other image processing applications such as image mosaicking, compression, and video processing. Image processing rates for such applications vary between 5 to 100 Hz along with varying input image sizes. Such requirements make image registration one of the constraining problems in image processing.

6.2 Description

Image registration can be defined as a mapping between two images both spatially and with respect to intensity [17]. Given an input image $\mathbf{I}$ which need to registered on a template image $\mathbf{T}$ denoted by 2D arrays $\mathbf{I}(x, y)$ and $\mathbf{T}(x, y)$, respectively. Then the mapping between images is given by Equation 6.1.

$$\mathbf{I}(x, y) = \mathrm{g}\left(\mathbf{T}\left(f(x, y)\right)\right) \tag{6.1}$$

where, f is a 2D spatial-coordinate transformation which maps two spatial coordinates, x and y, to new coordinates x' and y' such that $(x', y') = f(x, y)$. g is a 1D intensity transformation. Thus the image registration problem is to find optimal spatial and intensity transformations so that the images are matched. The change in intensity may arise due to change in sensors or by change in reflectance seen by sensor, or due to differences in the imaging scene. Spatial distortions between images arise due to differences in acquisition and scene characteristics which affect acquisition.

There are several methods to carry out the image registration procedure as given by Equation 6.1. These methods are typically categorized as intensity based, feature based, or hybrid [17]. Mendoza-Schrock et al. (2009) [76] evaluated several image-to-image registration methods with respect to the accuracy and robustness and more information of these methods can be found else where [17]. A novel solution to this constraining problem can include intensity-based methods (including correlation-based) and any feature-based methods that already include information on feature templates and do not require additional input information from the user.

6.2.1 Input

The input generator computer program generates a sequence of variable L number of grayscale images using one input image. A user has the choice to pick from different images from a set of images and also choose different image size. The sequence of L images is obtained for a chosen input template image after applying a series of image transformations such as translation in two directions, rotation, shear in two directions, and scaling in two directions. During evaluation of image registration, manual selection of domain for computation of evaluation metrics is a carried out [46], which is a cumbersome task. The images provided in this input image data set are padded at boundary uniformly with low intensity value. The use of padding minimizes the error which otherwise is introduced significantly at the edges, making evaluation of image registration very difficult. Padding of input images makes it possible to automate the computation of evaluation metrics for registered images. We provide three different image sizes with 256×256 pixels, 512×512 pixels, and 1024×1024 pixels corresponding to small, medium, and large sizes, respectively.

The input set of images contain following images in RYB color scheme for which information is included in the input generator numerical computer program.

- *Geometry-01:* This is an image with simple geometry of two squares rendered uniformly.
- *Geometry-02:* This is an image with two squares rendered uniformly overlaid on a triangle shaped line object and a circle represented as line objects.
- *Geometry-02:* This is an image with two squares rendered uniformly overlaid on a triangle shaped line object and a uniformly rendered circle as another object.
- *Chem-img-PNNL-01:* This is an image of nano-particles obtained using high resolution Transmission Electron Microscopy (TEM) with ultrasensitive nanoscopy. This image was obtained through the *PictureThis*, which is an online collection of photos, graphics, videos, and related image files at the Pacific Northwest National Laboratory (`http://picturethis.pnl.gov/`).

- *Lena-image:* An image of human face.
- *Airplane:* Photograph of an airplane obtained from *PictureThis*.
- *Richland-aerial-01:* Aerial Photograph of mixed urban, grassland, and water-body landscape obtained via open source.
- *Columbia-valley-landscape:* Photograph of mixed grassland and water-body landscape from *PictureThis*.

The input data are sequence of L images generated with the use of input generator while using a single template image of either small, medium, or large size. With the choice of several template images with three different sizes an large number of image sequences are used for evaluation of a novel solution to this constraining problem.

6.2.2 Output

The output is a series of image registrations, where image i ($i = 1 \dots L$) is registered on the input template image (in grayscale). The sequence of image registrations is checked against the original template image, which is used in the input generator. A comparison of mean square error (MSE) values is used to compare different solutions to this constraining problem. Any method is considered acceptable when the MSE values is less than or equal to the threshold value provided for a given input image.

6.3 Evaluation

6.3.1 Correctness

The input image I is registered as image R on the template image T. The Mean Square Error (MSE) between the template image, T and the registered image R is given by Equation 6.2.

$$\mathrm{MSE} = \left(\frac{\sum_{i=1}^{M} \sum_{j=1}^{N} \left[\mathbf{T}(i,j) - \mathbf{R}(i,j) \right]^2}{M \times N} \right) \tag{6.2}$$

where, $\mathrm{R}(i,j)$ and $\mathrm{F}(i,j)$ are the image pixel values of the reference (golden standard) image and the fused image, respectively. $(M \times N)$ is the image size corresponding to number of rows M and number of columns N of pixel.

An average of MSE is computed for total of L registration sequences performed. The peak signal to noise ratio is computed for an average MSE values as per Equation 6.3.

$$PSNR = \begin{cases} \frac{10 \cdot log_{10}\left(I_{\max}^2 / \mathrm{MSE}\right)}{log(10)}, & \text{if MSE} > 0 \\ 120, & \text{if MSE} = 0 \end{cases} \tag{6.3}$$

In Equation 6.3, $I_{\max}$ is the maximum pixel value for grayscale images.

6.3.2 Performance and Power

Execution time for the entire sequence of image-to-image registrations. This number will be normalized for total number of registrations performed and by the total number of pixels in each image.

The average power consumed the entire sequence of image-to-image registrations. The power will be normalized for total number of registrations performed and by the total number of pixels in each image. We also use peak power and standard deviation.

6.4 Appendix

Mendoza-Schrock et al. (2009) [76] evaluated several image-to-image registration methods with respect to the accuracy and robustness. This study showed that the Lucas-Kanade method [71] outperformed other algorithms. Lucas-Kanade works well because it is designed to optimize the RMSE. However, other variants of Lucas-Kanade method and methods derived from this method have been shown to perform faster [83], [91]. Here, we describe in brief the Lucas-Kanade method.

Lucas-Kanade is an intensity based image-registration method (Lucas and Kanade, 1981). It is a gradient descent algorithm that minimizes the sum of square of errors between two images. Minimizing this sum is a non-linear optimization problem. The Lucas-Kanade method is essentially Gauss-Newton gradient descent non-linear optimization algorithm. This algorithm uses images in grayscale format. Gradient descent algorithms such as the Lucas-Kanade method work by minimizing the square of errors (Equation 6.2) between an image template T and input image I that is warped into the coordinate frame of the template. The Lucas-Kanade algorithm uses an initial estimate of warp parameter $\mathbf{p}$ and then iteratively solves until $\Delta\mathbf{p}$ becomes sufficiently small. In particular, the following expression is minimized:

$$E(x,y) = \sum_{N_x} \sum_{Ny} [I(W(\,(x,y);\mathbf{p})\,) - T(x,y)] \tag{6.4}$$

where, $W((x,y);\mathbf{p})$, denotes set of warps; $\mathbf{p} = \langle p_1, \ldots, p_n \rangle$, is a vector of warp parameters; and N_x and N_y are the number of pixels along the two coordinates.

Baker and Matthews [7] describe different gradient descent algorithms. These techniques broadly fall into two categories as either additive or compositional, and as forwards or inverse. Table 6.1 shows four categories of implementations based on the above two groups and their mathematical complexities. One other difference in the implementations of gradient descent algorithms is the use of different type of approximations. Tables 6.2 and 6.3 show the convergence properties and the complexities of different implementations.

Table 6.1: Various gradient descent image alignment algorithms showing their complexity and application, source [7].

Algorithm	**Example**	**Complexity**	**Application to**
Forward Additive	Lucas-Kanade (1981)	$O(n^2N + n^3)$	Any
Forward Compositional	Shum-Szeliski (2000)	$O(n^2N + n^3)$	Any semi-group
Inverse Additive	Hagen-Belhumeur (1998)	$O(nN + n^3)$	Simple linear $2D$
Inverse Compositional	Baker-Matthews (2001)	$O(nN + nK + k^3)$	Any group

Table 6.2: Convergence rate and convergence frequency of the six gradient descent approximations [7].

Algorithm	**Convergence rate**	**Convergence frequency**
Gauss-Newton	Fast	High
Newton	Medium	Medium
Steepest Descent	Slow	Low
Gauss-Newton Diagonal Hessian	Slow	Low
Newton Diagonal Hessian	Slow	Low
Levenberg-Marquardt	Fast	High

Table 6.3: Algorithm complexity with inverse and forward composition and the implementation efficiency [7].

Algorithm	**Complexity with inverse composition**	**Complexity with forward composition**
Gauss-Newton	$O(nN + n^3)$ Efficient	$O(n^2N + n^3)$
Newton	$O(n^2N + n^3)$	$O(n^2N + n^3)$
Steepest Descent	$O(nN + n^2)$ Efficient	$O(nN + n^2)$ Efficient
Gauss-Newton Diagonal Hessian	$O(nN + n^2)$ Efficient	$O(nN + n^2)$ Efficient
Newton Diagonal Hessian	$O(nN + n^2)$ Efficient	$O(nN + n^2)$ Efficient
Levenberg-Marquardt	$O(nN + n^3)$ Efficient	$O(n^2N + n^3)$

Constraining Problem 7

Image: Multisensor Image Fusion

Nitin A. Gawande (PNNL), Joseph B. Manzano (PNNL), Nathan R. Tallent (PNNL)

7.1 Justification

Multisensor fusion refers to combining different sources of sensory information [127]. A fused image may contain a richer or more accurate description of a scene in comparison to any individual source images [127]. As a result, fused images may enable detection of more targets or increased accuracy in detecting targets. The use of multisensor image fusion in areas of military and defense, medical imaging, satellite imagery, geosciences, and a number of industrial applications is well documented [13], [87]. These methods find wide use in military and security related applications such as night-vision, synthetic aperture radar (SAR), satellite imagery, and screening check-points [131], [64], [127], [12].

When sensors measure the same physical phenomenon, then fusion is possible. However, fusion may occur at different levels. If sensor data is noncommensurate, then these data must be fused at a higher or feature level [47]. Signal-level image fusion refers to the process of combining multiple input image signals into a single fused image output [86].

This problem focuses on pixel-level image fusion, the lowest level of image fusion for multisensor images. Pixel-level fusion helps to improve the performance of image processing tasks such as segmentation and feature extraction [65]. Pixel-level image fusion require that [130]: (1) the fusion process preserve all relevant information from the input images into the fused image; and (2) the fusion process should not introduce any artifacts or inconsistencies.

Multisensor image fusion implementations have several challenges. Although pixel-level image fusion algorithms are well known, challenges remain with respect to exploiting sensor modalities, robustness to environmental and operational conditions and offering performance benefit [104]. A critical performance issue is system latency. A sophisticated fusion algorithm runs the risk of not offering benefits if the latency is too high for real-time implementation [104]. The limits on the power requirements of computing devices used for image fusion in real-time applications add another dimension to this challenge. As a result, *multisensor image fusion* is a constraining problem in several DoD applications.

7.2 Description

For informational purposes, this section first gives an overview of different image fusion methods. We then describe input and output specifications for this constraining problem.

There are several methods to perform image fusion. Image fusion algorithms/methods can be classified broadly in three categories: (1) Weighted averaging; (2) Multiscale Methods; and (3) Other methods.

Weighted Averaging

This is the simplest image fusion method based on linear weighted average of pixels of source images [104]. This method is easy to implement and fast to execute. However, a fused image obtained through this method has low contrast. This leads to suppression of important image features in the fused image.

Multiscale Methods

Multiscale image fusion methods are further categorized into (a) Pyramid methods and (b)Discrete wavelet transform (DWT) based methods;

Pyramid image fusion methods are multiscale decomposition methods where a pyramid structure can be used to represent collection of images at different scales [127]. Pyramid fusion methods may differ in implementation and categorized as under:

- Laplacian pyramid
- Ratio of low-pass pyramid and contrast pyramid
- Gradient pyramid
- Morphological pyramid

Burt and Adelson [19] showed implementation of the Laplacian pyramid method. In a Laplacian pyramid, each level of pyramid is recursively constructed from its lower level following a series of steps.

The DWT based multiscale methods differ from other pyramid methods of the first category mentioned above. In DWT based pyramid methods, successive layers of the pyramid include only additional details that are not available at preceding levels [73].

DWT based byramid methods are further categorized as under:

- Only DWT based method
- DWT with principal component analysis (PCA) and morphological processing
- Hybrid methods using combined pyramid and DWT methods
- Other variants of DWT method

Other Methods

Image fusion methods not included in above two categories can be grouped separately. Multisensor image fusion methods based on compressive sensing [112] and methods using neural network [16] [66] algorithm can be grouped in this category. Compressive sensing offers advantage such that a fused image can be obtained without having to acquire the original input images [112]. This method is implemented using following procedure [112]:

1. Take the compressive measurements, Y_{D} of input images $\mathbf{A}$ and $\mathbf{B}$ ($D = \{\mathbf{A}, \mathbf{B}\}$) using a suitable sampling pattern.

2. Calculate vector of fused measurements, Y_{M} with measurements $M = \Psi_{\mathrm{D}}(Y_{\mathrm{D}})$ for $D = \{\mathbf{A}, \mathbf{B}\}$; Ψ is a function used for compressive sensing of input images.

3. Reconstruct the fused image $\mathbf{F}$ from the composite measurements Y_{F} via total variation minimization method [20].

7.2.1 Input

A prerequisite for successful image fusion is that multisensor images are correctly aligned on pixel-by-pixel basis [64]. Registration of images prior to fusion is very important and we consider that the input images used for fusion are already registered. Any image resampling which is often required prior to image registration is also applied to input images. As the main focus of this constraining problem is the basic image fusion process, only grayscale images are used. Four sets of grayscale images at three different resolutions are used as input. Each image has M number of pixel rows and N pixel columns. These input images are as follows:

1. One visible and one single band infrared image as input. These images are from a scenery in grassland environment with 2 or more humans in the background.

2. One visible and two infrared images from two different bands. These images are from a scenery in grassland environment with 2 or more humans in the background.

3. Two multi-focued visible images from a scenery in urban environment.

4. One visible, one single band infrared, and one submillimeter (SMM) image of a human with hidden metal object detected in SMM image.

7.2.2 Output

Any of the above mentioned image fusion schemes or any other alternative scheme is used to obtain a fused image as per input. A generalized image fusion procedure is shown in Figure 7.1. The output is one fused image for one set of input images. The fused image will have a resolution equal to the size of input images.

An implementation of image fusion algorithm that introduces noise in the fused image due to inherent characteristics of the algorithm and that the fused image requires an use of enhancement procedure for the evaluation of fused image, then the enhancement algorithm will be treated as a part of image fusion algorithm. The fused image obtained through the application of suitable image

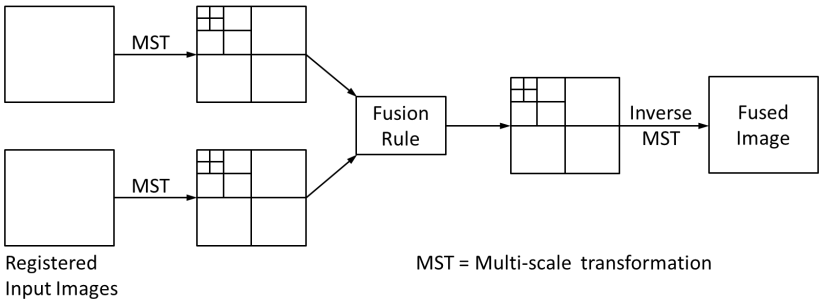


Figure 7.1: A generalized representation of image fusion procedure.

fusion algorithm will be evaluated using fusion evaluation metrics described below.

A large number of quantitative image fusion metrics are used in practice to evaluate quality of fused image obtained through the use of image fusion algorithms [12], [69], [129], [68]. Image fusion metrics broadly fall into two categories, one that require a reference image and another that does not. For metrics that require a reference image, the fusion metric is a measure of deviation of fused image with respect to reference image.

The fused output image will be evaluated using some or all of the fusion metrics described here. An additional metric, complimentary to the ones mentioned here will be included for any novel image fusion implementation that inherently demand such requirement.

The absolute values of metrics used for evaluation of image fusion algorithms are highly dependent on the input images that are fused. Therefore it is critical to have a basis for comparison of novel image fusion implementations. An absolute value of fusion metric using a reference implementation will be used to normalize the fusion metric value obtained using any other implementation. The quantitative metrics that will be used to evaluate any fusion method are described here.

Fusion Metrics

1. Root mean square error (RMSE)

   The RMSE between the reference image and fused image is given by Equation 7.1.

$$\text{RMSE} = \left(\frac{\sum_{i=1}^{M} \sum_{j=1}^{N} [\mathbf{R}(i,j) - \mathbf{F}(i,j)]^2}{M \times N} \right)^{\frac{1}{2}} \tag{7.1}$$

   where, $I_{\text{R}(i,j)}$ and $I_{\text{F}(i,j)}$ are the image pixel values of the reference (golden standard) image and the fused image, respectively. $(M \times N)$ is the image

size corresponding to number of rows M and number of columns N of pixel.

2. Entropy (H) Entropy (H) is indicator of image complexity and thus a measure of information contained in an image.

$$H = -\sum_{l=0}^{L-1} p(l)\ \log_2 p(l) \tag{7.2}$$

where, $p(l)$ is the probability of gray level which is computed using data of image histogram, $[0, L-1]$ is the dynamic range of analyzed image. For a 8-bit single channel image L is equal to 256.

3. Spatial frequency (SF)

Zheng et al. [130] used this metric to measure the overall activity level of an image $I(i,j)$ as defined below:

$$\mathrm{SF} = \sqrt{(RF)^2 + (CF)^2 + (MDF)^2 + (SDF)^2} \tag{7.3}$$

where, RF and CF are row frequency and column frequency respectively; MDF and SDF are frequencies along the main diagonal and secondary diagonal, respectively.

$$RF = \sqrt{\frac{1}{MN}\sum_{i=1}^{M}\sum_{j=2}^{N}\left[I\left(i,j\right) - I\left(i,j-1\right)\right]^2} \tag{7.4}$$

$$CF = \sqrt{\frac{1}{MN}\sum_{j=1}^{N}\sum_{i=2}^{M}\left[I\left(i,j\right) - I\left(i-1,j\right)\right]^2} \tag{7.5}$$

$$MDF = \sqrt{w_{\mathrm{d}}\frac{1}{MN}\sum_{i=2}^{M}\sum_{j=2}^{N}\left[I\left(i,j\right) - I\left(i-1,j-1\right)\right]^2} \tag{7.6}$$

$$SDF = \sqrt{w_{\mathrm{d}}\frac{1}{MN}\sum_{j=1}^{N-1}\sum_{i=2}^{M}\left[I\left(i,j\right) - I\left(i-1,j+1\right)\right]^2} \tag{7.7}$$

where, $w_{\mathrm{d}} = 1/\sqrt{2}$ is distance weight for diagonals. Equation 7.3 is used to compute spatial frequency $\mathrm{SF_A}$, $\mathrm{SF_B}$, and $\mathrm{SF_F}$ for input images **A** and **B**, and fused image **F**, respectively. The reference SF ($\mathrm{SF_R}$) is computed based on calculation of gradients along four directions. This gradient, denoted as $\mathrm{Grad}(I(i,j))$ is computed as per Equation 7.8.

$$\mathrm{Grad}^{\mathrm{D}}(I(i,j)) = \max\left\{\mathrm{abs}\left[\mathrm{Grad}^{\mathrm{D}}(I_{\mathrm{A}}(i,j))\right], \mathrm{abs}\left[\mathrm{Grad}^{\mathrm{D}}(I_{\mathrm{B}}(i,j))\right]\right\}, \quad \text{for each of four directions, i.e., } \mathrm{D} = \{\mathrm{H}, \mathrm{V}, \mathrm{MD}, \mathrm{SD}\} \tag{7.8}$$

where, the superscript 'D' denotes one of four directions along image row, column, main diagonal, and secondary diagonal. The gradient from Equation 7.8 is substituted into Equations 7.4 through 7.7 to obtain four reference frequency values. These four reference frequency values are used to compute the reference spatial frequency $\mathrm{SF_R}$. Finally, the performance metric for spatial frequency rSFe is computed using Equation 7.9.

$$\mathrm{rSFe} = \left(\mathrm{SF_F} - \mathrm{SF_R}\right)/\mathrm{SF_R} \tag{7.9}$$

Smaller the $rSFe$ value, better is the fused image. An ideal fusion has $rSFe = 0$; $rSFe > 0$ denotes an over-fused image with some distortion or noise introduced; $rSFe < 0$ denotes an under-fused image with loss of some meaningful information.

4. Gradient-based Fusion Metric

   Xydeas and Petrovic [121] proposed this metric to evaluate the amount of edge information transferred from input images to the fused image. The methodology adopted in this metric can be applied to more than two images. We consider two input images **A** and **B** resulting in a fused image **F**. A Sobel edge operator is applied to obtain the edge strength $g(n,m)$ and orientation $\alpha(n,m)$ information for each pixel(i,j); $1 \leq i \leq M$ and $1 \leq i \leq N$. For image **A**, these parameters are computed as follows:

$$g_{\mathrm{A}}(i,j) = \sqrt{s_{\mathrm{A}}^{x}(i,j)^2 + s_{\mathrm{A}}^{y}(i,j)^2} \tag{7.10}$$

$$\alpha_{\mathrm{A}}(i,j) = \tan^{-1}\left(\frac{s_{\mathrm{A}}^{y}(i,j)}{s_{\mathrm{A}}^{x}(i,j)}\right) \tag{7.11}$$

where $s_{\mathrm{A}}^{x}(i,j)$ and $s_{\mathrm{A}}^{y}(i,j)$ are the output of the horizontal and vertical Sobel template centered on pixel(i,j) and convolved with the corresponding pixels of image **A**. The relative strength $G^{\mathrm{AF}}(i,j)$ and $A^{\mathrm{AF}}(i,j)$ of an input image **A** with respect to fused image **F** are computed as follows:

$$G^{\mathrm{AF}}(i,j) = \begin{cases} \frac{g_{\mathrm{F}}(i,j)}{g_{\mathrm{A}}(i,j)}, & \text{if} g_{\mathrm{A}} > g_{\mathrm{F}}(i,j) \\ \frac{g_{\mathrm{F}}(i,j)}{g_{\mathrm{A}}(i,j)}, & \text{otherwise} \end{cases} \tag{7.12}$$

$$\Delta^{\mathrm{AF}}(i,j) = 1 - \frac{\mid \alpha_{\mathrm{A}}(i,j) - \alpha_{\mathrm{F}}(i,j) \mid}{\pi/2} \tag{7.13}$$

Above equations are used to derive the edge strength and orientation preservation values

$$Q_g^{\mathrm{AF}}(i,j) = \frac{\Gamma_g}{1+\exp^{\kappa_g(G^{\mathrm{AF}}(i,j)-\sigma_g)}} \tag{7.14}$$

$$Q_\alpha^{\mathrm{AF}}(i,j) = \frac{\Gamma_\alpha}{1+\exp^{\kappa_\alpha(\Delta^{\mathrm{AF}}(i,j)-\sigma_\alpha)}} \tag{7.15}$$

Constants Γ_g, κ_g, σ_g, and Γ_α, κ_α, σ_α determine the exact shape of the sigmoid functions used to form the edge strength and orientation values. Edge information preservation values are then defined as follows:

$$Q^{\mathrm{AF}}(i,j) = Q_g^{\mathrm{AF}}(i,j)Q_\alpha^{\mathrm{AF}}(i,j) \tag{7.16}$$

After computing $Q^{\mathrm{AF}}(i,j)$ and $Q^{\mathrm{BF}}(i,j)$ for $(M \times N)$ size images $\mathbf{A}$ and $\mathbf{B}$, respectively, a normalized weight performance metric $Q_P^{\mathrm{AB/F}}$ of a given fusion process P is obtained as follows:

$$Q_P^{\mathrm{AB/F}} = \frac{\sum_{i=1}^{M}\sum_{j=1}^{N} Q^{\mathrm{AF}}(i,j)w^{\mathrm{A}}(i,j) + Q^{\mathrm{BF}}(i,j)w^{\mathrm{B}}(i,j)}{\sum_{i=1}^{M}\sum_{j=1}^{N}(w^{\mathrm{A}}(i,j)+w^{\mathrm{B}}(i,j))} \tag{7.17}$$

where $w^{\mathrm{A}}(i,j) = \mid g_A(i,j) \mid^L$ and $w^{\mathrm{B}}(i,j) = \mid g_B(i,j) \mid^L$ are the weighting coefficients and L is a constant. This performance metric is bound such that $0 \leq Q_P^{\mathrm{AB/F}}(i,j) \leq 1$.

5. Similarity metric based on Universal Image Quality Index

   Cvejic et al., [29] proposed a similarity metric where each window block in spatial domain between input image and fused image is assigned a weighting factor similarity to fused image. The impact of less similar block in spatial domain is accordingly decreased. This metric is particularly important in cases where the input images are distorted versions of the ground-truth data; obtained by e.g. blurring, JPEG compression, noise addition, etc. The similarity metric Q_b is defined as per Equation 7.18.

$$Q_b = \sum_{w\in W} \mathrm{sim}(\mathrm{A},\mathrm{B},\mathrm{F}\,|\,w)Q(\mathrm{A},\mathrm{F}\,|\,w) + (1-\mathrm{sim}(\mathrm{A},\mathrm{B},\mathrm{F}\,|\,w))Q(\mathrm{B},\mathrm{F}\,|\,w) \tag{7.18}$$

   where $\mathbf{A}$ and $\mathbf{B}$ are the input images, $\mathbf{F}$ is the fused image, w is the analysis window block, and W is the family of all windows. The function $\mathrm{sim}(\mathrm{A},\mathrm{B},\mathrm{F}\,|\,w)$ is given by Equation 7.19.

$$\mathrm{sim}(\mathrm{A},\mathrm{B},\mathrm{F}\,|\,w) = \begin{cases} 0, & \text{if}\, \frac{\sigma_{\mathrm{AF}}}{\sigma_{\mathrm{AF}}+\sigma_{\mathrm{BF}}} < 0, \\ \frac{\sigma_{\mathrm{AF}}}{\sigma_{\mathrm{AF}}+\sigma_{\mathrm{BF}}}, & \text{if} 0 \leq \frac{\sigma_{\mathrm{AF}}}{\sigma_{\mathrm{AF}}+\sigma_{\mathrm{BF}}} \leq 1, \\ 1, & \text{if}\, \frac{\sigma_{\mathrm{AF}}}{\sigma_{\mathrm{AF}}+\sigma_{\mathrm{BF}}} > 1. \end{cases} \tag{7.19}$$

where σ_{AF} and σ_{BF} are defined as follows:

$$\sigma_{\mathrm{AF}} = \frac{1}{MN-1} \sum_{i=1}^{M} \sum_{j=1}^{N} (\mathbf{A}(i,j) - \bar{\mathrm{A}})(\mathbf{F}(i,j) - \bar{\mathrm{F}}) \tag{7.20}$$

$$\sigma_{\mathbf{BF}} = \frac{1}{MN-1} \sum_{i=1}^{M} \sum_{j=1}^{N} (\mathbf{B}(i,j) - \bar{\mathrm{B}})(\mathbf{F}(i,j) - \bar{\mathrm{F}}) \tag{7.21}$$

where $\bar{\mathrm{A}}$, $\bar{\mathrm{B}}$, and $\bar{\mathrm{F}}$ are the average of image signals for images **A**, **B**, and **F**, respectively.

6. Normalized Fusion Mutual Information (NFMI)

   Mutual information metric is a quantitative measure of degree of dependence of two images. Qu et al. [94] provided the expression to estimate the joint information between two source images $\mathrm{A}(i,j)$ and $\mathrm{B}(i,j)$ and the fused image $\mathrm{F}(i,j)$ which is based on computation of joint entropy and marginal entropy of as under:

$$(FMI)_{\mathrm{F}}^{\mathrm{AB}} = MI(\mathrm{A},\mathrm{F}) + MI(\mathrm{B},\mathrm{F}) \tag{7.22}$$

$$MI(\mathrm{A},\mathrm{F}) = \sum_{i=1}^{M} \sum_{j=1}^{N} h_{\mathrm{AF}}(i,j) \log_2 \left(\frac{h_{\mathrm{AF}}(i,j)}{h_{\mathrm{A}}(i,j) h_{\mathrm{F}}(i,j)} \right) \tag{7.23}$$

$$MI(\mathrm{B},\mathrm{F}) = \sum_{i=1}^{M} \sum_{j=1}^{N} h_{\mathrm{BF}}(i,j) \log_2 \left(\frac{h_{\mathrm{BF}}(i,j)}{h_{\mathrm{B}}(i,j) h_{\mathrm{F}}(i,j)} \right) \tag{7.24}$$

   where, $h_{\mathrm{AF}}(i,j)$ is the normalized joint histogram of images $\mathrm{A}(i,j)$ and $\mathrm{F}(i,j)$; $h_{\mathrm{A}}(i,j)$, $h_{\mathrm{B}}(i,j)$, and $h_{\mathrm{F}}(i,j)$ are the normalized marginal histogram of images **A**, **B**, and **F**, respectively.

   The fused mutual information (FMI) computed as per Equation 7.22 make use of two joint entropies computed at different scales. We use normalized fusion mutual information (NFMI) as suggested by Hossny et al. [49] as per Equation 7.25.

$$(NFMI)_{\mathrm{F}}^{\mathrm{AB}} = 2 \left[\frac{MI(\mathrm{A},\mathrm{F})}{H(\mathrm{A}) + H(\mathrm{F})} + \frac{MI(\mathrm{B},\mathrm{F})}{H(\mathrm{B}) + H(\mathrm{F})} \right] \tag{7.25}$$

   where, $MI(\mathrm{A},\mathrm{F})$ and $MI(\mathrm{B},\mathrm{F})$ are defined as per Equations 7.23 and 7.24,

respectively; $H(\mathrm{A})$, $H(\mathrm{B})$, and $H(\mathrm{F})$ are the marginal entropies of A, B, and F, respectively.

7.3 Evaluation

7.3.1 Correctness

The computations are expected to follow the IEEE standard for floating-point arithmetic (IEEE 754-2008) on a standard x86 system. Quantitative evaluation of the quality of fused image using any image fusion algorithm will be done using some or all of the quantitative evaluation metrics mentioned in previous section.

7.3.2 Performance and Power

The unit of work is one pixel in the fused image. Consequently, there are $(M \times N)$ units of work; corresponding to number of rows M and number of columns N in the fused image.

Performance is evaluated with time per units of work.

Power is evaluated with Watts per unit of work.

7.4 Appendix

An image fusion scheme based on discrete wavelet transform (DWT) utilizing multi-level image decomposition is shown in Figure 7.2.

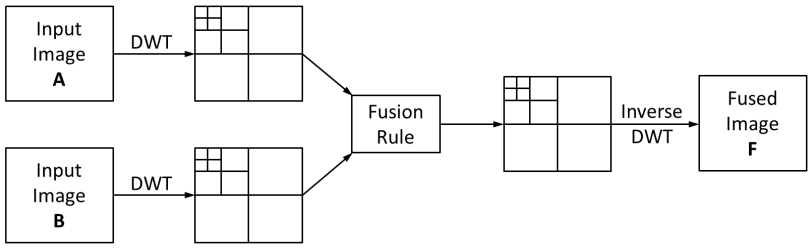


Figure 7.2: Discrete wavelet transform (DTW) image fusion scheme with multi-level image decomposition.

A DWT based image fusion scheme is implementated as per steps mentioned below:

1. A two dimensional discrete wavelet transform (DWT), $f(x, y)$ is obtained by applying DWT on an image over pixel rows and columns.

2. DWT coefficients are computed by convolving the image with low-pass filter (h_k) and high-pass filter (g_k) and performing down sampling by 2 over each row and column. This processing generates four subbands, LL,

HL, LH, and HH obtained via combination of h_k and g_k. These subband images contain the approximation coefficients convolved with low-pass filter and/or high-pass filter.

Here, LL denotes the subband that contains the approximation coefficients convolved with the low-pass filter h_k; LH denotes the horizontal detail coefficients from the original image; HL denotes the vertical detail coefficients with respect to high frequency components in the rows and low frequency in the columns; HH subband stores the diagonal detail information which is related to sharp changes in the image.

In a multi-level decomposition scheme, subsequent decomposition is performed only on the LL image.

3. A suitable fusion scheme is applied to the decompositions of two or more input images generated over multiple levels, to obtain a fused-image-decomposition. This is achieved by choosing an appropriate scheme to combine the coefficients obtained from different input images.

4. Inverse DWT is applied to the fused-image-decomposition, to generate a fused image.

Constraining Problem 8

Image: Face Detection

Nitin A. Gawande (PNNL), Joseph B. Manzano (PNNL), Nathan R. Tallent (PNNL)

8.1 Justification

Face recognition from set of images or stream of video images is an important procedure in biometric analytics and face recognition in particular. In the entire process of face recognition, face/object detection is the first step. Next setup is the extraction of features from detected facial objects. A suitable training procedure is then employed using extracted features for classification of a large number of images. Based on the classification of a large number of images, face recognition on an unknown image is then performed. Face detection is the first and critical step in the image recognition pipeline. Face detection is often performed in real-time. However, the training procedure which uses extracted features from images to perform classification, can be done offline. Other important applications of face detection include video surveillance and intelligent human computer interactions [53].

The face-detection throughput can vary with the pixel-size of input images, complexity of detector (algorithm), and the computing hardware. While the input image can be rescaled to a smaller size for faster processing however this may conversely affect detection of face. There are a number of challenges to face detection [123] such as; pose, which can be directly facing the camera or at an angle; lighting and face background; presence or absence of additional facial structural components like beards, mustaches, and wearable lenses; occlusion due to surrounding objects or clothing.

With regards to implementation of face detection algorithms, there are two main drawbacks [38]: (1) low processing throughput, and (2) lack of accurate face detection performance. Video streaming requires a throughput of more than 25 to 30 frames per second [2], [79]. Each frame may consist of frames with 720×480 or more pixels. Face detection is required to be processed for such large frame sizes at more than 30 frames per second with real time processing.

Given the requirement of high through put with constraints of real time processing and a need for higher detection rates, we define face detection as one of the most constraining problems in the wide domain of image processing.

8.2 Description

There are several approaches to solve the face detection problem. These approaches are broadly classified in to following categories as described by Yang et

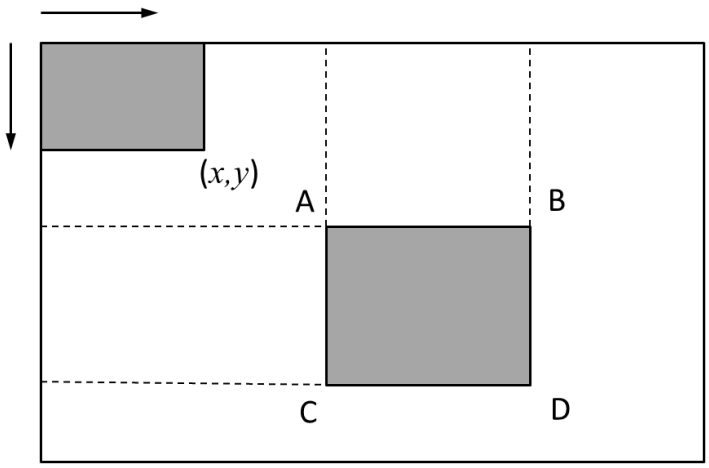


Figure 8.1: Illustration of integral image used in Viola and Jones algorithm [111].

al. [123]: 1) knowledge-based methods; 2) feature invariant approaches; 3) template matching methods; and 4) appearance-based method. The face detection algorithm proposed by Viola and Jones [111] was among the earlier approaches for implementation in real time with improved detection rates. This algorithm made use of rectangular features to build a classifier to select a small number of important features. In order to compute these features, this algorithm made use of integral image representation of input images. The detection method is based on a cascade of classifiers that are applied such that it significantly improves the performance of detection process. As an example of a typical real-time face detection algorithm, the one by Viola and Jones [111] is summarized below. However, any face detection algorithm or novel combination of different algorithms can be used to provide solution to this constraining problem.

Viola and Jones algorithm [111] makes use of integral image to compute simple Haar like rectangular features. Integral image or summed area table is an algorithm used for efficiently generating the sum of values for cells (pixels) in a rectangular grid. The integral image at location x, y in a given image contains the sum of the pixels above and to the left of x, y, including the pixel value at x, y. Integral image at location x, y shown in Figure 8.1 is computed as per Equation 8.1.

$$ii(x, y) = \sum_{x' \leq x, y' \leq y} i(x^{'}, y^{'}) \tag{8.1}$$

Where, $i(x^{'}, y^{'})$ and $ii(x, y)$ correspond to original image and integral image at pixel location (x, y), respectively. The sum of pixels in the region $ABCD$, shown in Figure 8.1 is calculated as per Equation 8.2.

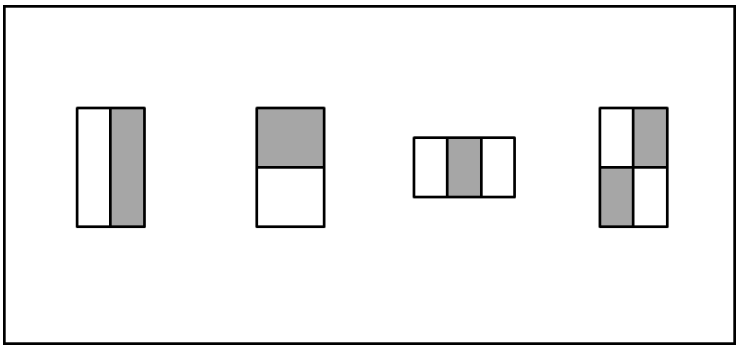


Figure 8.2: Simple Haar-like rectangle features used in Viola and Jones algorithm [111].

$$\sum_{(x,y)\in ABCD} i(x,y) = ii(D) + ii(A) - ii(B) - ii(C) \tag{8.2}$$

Using the integral image, computations are performed for simple Haar like reactangular features, shown in Figure 8.2. These computations include the weighted intensity difference between the two or more rectangular regions, when superimposed on a given input image. The Haar like features shown in Figure 8.2 are rescaled according to the size of window of input image.

There can be extremely large number of rectangle features associated with each image sub-window of an input image. Computing over extremely large number of features becomes expensive. Therefore Viola and Jones made use of AdaBoost, a boosting algorithm to select features and combine them to form an effective classifier. The process of classifying a sub-window region of input image is carried out as a decision tree or in a cascade structure.

Each sub-window of input image is processed by a sequence of clasifiers, with each classifier being more complex than the previous. If a classifier rejects a sub-window then no further processing is performed. A cascade structure of their face detector makes it possible to build boosted classifiers which reject most of the negative sub-windows. The number of weak classifiers and the decision threshold for early rejection at each node in the decision tree are specified. Learning with the use of a suitable boosting algorithm and using a set of features, a classifier is constructed which yields to final detection of face in an input image.

8.2.1 Input

The input to the face detection problem consists of following:

1. *Labeled dataset of images* The input image dataset consists of images available through open source image databases. We labeled images from the multiple encounter dataset (MEDS) [52] which has $\approx$ 1300 images. The label information is included in two text files that provide lists for the

name of image-file, image size as number of rows and columns of pixels, number of faces in this image and the coordinates of the bounding box for these faces. In order to label MEDS images we made use of a few state of the art face-detectors. Where the face was not detected using one or more face-detectors, the bounding box for faces was marked manually.

2. *Detection accuracy* The face detection rate depends on the allowable false positives. An allowable false positive rate is provided as input to the face detection problem.

8.2.2 Output

Without accurate face and facial-feature location, a noticeable degradation in the performance of face-recognition is observed. Face detection is considered successful if a presence and a rough location of a face is correctly identified [128].

Output from face detector is: 1) An integer equal to the number of faces detected in an input image; and 2) coordinates of rectangular region to annotate where a face is detected in an input image. The coordinates of the rectangle are in terms of pixel rows and columns.

To obtain a score for matching detections using a face detector, we make use of two scores as described by Jain and Miller [54]. They made use of receiver operating characteristics (ROC) curves. The inference drawn from the ROC curve is equivalent to non-parametric statistical hypethesis test. Unlike the receiver operating characteristics (ROC) curves, we provide a threshold for maximum false positives and accordingly evaluate detection rates for different face detectors. We make use of two separate metrics, discrete score and continuous score. The discrete score holds a value of either 0 or 1 while the continuous score is the ratio of intersected area of detection to joined area as described by Jain and Miller [54]. A face detection algorithm is evaluated based on true positive detections, the two scores (discrete and continuous) obtained for all detections with a given false positive detection rate.

8.3 Evaluation

8.3.1 Correctness

A given solution to face detection problem is evaluated for correctness on the basis of detection accuracy achieved with allowable false positives. Both parameters, detection accuracy and false positives are the percentages of input image dataset. The computations are expected to follow the IEEE standard for floating-point arithmetic (IEEE 754-2008) on a standard x86 system.

8.3.2 Performance and Power

The unit of work is equal to one pixel of input image. Consequently, there are $(M \times N)$ units of work; corresponding to number of rows M and number of

columns N of an input image. The performance of solution to face detection is measured in terms of time. Power and performance numbers are average of measurements over a large number of images in a given image dataset.

Performance is evaluated with time per units of work.

Power is evaluated with Watts per unit of work.

Constraining Problem 9

Image: Text Image Classification

Seunghwa Kang (PNNL), Joseph B. Manzano (PNNL), Nathan R. Tallent (PNNL)

9.1 Justification

This constraining problem solves a classification problem with supervised learning taking text images as input data. A classification problem is to assign a class label to every input item. For example, given a handwritten image of a single digit ranging from 0 to 9, assigning a correct label (0, 1, 2, 3, ... , or 9) to the image is a classification problem. In *supervised* learning, a training set — sample input items and the correct labels — is provided in addition to a set of items to label; *unsupervised* learning attempts to learn the structure of the input items without requiring correct labels. In this constraining problem, we focus on classifying text images: handwritten digits, texts in shop signboards, street house number images, and other text or artificial symbol images.

Text image classification is relevant in multiple military applications: promoting situation awareness in complex battlefields through portable augmented reality devices [122, 35], digitizing military documents [28, 27, 109], and developing a digital user interface for battle planning [39, 10]. Situation awareness is about recognizing the surrounding environment, understanding the current situation, and predicting changes in the near future [36]. Situation awareness is critically important in decision making in complex battlegrounds (*e.g.*, urban areas), and augmented reality devices have been proposed to aid military personnel to better comprehend the surrounding situation. Supported by the DARPA Ultra-Vis program, Applied Research Associates and BAE Systems have developed a wearable augmented reality system (*iLeader*) to assist soldiers on various military tasks such as locating friendly forces and enemies and identifying military targets [5]. Naval research lab has developed the battle field augmented reality system (BARS) using wearable devices [55]. Comprehending the surrounding environment is even more challenging in foreign areas. Augmented reality devices that detect and translate foreign signs and texts can assist military forces on decision making via promoting situation awareness [122, 35]; converting text images to digital text data is a key step in the computational pipeline.

Text image classification is also relevant in digitizing damaged military documents [28], handwritten annotations in military forms [27], and old handwritten military registers [109]. Military commanders use courses of actions (COA) diagrams to develop and share battle plans, and a digital pen based COA user interface, such as *nuSketch Battlespace*, facilitates automation of battle plan development and communication [39]. Courses of actions diagrams contain both shapes and texts [10], and recognizing such shapes and texts is an important

computational task.

We adopt the MNIST handwritten digit database and the street view house numbers (SVHN) dataset as proxy data to represent various text and artificial symbol images in the above applications. The MNIST database [60] has images of handwritten digits (from 0 to 9) and the corresponding labels. The database has 60,000 images and labels to train machine learning models. The database provides additional 10,000 images and labels to test trained models. A trained model attempts to correctly label the 10,000 test images. The MNIST database has been widely used to compare various machine learning models. The current best performing model [25] assigns incorrect labels to only 0.23% of the entire test set images. We selected the MNIST database considering that recognizing handwritten digits in the database resembles recognizing handwritten texts in military documents and courses of actions diagrams.

The street view house numbers (SVHN) dataset [80] has images of numbers embedded in natural backgrounds. The SVHN database is composed of house number images extracted from Google street view data. The SVHN database has 600,000 labeled images, which is 10 times more images than the MNIST database. Lee *et al.* [62] labeled test images from the SVHN dataset achieving the error rate of 1.92%. We adopted the SVHN database due to the similarity between recognizing house digits embedded in natural scenes and recognizing foreign characters or signs in signboards.

9.2 Description

9.2.1 Input

The MNIST homepage (`http://yann.lecun.com/exdb/mnist/`) provides the following four files.

- *train-images-idx3-ubyte.gz*: 60,000 images for training.
- *train-labels-idx1-ubyte.gz*: 60,000 labels for the training images (one label per image).
- *t10k-images-idx3-ubyte.gz*: 10,000 images for testing.
- *t10k-labels-idx1-ubyte.gz*: 10,000 labels for the testing images (one label per image).

A single image has 28×28 pixels, each pixel occupying one byte to represent a gray scale color (0 for white and 255 for black). A single label is one byte in size holding an integer value ranging from 0 to 9. An image data file starts with four 32 bit integers (big-endian) for the magic number (2051), number of images (60,000 for the training file and 10,000 for the test file), number of rows (28), and number of columns (28), respectively. The remaining part of the file holds pixel data. A label data file starts with two 32 bit integers (big-endian) for the magic number (2049) and the number of labels (60,000 for

the training file and 10,000 for the test file), respectively. The remaining part of the file stores label data. Refer the MNIST homepage for further information. *train-images-idx3-ubyte.gz* and *train-labels-idx1-ubyte.gz* are used for the training step, and *t10k-images-idx3-ubyte.gz* and *t10k-labels-idx1-ubyte.gz* are used for the testing step.

The Street View House Numbers (SVHN) dataset (`http://http://ufldl.stanford.edu/housenumbers/`) provides over 600,000 labeled images in two different formats: original full sized house number images with bounding boxes for digit characters (one bounding box for one character) and a set of 32×32 pixel images (each image contains a single digit character at the center). We use the latter format in this constraining problem to minimize pre-processing efforts. The SVHN dataset provides the following three files.

- *train_32x32.mat*: 73,257 color images and labels for training.
- *test_32x32.mat*: 26,032 color images and labels for testing.
- *extra_32x32.mat*: additional 531,131 color images and labels for training—these images are less challenging to automatically label than the images in the default training set (having 73,257 images).

These files are stored using the MAT-file format; one can load the files using MATLAB. Each file contains one matrix containing the entire set of images and one vector containing the correct labels for the images in the matrix. Refer the SVHN homepage for additional details. This constraining problem uses *train_32x32.mat* and *extra_32x32.mat* for training and *test_32x32.mat* for testing.

To evaluate submitted solutions, test images may be slightly distorted (rotated, scaled, horizontally and vertically shifted, and contrast and brightness adjusted) to test the robustness of a solution and penalize overfitted solutions.

9.2.2 Output

A trained model should output one class label per one test image. For the MNIST database, class labels 0, 1, 2, 3, 4, 5, 6, 7, 8, and 9 are assigned for digits 0, 1, 2, 3, 4, 5, 6, 7, 8, and 9, respectively. For the SVHN dataset, class labels 0, 1, 2, 3, 4, 5, 6, 7, 8, and 9 are assigned for digits 0, 1, 2, 3, 4, 5, 6, 7, 8, and 9, respectively.

9.3 Evaluation

9.3.1 Correctness

The error rate of a trained model is the percentage of misclassified images (a model output label does not coincide the correct label included in the dataset) over the entire of test images.

9.3.2 Performance and Power

Execution time is separately measured for training and testing. Power consumption is separately measured for training and testing as well. Execution time and power consumption for the testing step will be normalized to the number of test images. The overall performance of a classification model and its implementation will be evaluated based on the error rate (lower is better), execution time for training, power consumption for training, normalized execution time for testing, and normalized power consumption for testing. Different weights will be assigned to different metrics based on varying requirements in military sites; in general, execution time and power consumption for training are less important as training can be performed off-site using cluster computers while testing (and actual classification of texts and symbols) will more likely to be performed using embedded systems (e.g., wearable devices such as glasses).

Constraining Problem 10

Image: Natural Image Classification

Seunghwa Kang (PNNL), Joseph B. Manzano (PNNL), Nitin A. Gawande (PNNL), Nathan R. Tallent (PNNL)

10.1 Justification

This constraining problem solves a classification problem with supervised and semi-supervised learning taking natural images as input data. A classification problem is to assign a class label to an input item. For example, given images of multiple military vehicle models, finding the correct model name for a vehicle image is a classification problem. In supervised learning, a training set, sample input items and the correct labels, is provided in addition to a set of items to label—unsupervised learning attempts to learn the structure of the input items without requiring correct labels. In semi-supervised learning, only a subset of the input items in a training set are labeled; note that obtaining labeled data is often labor intensive and significantly more expensive than obtaining unlabeled data, and the Defense Advanced Research Projects Agency (DARPA) also emphasized the necessity to develop a semi-supervised learning platform [30].

In this constraining problem, we focus on classifying natural images: photos of various objects such as military vehicles. Natural image classification is relevant in multiple military applications: processing wide area motion imagery (WAMI) data, synthetic aperture radar (SAR) images, and video streams from typical narrow view cameras for surveillance and reconnaissance. WAMI sensors are attached on flying vehicles and produce image streams of wide areas; analyzing WAMI data is a significant technical challenge [92]. A SAR device, placed on a moving airplane or satellite, emulates the behavior of an extremely wide aperture radar by computationally processing signals collected along the device's forward motion [21]. Surveillance cameras, whether fixed or mounted on ground vehicles, also produce a large volume of streaming videos. It is impractical to manually monitor multiple streams of high resolution images from these devices. Data mining techniques are often applied to automatically detect important events, and image classification is an important step in the computational pipeline—such as identifying different models of military vehicles [108], distinguishing vehicles from background objects or other distractors [67, 103], and classifying land cover types (sea, park, and urban areas) [6, 3].

We adopt the MSTAR (Moving and Stationary Target Acquisition and Recognition) dataset, the NORB dataset, and the STL-10 dataset as proxy data to represent various natural image classification challenges in the above applications. The MSTAR dataset [99] has SAR images of military and civilian vehicles, a stationary structure (SLICY), and rural and urban background scenes (clutter). Images in the dataset are collected using different angles between a SAR

device and an object (or using different depression angles in SAR terminology). The dataset has SAR images having varying numbers of pixels stored in multiple compressed files. Typically, only a subset of the dataset is used for different purposes. Similar to [115, 84, 120, 50], this constraining problem uses 128 pixel $\times$ 128 pixel images of three military vehicle models: T72, BMP2, and BTR70. T72 and BMP2 in this subset has three variants (with three different serial numbers), and BTR70 has only one variant. Images in the subset are collected using 17 degree and 15 degree depression angles. 1622 images (used for training) are collected using a 17 degree depression angle, and 1365 images (used for testing) are collected using a 15 degree depression angle. Researchers often use images belonging to only one variant per model for training to demonstrate the performance of their classification algorithm on unseen variants [115, 120]. This constraining problem follows this practice; this reduces the training set size to 698 images. The MSTAR dataset is relatively small and old compared to other widely used machine learning datasets, but this constraining problem includes this dataset considering its direct relevance to military applications.

The NORB dataset [61] contains post-processed images of 50 toy instances belonging to 5 classes (four legged animals, human figures, airplanes, trucks, and cars)—10 toy instances per class. Each instance is photographed under 9 different angles between a camera and a toy instance, 18 different azimuths, and 6 different lighting conditions. Post-processing distorts (rotates, scales, horizontally and vertically shits, and increases or decreases brightness and contrast) images; superposes images on complex backgrounds; and adds distracting objects near image boundaries. This dataset is created to test classification algorithms' invariant properties; the above mentioned post-processing does not change the correct class label of a processed image and a classification algorithm should be able to find the correct labels for distorted images. The dataset also contains images without main object; images in this class has only backgrounds and distracting objects. A classification algorithm should refuse to map an image with no main object to one of the 5 classes; or should map the image to the "blank" class. The NORB dataset provides 583,200 labeled images for training and 116,640 labeled images for testing. Cireşan and his collaborators labeled this dataset with the 2.7% error rate using a classification algorithm based on multi-column deep neural networks [25]. This dataset is adopted owing to the dataset's relevance in classifying military object photos taken under different conditions and having different backgrounds and distracting objects.

The STL-10 dataset [26] is composed of a small number of labeled images and a large number of unlabeled images for semi-supervised learning. The dataset has labeled images belonging to 10 classes: airplane, bird, car, cat, dear, dog, horse, monkey, ship, and truck. The dataset provides 500 images per class and 800 images per class for training and testing, respectively. The dataset also provides 100,000 unlabeled images. The unlabeled dataset is composed of images belonging to the 10 classes and images outside the 10 classes (such as images of bears, rabbits, trains, and buses). Each image has 96 $\times$ 96 pixels. Images in this dataset were acquired from the ImageNet database [32]. Swersky, Snoek, and Adams reported the accuracy of 70.1% in classifying this dataset [106]. We

selected this dataset considering the importance of semi-supervised learning in military applications; the Defense Advanced Research Projects Agency (DARPA) published a broad agency announcement to develop a semi-supervised learning platform based on deep learning architectures [30].

10.2 Description

10.2.1 Input

The MSTAR dataset is available to download from the following url: `https://www.sdms.afrl.af.mil/index.php?collection=mstar`. The MSTAR dataset includes multiple compressed files. This constraining problem uses the *MSTAR-PublicTargetChips-T72-BMP2-BTR70-SLICY.zip* file. This file has image chips of three vehicle models (T72, BMP2, and BTR70). Table 10.1 summarizes the numbers of images used for training and testing for different models and variants. For training, images belonging to the second and third serial numbers (SN_812 and SN_S7 for T72 and SN_9566 and SN_C21 for BMP2) should not be used; this tests the performance of a classification algorithm on unseen variants [115, 120].

model	T72			BMP2			BTR70
serial no.	SN_132	SN_812	SN_S7	SN_9563	SN_9566	SN_C21	SN_C71
training (17 degree)	232	(231)	(228)	233	(232)	(233)	233
testing (15 degree)	196	195	191	195	196	196	196

Table 10.1: This table summarizes the numbers of images for different vehicle models and serial numbers. Images collected using a 17 degree depression angle are used for training. Images collected using a 15 degree depression angle are used for testing. For training, images belonging to only one serial number are used for each vehicle model (the numbers inside parentheses denote the numbers of images for the variants excluded for training).

Decompressing the *MSTAR-PublicTargetChips-T72-BMP2-BTR70-SLICY.zip* file produces multiple MSTAR image files, one file per image. Each file has a variable length text header followed by two blocks of $128 \times 128 \times 4$ (single-precision big-endian floating point number) bytes; the first block stores magnitude data and the second block stores phase data (phase data can be used to enhance the image quality). The header in a file stores detailed information about the image (the header size, image size, depression angle, model name, serial number, and other auxiliary information). A software tool to convert an MSTAR file to a JPEG or TIFF image file is available form `https://www.sdms.afrl.af.mil/index.php?collection=tools_mstar` as well. Refer to the MSTAR homepage for additional details.

The NORB dataset (`http://www.cs.nyu.edu/~ylclab/data/norb-v1.0/`) provides 10 <"-dat" file, "-cat" file, "-info" file> triplets for training and 2 triplets

for testing.

- *norb-5x46789x9x18x6x2x108x108-training-(01,02,03,04,05,06,07,08,09, or 10)-dat.mat.gz*: 10 "-dat" files for training. Each file stores a header followed by a 29,160 $\times$ 2 $\times$ 108 $\times$ 108 matrix. This file stores 29,160 image pairs (6 classes $\times$ 5 instances $\times$ 9 elevations $\times$ 18 azimuths) of 108 $\times$ 108 pixels. Each element in a matrix is a 1 byte integer storing a pixel value.

- *norb-5x46789x9x18x6x2x108x108-training-(01,02,03,04,05,06,07,08,09, or 10)-cat.mat.gz*: 10 "-cat" files for training. Each file stores a header followed by a 29,160 dimensional vector. Each vector element is a 4 byte integer storing the correct label for the corresponding image pair.

- *norb-5x46789x9x18x6x2x108x108-training-(01,02,03,04,05,06,07,08,09, or 10)-info.mat.gz*: 10 "-info" files for training. Each files stores a header followed by a 29,160 $\times$ 10 matrix. A "-info" file provides 10 additional attributes per image (such as the instance number within the class, camera elevation, azimuth, and other auxiliary information) for 29,160 images. Each attribute is a 4 byte integer.

- *norb-5x01235x9x18x6x2x108x108-testing-(01 or 02)-dat.mat.gz*: 2 "-dat" files for testing. The file format is identical to the "-dat" files used for training.

- *norb-5x01235x9x18x6x2x108x108-testing-(01 or 02)-cat.mat.gz*: 2 "-cat" files for testing. The file format is identical to the "-cat" files used for training.

- *norb-5x01235x9x18x6x2x108x108-testing-(01 or 02)-info.mat.gz*: 2 "-info" files for testing. The file format is identical to the "-info" files used for training.

A header file has two 4 byte integers storing a magic number and the dimensionality of the matrix followed by 3 or more integers storing the size of the matrix in each dimension. A magic number specifies the type of matrix elements (*e.g.* single-precision floating point number, double-precision floating point number, one byte integer, and four byte integer). A vector is considered as an 1-dimensional matrix. If the dimensionality of the matrix is equal to or smaller than 3, 3 integers store the matrix size in each dimension; only the first N integers are valid for an N-dimensional matrix. If N is larger than 3, N integers store the matrix size in each dimension. According to the NORB homepage, only "-dat" and "-cat" files are used for typical training and testing tasks. This constraining problem also asks to use only "-dat" and "-cat" files assuming that such auxiliary data are not always available. See the NORB homepage for additional information about the files in the dataset.

The STL-10 homepage (`http://cs.stanford.edu/~acoates/stl10/`) provides 100,000 unlabeled images, 500 labeled images per class for training, and 800 labeled images per class for testing. Each image has three channels (R, G,

and B), and each channel has 96 × 96 pixels (one byte unsigned integer per pixel). The STL-10 homepage asks participants to make 10 trials and report the average error rate. In each trial, a classification model is trained using the entire set of unlabeled images and only a predefined subset of the labeled training images. 10 folds are provided as well—each fold contains the indices of the labeled images to be used in each trial. The entire set of labeled testing images are used for testing in every trial. The STL-10 dataset provides MATLAB files and binary files; users can pick either MATLAB files or binary files. The STL-10 dataset provides the following MATLAB files.

- *train.mat*: This file has the following variables: "X", "y", "class_names" and "fold_indices". The matrix "X" contains 1 image per row. A single row stores a 2-dimensional image in the column-major order for the R, G, B channels (the R channel data comes first, followed by the G and B channel data). The vector "y" contains labels, one label per image. The "class_names" variable contains class names. The "fold_indices" variable contains the indices for the subset of the labeled images to be used in each trial. "fold_indices{i}" provides the indices to be used for the i^{th} trial.
- *test.mat*: This file has the following variables: "X", "y", and "class_names". The format of each variable is identical to the variables in the *train.mat* file.
- *unlabeled.mat*: This file has the following variables: "X" and "class_names". The format of each variable is identical to the variables in the *train.mat* file.

The STL-10 dataset also provides the following binary files as well.

- *train_X.bin*: This file stores training images.
- *train_y.bin*: This file stores labels for the training images stored in the *train_X.bin* file.
- *test_X.bin*: This file store testing images.
- *test_y.bin*: This file stores labels for the testing images stored in the *test_X.bin* file.
- *unlabeled.bin*: This file stores additional unlabeled images.
- *class_names.txt*: This file stores class names.
- *fold_indices.txt*: This file stores the indices for the subsets to be used in 10 trials.

Refer to the STl-10 homepage for additional details.

To evaluate submitted solutions, test images may be slightly distorted (rotated, scaled, horizontally and vertically shifted, and contrast and brightness adjusted) to test the robustness of a solution and penalize overfitted solutions.

10.2.2 Output

A trained model should output one class label per one test image. For the MSTAR database, class labels 0, 1, and 2 are assigned for vehicle models T72, BMP2 and BTR70, respectively. For the NORB dataset, class labels 0, 1, 2, 3, 4, and 5 are assigned for animal, human, plane, truck, car, and blank (which has no main object), respectively. For the STL-10 dataset, class labels 0, 1, 2, 3, 4, 5, 6, 7, 8, and 9 are assigned for airplane, bird, car, cat, deer, dog, horse, monkey, ship, and truck, respectively.

10.3 Evaluation

10.3.1 Correctness

The error rate of a trained model is the percentage of the number of misclassified images (a model output label does not coincide the correct label included in the dataset) over the number of test images.

10.3.2 Performance and Power

Execution time is separately measured for training and testing. Power consumption is separately measured for training and testing as well. Execution time and power consumption for the testing step will be normalized for the number of test images. The overall performance of a classification model and its implementation will be evaluated based on the error rate (lower is better), execution time for training, power consumption for training, normalized execution time for testing, and normalized power consumption for testing. Different weights will be assigned to different metrics based on varying requirements in military sites; in general, execution time and power consumption for training are less important as training can be performed off-site using cluster computers while testing (and actual classification of images) will more likely to be performed using embedded systems attached to military vehicles carrying radars, sensors, or cameras.

Constraining Problem 11

Hyperspectral Image: Signature Extraction

Jonathan D. Suter (PNNL), Nitin A. Gawande (PNNL), Joseph B. Manzano (PNNL), Nathan R. Tallent (PNNL)

11.1 Justification

Hyperspectral imaging (HSI) is a data-rich tool that acquires a large number of images of a scene or object, each capturing the irradiance values at a different wavelength band [42]. A full set of these images represents a three dimensional array, with two spatial dimensions (the imaging field of view) and a wavelength or frequency dimension. This three dimensional array is typically referred to as a *hypercube*. A hypercube offers an added degree of utility for spectroscopic applications, for which spectra can now be registered to specific locations. For the most part, HSI applications use the fingerprinting capability of optical spectroscopy to identify substances and chemicals within a field of view. The high spectral resolution of modern HSI cameras makes it possible to identify chemical species with a greater degree of confidence than ever before.

In practice, the collection of data corresponding to a hypercube requires a HSI camera and a light source. Light from the source interacts with the materials or objects of interest and is transmitted, scattered, reflected, or re-emitted into the lens of the HSI camera, which then uses one of several technologies (dispersive prism, Fourier transform, etc.) to record the spatially-and-spectrally resolved information.

The applications being pursued for HSI are as diverse as the applications for optical spectroscopies in general. Although, most common applications seem to be variations of remote sensing, near-field and forensic applications have been explored with great success. The most popular applications for HSI include industrial process monitoring [51], geological mineralogy [110], agriculture [72], astronomy [95], chemical imaging [37], aerial surveillance [124], and even medical applications [70]. The wavelengths of interest for these applications can include anything from the ultraviolet to the longwave infrared, are dictated by the spectral features that the user is interested in, and necessarily inform the type of HSI camera employed in a given study.

In order to extract useful data from a hypercube, a large amount of data processing is often required. The three dimensional nature of the hypercube is both a blessing and a curse, because it accommodates a wealth of data, and yet is very difficult to view without rigorous processing. A number of steps are involved in processing, and these will vary depending on the application and depending on what the user knows about the materials within the field of view. Regardless of application, the first step in the data processing sequence should entail normalizing the spectral radiance values in the hypercube relative

to the spectral radiance of the incident light. The normalization curve could be the solar spectrum or it could be the spectral intensity of a man-made source like a laser, bulb, or LED. Once a normalized hypercube has been generated, the subsequent actions depend largely on what the analyst knows about the materials and surfaces within the hypercube field of view. If library spectra are available for known or suspected chemical species, then those can be searched first. By "library" spectra, we refer to reflection spectra that have been acquired in a controlled laboratory setting for known chemicals. Simple linear unmixing algorithms like least squared analysis can be used to identify spatial pixels that strongly match the library spectra. One of the weaknesses of library spectrum fitting is that the controlled laboratory settings under which the library spectrum was acquired may present the chemical in a state that is not realistic compared to the states or morphologies that exist under field conditions. Therefore, it is likely that additional analysis will have to be undertaken to deconvolve and interpret the remaining component spectra. These types of subsequent processing are typically referred to as *endmember extraction* or *signature extraction*.

Endmembers are defined as the full set of spectra which can be linearly combined to explain all of the spectral curves within a hypercube [14]. Library spectra fit within this broad definition of "endmember," but there are almost always many unsuspected components leftover after the anticipated spectra are accounted for. Often, these endmembers will represent signatures from interference or background compounds, but sometimes the most interesting features in a hypercube lie within the unknown endmember spectra. Due to the unknown aspect of endmembers, extraction can be a significant challenge and it has been the focus of a great deal of research effort in recent decades [125]. Researchers are currently tackling the problem from two simultaneous directions. One is by exploring the fundamental physics of how light and matter interacts in order to develop sophisticated predictive models. On the other side, exhaustive efforts to boost the effectiveness of extraction algorithms have also improved our ability to extract and classify endmembers in terms of distribution and importance. Many different algorithms have been explored, and side-by-side comparisons between them are not always conveniently available. Therefore, improvement and better characterization of these algorithms remains an urgent and unfinished scientific task. We therefore describe endmember or signature extraction as one of the most constraining problem in the domain of hyperspectral image processing.

11.2 Description

11.2.1 Input

The input dataset is a hyperspectral scene called *hypercube* shown in figure 11.1. The input data set has images consisting of M number of pixel rows, N number of pixel columns, and L number of bands corresponding to different wavelengths. An input generator is made available for any user to make a custom hypercube

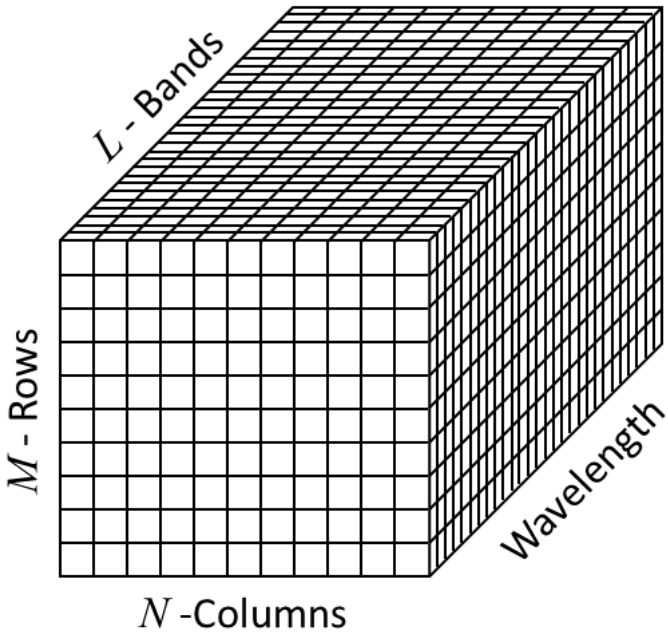


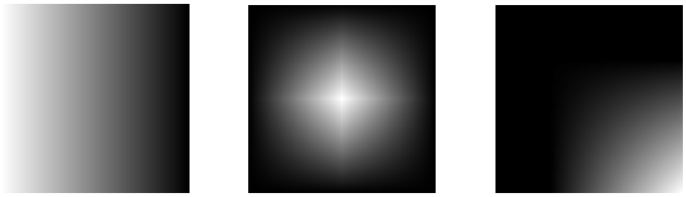
Figure 11.1: Hyperspectral image data cube (hypercube).

for this constraining problem.

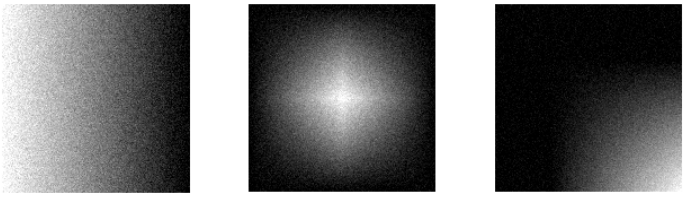
Input Generator

The input generator is a software tool which enables making a hypercube with variable size and hyperspectral scene complexity. A user of this tool can choose size of the hyperspectral imaging scene. While any variable size can be used, default image sizes of 256 $\times$ 256 pixels, 1024 $\times$ 1024, and 2048 $\times$ 2048 are provided corresponding to small, medium, and large cases, respectively. The next input to the input generator is information on spectra which is used to construct hyperspectral scene. The input generator includes information on a large number of spectra from the ASTER Spectral Library Version 2.0 [8]. Any other spectrum of interest can also be included in addition to already included spectra. The format of these spectra is ASCII with information in two columns corresponding to spectra wavelengths and percent reflectance or absorbance. The input generator makes use of library spectra to make different spectrum regions shown as in Figure 11.2. The extent of spread of these regions can be changed and the orientation can be altered by rotation with a 90° increment. Figure 11.2 correspond to individual spectrum scene with a total L number of bands that can be included in the hypercube. The input generator provides a choice to the users to select specific portions from the input library spectra.

Different disjointed portions of spectra wavelenth can be included to form a total L number of bands.



(a)



(b)

Figure 11.2: Spectra regions: (a) Different geometric models (b) with noise.

Finally a hypercube is constructed by mixing different regions as shown in Figure 11.2. Each region used in mixing correspond to one spectrum. A variable number of spectra can be included in the construction of hypercube from library spectra. A linear mixture model as adopted by Plaza et al. [90] is used to create input hyperspectral scenes as per Equation 11.1. The contribution of ambient clutter and instrument noise is added by injecting random noise into the input hyperspectral scene data. Let $\mathbf{X}(i,j)$ be a vector containing discrete spectrum at pixel with spatial coordinates (i,j) in the hypercube. Then $\mathbf{X}(i,j)$ is denoted by Equation 11.1.

$$\mathbf{X}\left(i,j\right) = \left(\mathrm{SNR} + n\left(i,j\right)\right) \cdot \sum_{k=1}^{R} \alpha_k\left(i,j\right) \cdot \mathbf{r}_k \tag{11.1}$$

where, R is the total number of reference spectral signatures used to simulate the scene, $\alpha_k\left(i,j\right)$ is the assigned fractional abundance of spectral signature $\mathbf{r}_k$ at pixel (i,j). The signal to noise ratio (SNR) and noise factor $n\left(i,j\right)$ inject noise which is randomly distributed. The input value for SNR and the distribution model for n can be modified in the input generator.

11.2.2 Output

Endmember or signature extraction is the process of selecting a set of pure signature spectra of the materials present in a remotely sensed hyperspectral scene [102]. A mathematical representation of endmember with regards to hyperspectral imaging can be found elsewhere [133], [132], [22]. A simple description considering a linear spectral mixture model of endmembers is described here. In a hyperspectral imagery scene with L bands, a pixel in this imagery at discrete spatial coordinates (i, j) is represented by a vector $\mathbf{X}(i, j)$ as follows:

$$\mathbf{X}(i,j) = [x_1(i,j), x_2(i,j), \cdots, x_L(i,j)] \tag{11.2}$$

Considering a linear mixture model, each pixel vector in the original scene can be modeled as follows:

$$\mathbf{X}(i,j) = \sum_{k=1}^{p} \Phi_k(i,j) \cdot \mathbf{E}_k + \mathbf{n}(i,j) \tag{11.3}$$

where, $\mathbf{E}_k$ is the spectral response of endmember k, $\Phi_k(i,j)$ is a scalar value assigning the fractional abundance of endmember k at pixel $\mathbf{X}(i,j)$, p is the total number of endmembers, $\mathbf{n}(i,j)$ is a noise vector. The solution to the linear spectral mixture problem as defined by Equation 11.2 is successful estimation of p number of endmembers that are present in the input and the correct determination of a set of endmembers $\{\mathbf{E}_k\}_{k=1}^{p}$ and their corresponding abundance fractions $\{\mathbf{\Phi}_k(i,j)\}_{k=1}^{p}$ at each pixel $\mathbf{X}(i,j)$. The two constraints that are normally imposed on such model are: (a) abundance non-negativity constraint, $\mathbf{\Phi}_k(i,j) \geq 0$; and (b) abundance sum-to-one constraint, $\sum_{k=1}^{p} \mathbf{\Phi}_k(i,j) = 1$.

Solution to this signature extraction constraining problem can be obtained using any appropriate signature extraction algorithm, combination of different algorithms or an entirely novel implementation. There are several different algorithms to find endmembers that are pure signatures in a given hyperspectral data cube. Plaza and Chang [89] described a few different class of signature extraction algorithms and the challenges to these approaches. Described below are two popular algorithms out of several commonly used algorithms. The first algorithm is N-Finder algorithm (N-FINDR), which was originally developed by Winter [118]. We described the implementation of this algorithm given by Chang [22]. The second commonly used algorithm which is described here is the pixel purity index (PPI) algorithm by Boardman et al. [15].

N-Finder (NFINDR) algorithm

1. Given the datacube of size $(M \times N \times L)$, with image having M rows and N columns, and L spectral bands. There are p endmembers that are to be extracted.

2. Data size is reduced by applying a suitable dimensionality reduction transform (DRT) to reduce data dimensionality from L to $(p-1)$.

3. An initial set of endmembers that are randomly generated from the input data given by p sample vectors $\mathbf{e}_1, \mathbf{e}_2, \cdots \mathbf{e}_p$ form a p-vertex simplex $\mathbf{S}(\mathbf{e}_1, \mathbf{e}_2, \cdots \mathbf{e}_p)$ and has volume, $\mathbf{V}(\mathbf{e}_1, \mathbf{e}_2, \cdots \mathbf{e}_p)$.

$$\mathbf{V}(\mathbf{e}_1, \mathbf{e}_2, \cdots \mathbf{e}_p) = \frac{\left|\det \begin{bmatrix} 1 & 1 & \cdots & 1 \\ \mathbf{e}_1 & \mathbf{e}_2 & \cdots & \mathbf{e}_p \end{bmatrix}\right|}{(p-1)!} \tag{11.4}$$

4. Find a set of p sample vectors $\{\mathbf{e}_1^*, \mathbf{e}_2^*, \cdots \mathbf{e}_p^*\}$, that maximizes volume $\mathbf{V}$

$$\left\{\mathbf{e}_1^*, \mathbf{e}_2^*, \cdots \mathbf{e}_p^*\right\} = \arg\left\{\max \mathbf{V}\left(\mathbf{e}_1, \mathbf{e}_2, \cdots \mathbf{e}_p\right)\right\} \tag{11.5}$$

Pixel Purity Index (PPI) algorithm

The PPI implementation is performed in three steps: reduction to apparent surface reflectance; pixel purity determination; and partial unmixing [15]. However, this original implementation lacks full description and therefore a few variants of PPI algorithm are cited in literature [89], [89]. The PPI algorithm is implemented by performing following steps:

1. Use virtual dimensionality (VD) to determine the number of dimensions p that are required to be retained after dimensionality reduction. Apply the maximum noise fraction (MNF) or principal component analysis (PCA) transform to reduce dimensionality of the data set to p component images.

2. Randomly generate a set of K unit vectors, called *skewers*, $\{\mathbf{skewer}_j\}_{j=1}^{K}$. Where K is a presumed sufficiently large positive integer.

3. Project all data sample vectors $\mathbf{r}_i$ onto each skewer, $\mathbf{skewer}_j$ via dot product $(\mathbf{r}_i \cdot \mathbf{skewer}_j)$ to find extrema set for $\mathbf{skewer}_j$, denoted by $S_{extrema}\left(\mathbf{skewer}_j\right)$. An indicator function of data set $\mathbf{R}$, denoted by $I_S(\mathbf{x})$ represent membership of an element $\mathbf{x}$ to a particular set S as follows:

$$I_S(\mathbf{x}) = \begin{cases} 1, & \text{if } \mathbf{x} \in S \\ 0, & \text{if } \mathbf{x} \notin S \end{cases} \tag{11.6}$$

4. Calculate the PPI score N_{PPI} associated to each pixel vector $\mathbf{r}_i$ as follows:

$$N_{\mathrm{PPI}}(\mathrm{r}_i) = \sum_{j=1}^{K} I_{S_{extrema}}\left(\mathbf{skewer}_j\right)(\mathbf{r}_i) \tag{11.7}$$

5. Using an appropriate threshold value t_v find pixel vectors with scores of $N_{\mathrm{PPI}}(\mathrm{r}_i)$ above t_v and label them as spectral endmembers.

Metric

A given solution to endmember extraction problem is evaluated using a spectral angle mapper (SAM) metric [58]. The SAM metric computes the angle between any pixel vector and a reference spectrum. The reference spectrum can be at any other pixel or a spectrum from a spectra library. The calculation of spectral angle Mapper (SAM) as per Equation 11.8 makes this metric independent of length of the two vectors. This makes SAM insensitive to spectrum gain factors and therefore an apparent reflectance spectra can be directly compared with a laboratory spectra [58]. In this constraining problem, we use the SAM metric for a group of pixels using Equation 11.8 and comparison of group of pixels with a reference spectrum using Equation 11.8.

$$\mathrm{SAM}\left(\mathbf{X}\left(i,j\right),\mu\right)=\cos^{-1}\left(\frac{\langle\mathbf{X}\left(i,j\right),\mu\rangle}{\|\mathbf{X}\left(i,j\right)\|\|\mu\|}\right) \tag{11.8}$$

where,

$\langle\mathbf{X}\left(i,j\right),\mu\rangle=\sum_{k=1}^{L}\mathbf{X}\left(i,j\right)_k\mu_k$,

$\|\mathbf{X}\left(i,j\right)\|=\left(\sum_{k=1}^{L}\left(\mathbf{X}\left(i,j\right)_k\right)^2\right)^{1/2}$, and

$\|\mu\|=\left(\sum_{k=1}^{L}\left(\mu_k\right)^2\right)^{1/2}$,

μ is a reference spectrum and L is the total number of spectrum bands used for computing SAM,

A low SAM value corresponds to mean high spectral similarity between the two k-dimensional vectors $\mathbf{X}$ and μ.

11.3 Evaluation

11.3.1 Correctness

The computations are expected to follow the IEEE standard for floating-point arithmetic (IEEE 754-2008) on a standard x86 system. Quantitative evaluation of signature spectra extracted using a novel implementation will be done using the spectral angle mapper metric mentioned in previous section.

11.3.2 Performance and Power

The unit of work is one pixel in the hypercube. Consequently, there are ($M\times N\times L$) units of work; corresponding to number of rows M and number of columns N of images, and number of spectral wavelength bands L .

Performance is evaluated with time per units of work.

Power is evaluated with Watts per unit of work.

Constraining Problem 12

Hyperspectral Image: Target Detection

Erin I. Barker (PNNL), Nitin Gawande (PNNL), Joseph B. Manzano (PNNL), Nathan Tallent (PNNL)

12.1 Justification

Hyperspectral imaging collects information at a large number of narrow and contiguous wavelength bands from the electromagnetic spectrum for a given field of view or scene. Essentially a vector of reflectance values with each component corresponding to a particular wavelength band is gathered at each pixel of the image. This results in what is commonly known as a hyperspectral data cube. This three dimensional array is typically referred to as a *hypercube*. Hyperspectral imagery is used for a variety of applications. Images collected from a given sensor can be post-processed in multiple ways to extract different information. With no *a priori* knowledge of the image's purpose the entire data cube must be stored and transmitted. The size of the cube increases with increasing scene size or spatial resolution as well as increasing spectral resolution. The large data volumes put stringent requirements on transmission capability from the sensor, storage on and off the sensor, and computational resources for post-processing the imagery [11].

Target detection attempts to identify a given target within a scene. However, the target typically exists in a relatively small number of the pixel. Due to spatial resolution restrictions the target may not fill the pixel size resulting in mixed pixels. The scarcity of targets also results in a lack of training data from which to determine the target's spectral signature. This complicates target detection algorithms by having to estimate the target's signature and statistics about how it varies due to interference. Therefore most classification algorithms developed and optimized for hyperspectral image processing are not applicable [74]. The need for real-time post-processing of this computational intensive problem makes it a constraining problem for hyperspectral image processing.

12.2 Description

Creating a photographic image generates a 2D representation of a scene. For each pixel of the photograph a combination of the visible light bans, red, green, and blue, is captured. However, these are a very small fraction of the available spectral wavelengths. Spectral imaging captures the radiance of many more spectrum at each spatial pixel. If a discrete number of wavelengths from the electromagnetic spectrum are captured this is refereed to as spectral imaging or multispectral imaging. Hyperspectral imaging (HSI) refers to capturing the radiance for a very large number of contiguous spectral wavelength bands.

Capturing a hyperspectral image requires a source of radiation or illumination, typically the sun, a surface to be imaged, and the sensor capturing the radiance of the various wavelength bands. What the spectra for a given pixel looks like depends on how the material within the pixel reflects, absorbs, and emits the radiation and the path through the atmosphere that the radiation took to reach the material and then reach the sensor. How the atmosphere modifies and interferes with the spectra becomes important later on.

The digital representation of a simple 2D photograph is a single color value for each pixel. Since HSI is capturing a radiance value for large number of wavelengths the digital representation becomes 3D. The x and y axes are the spatial coordinates of the scene and the third dimension, γ, is the radiance value for each wavelength. This 3D data representation is often referred to as a hyperspectral data cube. As the spatial resolution or the number of wavelengths captured increases so does the size of the data cube. Typically the data volume increases linearly with the number of spectral bands and as the square of the spatial resolution [59]. Therefore in practice the cost of data storage, transmission from the sensor, and post-processing must be weighed against the benefit of increasing either the spatial or spectral dimensions. Because sensors capture radiance at all specified wavelength the is no need for *a priori* knowledge of what the hyperspectral image will be utilized for or the type of post-processing that will be conducted. Therefore, a single hyperspectral image can be used for multiple purposed. However, this means many wavelength may be capture that will not be used in post-processing which increases the data storage and transmission requirements without explicit benefit.

Hyperspectral imaging was initial developed for mining and geology applications but is now utilized in a wide range of applications including agriculture for determining crop health, astronomy. biomedical imaging, mineralogy, physics, and surveillance. Hyperspectral imaging is particularly useful when shape and morphology information are not sufficient to describe a scene. For example, when multiple materials are in close proximity or contained within a single pixel, when shape and morphology information are not known in advanced, and for cases in which the shape and morphology are altered due to neighboring objects or deception.

Once a hyperspectral image has been captured the typical workflow is shown in Figure 12.1. The type of detection statistic selected depends on the goal of the post-processing. Independent of specific application area the purpose of analyzing a hyperspectral image can fall into a small set of categories including classification, unmixing, change detection, and target or anomaly detection. The goal of classification is to group pixels that contain the same or similar materials until all pixels are assigned to a class. Unmixing strives to separate multiple materials that appear within a single pixel or neighboring pixels. This is particularly useful for mineral exploration to determine the percentage of a given mineral present. Given a series of hyperspectral images captures over a period of time, changes in the spectra for a given spatial location can determine changes occurring within a scene. Finally, target detection and anomaly detection seek to find a relatively few number of pixels within a scene that contain a specific

spectral signature or simply vary from the background spectra. For target detection, the goal is to separate a specific spectral signature from the rest of the scene deemed the background. The goal of anomaly detection is to find any pixel with a spectrum that deviates from the average spectra of the rest of the scene. Anomaly detection can be viewed as a specific case of target detection. This specification focuses on the larger problem of target detection.

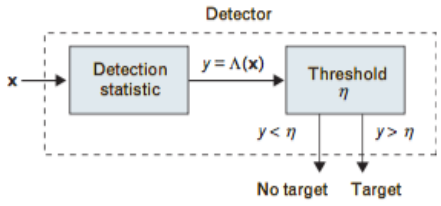


Figure 12.1: General workflow for hyperspectral image analysis [74].

Hyperspectral imaging can be particularly useful for target detection for several reasons. Spatial resolution may not allow for shape and morphology information to be extracted. Also spectral information can be more reliable when the target of interest is obscured due to surrounding materials or objects or when deception is used to conceal or alter the shape of a target.

In an ideal situation a spectral signature is known for the target. Then the workflow becomes comparing the spectra for each pixel to the target signature and classifying the pixel as having the target present or absent. The spectral signature for a given target or material is often extracted from sets of training data where a target is know to exist. However, one of the difficulties of target detection is that the target is present in relatively few pixels of a scene and limited to no training data is available. Target detection also suffers from the general HSI issues of variation of the signature signal due to sources of noise such as atmospheric conditions, angle of view, and characteristics of the radiation source (i.e. angle of sun, time of day, season, and latitude) and pixels containing multiple materials or target mixed with background.

The general workflow begins by specifying a binary hypothesis of the target being present in the current pixel or the target being absent (i.e. the pixel is part of the background) as shown in Equation 12.1. A detection statistic is utilized to determine the probability of the target being present or absent and cast as a likelihood ratio of $p(targetpresent)/p(targetabsent)$. If the likelihood ratio is greater than a determined threshold then the target is determined to be present. The determination of the threshold value is a complicated process and an area of continuing research but is beyond the scope of this specification.

$$\Lambda(\mathbf{x}) = \frac{p\left(\mathbf{x}|\text{target present}\right)}{p\left(\mathbf{x}|\text{target absent}\right)} \tag{12.1}$$

Different assumptions about input available and how this changes approach. What is known and what needs to be estimated.

The general class of detection statistics used is dependent on which variables are known and which need to be estimated, the presence of variation added to the spectra, and whether a target is full pixel or partial pixel (mixed). Since most target detection sensors are space based or aircraft mounted, all sources of noise can add variability to the spectra.

The simplest case is one where the mean and covariance of the target and background spectra are known and all pixels are pure target or background (i.e. full pixels). Given adequate training data and modeling the spectra as random vectors with multivariate normal distributions, the probability densities are known resulting in the following hypotheses:

$$\begin{array}{lll} H_0: & (\mathbf{x})\ N(\mu_0, \mathbf{\Gamma}_0) & \text{target absent} \\ H_1: & (\mathbf{x})\ N(\mu_1, \mathbf{\Gamma}_1) & \text{target present} \end{array}$$

In this case, a quadratic Neyman-Pearson detector can be designed as follows:

$$y = D(x) = (\mathbf{x} - \mu_0)^T \mathbf{\Gamma}_0^{-1} (\mathbf{x} - \mu_0) - (\mathbf{x} - \mu_1)^T \mathbf{\Gamma}_1^{-1} (\mathbf{x} - \mu_1) \tag{12.2}$$

This likelihoood-ratio detector calculates the Mahalanobis distance of the current spectrum from the centers of the target and background spectra. If the target and background have the same covariance matrix, $\Gamma_1 = \Gamma_0 = \Gamma$, then the detector reduces to a linear form

$$y = D(x) = \mathbf{c}_{MF}^T \mathbf{x}, \tag{12.3}$$

where

$$\mathbf{c}_{MF}^T = \kappa \mathbf{\Gamma}^{-1} (\mu_1 - \mu_0), \tag{12.4}$$

and where κ is a normalization constant. This is known as Fisher's linear discriminant or a matched filter (MF).

Due to sparsity of the target, in practice the statistics of the target are not know and must be estimated. Therefore, the quadratic detector and matched filter are not longer applicable. Only considering the case where the target and background have the same covariance matrix, $\Gamma_1 = \Gamma_0 = \Gamma$, reduces the problem to anomaly detection and makes it tractable. For this case, the likelihood ratio test becomes

$$y = D(x) = (\mathbf{x} - \mu_0)^T \mathbf{\Gamma}^{-1} (\mathbf{x} - \mu_0), \tag{12.5}$$

which calculates the Mahalanobis distance between the current pixel and the mean of the background spectra. Unfortunately the lack of information about the target spectra makes this equation nonlinear.

The class of adaptive detectors still considers full pixels, but expands to the case of the statistics of the background also being unknown. This requires the estimation of the mean and covariance of the background. This is typical

accomplished by using all of the pixel since the target pixels are assumed sparse. The number of pixels used to estimate the background statistics must be large enough to ensure the covariance matrix can be inverted but small enough for spectral homogeneity. It should be noted that adaptive detectors lose their Neyman-Pearson optimality properties and the detection statistics are no longer normal distributions. Finally, the performance of adaptive detectors approach optimum detectors as the estimation of the covariance matrix improves. In practice, the more common case is that pixels contain a mix of target and background (i.e. subpixel targets). Typically this case is assumed to be able to be modeled as a linear combination of the target and background spectra plus an additive amount of noise due to variability of the spectra. While variability such as the illumination level modify the length or magnitude of a given spectra, the direction or angular orientation of the spectra remains the same. Therefore, the goal of subpixel target detection algorithms is to determine the separation of a spectra from the target and background spectra.

Different mathematical models are utilized to describe the variability of a spectra which results in different families of detection algorithms. The variability of the target spectra is modeled using subspace models while the variability of the background is modeled with subspace or statistical models. It is this choice in background model that separates the families of subpixel target detection algorithms.

For detectors utilizing statistical models, it is often assumed that additive noise is present in the background spectra and is modeled by a multivariate normal distribution. The statistic of the background spectra are determined from training data that is assumed statistically independent from the current test pixel. The hypotheses become

$$\begin{aligned} H_0: &\quad \mathbf{x} = \mathbf{v} && \text{target absent} \\ H_1: &\quad \mathbf{x} = \mathbf{S}\mathbf{a} + \mathbf{v} && \text{target present} \end{aligned}$$

The current pixel spectra $\mathbf{x}$ has the mean of the background μ_0 already removed for simplicity. For this case, utilizing a generalized likelihood ratio (GLR) approach requires the maximum-likelihood estimate of the covariance matrix of the background. A simplified GLR detector is

$$D_A(x) = \mathbf{x}^T \hat{\mathbf{\Gamma}}^{-1} \mathbf{x}, \tag{12.6}$$

where $\hat{\mathbf{\Gamma}}$ is the maximum-likelihood estimate of the covariance matrix

$$\hat{\mathbf{\Gamma}} = \frac{1}{N} \sum_{n=1}^{N} \mathbf{x}(n) \mathbf{x}^T(n). \tag{12.7}$$

Since the amount of background mixed with target in a pixel changes, the hypotheses will change as well.

$$
\begin{aligned}
H_0 : &\quad \mathbf{x} = \mathbf{v} &\quad &\text{target absent} \\
H_1 : &\quad \mathbf{x} = \mathbf{Sa} + \sigma\mathbf{v} &\quad &\text{target present}
\end{aligned}
$$

This results in the same covariance of the background but a difference variance. This variance is directly related to the percentage of the pixel occupied by the target. For this case, the GLR approach results in an adaptive coherence or cosine estimator detector [56] [57].

The desirable features of any detection algorithm are a high probability of detection, a low probability of false alarm, robustness of the algorithm as real spectra vary from the assumed theoretical models of the spectra, constant false alarm rate (CFAR), as well as efficient software and hardware implementation.

12.2.1 Input

The input dataset is a hyperspectral scene called *hypercube*. The input data set has images consisting of M number of pixel rows, N number of pixel columns, and L number of bands corresponding to different wavelengths. The input generator is made available for any user to make a custom hypercube for this constraining problem.

Input Generator

The input generator is a software tool which enables making a hypercube with variable size and hyperspectral scene complexity. A user of this tool can choose size of the hyperspectral imaging scene. While any variable size can be used, default image sizes of 256 $\times$ 256 pixels, 1024 $\times$ 1024, and 2048 $\times$ 2048 are provided corresponding to small, medium, and large cases, respectively. The next input to the input generator is information on spectra which is used to construct hyperspectral scene. The input generator includes information on a large number of spectra from the ASTER Spectral Library Version 2.0 [8]. Any other spectrum of interest can also be included in addition to already included spectra. The format of these spectra is ASCII with information in two columns corresponding to spectra wavelengths and percent reflectance or absorbance. A hypercube is first constructed for a background scene without the present of target. Another hypercube is then constructed which includes the background scene with a target spectra injected into the scene.

12.2.2 Output

The output is expected to specify which pixels contain background and which pixels are full or partial targets. The probability of detection and the probability of false positives should be reported for each data set.

12.3 Evaluation

12.3.1 Correctness

The decision for each pixel will be compared to a known decision for both the synthetic and real data. The rate of false positives will be compared to some threshold. A successful algorithm should not only correctly identify all of the target pixels and minimize the incorrectly identifying pixels as background or target.

12.3.2 Performance and Power

Performance is evaluated with time per units of work.

Power is evaluated with Watts per unit of work.

Bibliography

[1] R. Adve, T. Hale, and M. Wicks. Practical joint domain localised adaptive processing in homogeneous and nonhomogeneous environments. 2. nonhomogeneous environments. *Radar, Sonar and Navigation, IEE Proceedings* -, 147(2):66–74, Apr 2000.

[2] I. Agi and L. Gong. An empirical study of secure mpeg video transmissions. In *Network and Distributed System Security, 1996., Proceedings of the Symposium on*, pages 137–144, Feb 1996.

[3] G. Angiulli, V. Barrile, and M. Cacciola. SAR imagery classification using multi-class support vector machines. *Journal of Electromagnetic Waves and Applications*, 19(14):1865—1872, 2005.

[4] M. A. Anusuya and S. K. Katti. Speech recognition by machine: A review. *Int'l Journal of Computer Science and Information Security*, 6(3):181–205, 2009.

[5] C. Argenta, A. Murphy, J. Hinton, J. Cook, T. Sherrill, and S. Snarski. Graphical user interface concepts for tactical augmented reality. In *Proc. SPIE 7688, Head- and Helmet-Mounted Displays XV: Design and Applications*, 2010.

[6] M. R. Azimi-Sadjadi, S. Ghaloum, and R. Zoughi. Terrain classification in SAR images using principal components analysis and neural networks. *IEEE Transactions on Geoscience and Remote Sensing*, 31(2):511–515, 1993.

[7] S. Baker and I. Matthews. Lucas-kanade 20 years on: A unifying framework. *Int. J. Comput. Vision*, 56(3):221–255, Feb. 2004.

[8] A. Baldridge, S. Hook, C. Grove, and G. Rivera. The aster spectral library version 2.0. *Remote Sensing of Environment*, 113(4):711–715, 2009.

[9] J. Barker, R. Marxer, E. Vincent, and S. Watanabe. The third 'chime' speech separation and recognition challenge: Dataset, task and baselines. In *Proc. IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2015.

[10] A. Bhat and T. Hammond. Using entropy to distinguish shape versus text in hand-drawn diagrams. In *Proc. 21st Int'l Joint Conf. on Artifical Intelligence*, 2009.

[11] J. Bioucas-Dias and J. Nascimento. Hyperspectral subspace identification. *Geoscience and Remote Sensing, IEEE Transactions on*, 46(8):2435–2445, Aug 2008.

[12] E. Blasch, Z. Liu, D. Petkie, R. Ewing, G. Pomrenke, and K. Reinhardt. Image fusion of the terahertz-visual naecon grand challenge data. In *Aerospace and Electronics Conference (NAECON), 2012 IEEE National*, pages 220–227, July 2012.

[13] R. Blum, Z. Xue, and Z. Zhang. An overview of image fusion. In R. S. Blum and L. Zheng, editors, *Multi-Sensor Image Fusion and Its Applications*, pages 1–35. Taylor and Francis, 2005.

[14] J. W. Boardman, F. Kruse, et al. Analysis of imaging spectrometer data using-dimensional geometry and a mixture-tuned matched filtering approach. *Geoscience and Remote Sensing, IEEE Transactions on*, 49(11):4138–4152, 2011.

[15] J. W. Boardman, F. A. Kruse, and R. O. Green. Mapping target signatures via partial unmixing of aviris data. In *Proc. JPL airborne earth sci. workshop*, volume 1, pages 23–26, 1995.

[16] R. P. Broussard, S. K. Rogers, M. E. Oxley, and G. L. Tarr. Physiologically motivated image fusion for object detection using a pulse coupled neural network. *IEEE Transactions on Neural Networks*, 10(3):554–563, May 1999.

[17] L. Brown. A survey of image registration techniques. *ACM Computing Surveys (CSUR)*, 24(4):325–376, Dec. 1992.

[18] W. M. Brown and L. J. Porcello. An introduction to synthetic-aperture radar. *Spectrum, IEEE*, 6(9):52–62, Sept 1969.

[19] P. Burt and E. Adelson. The laplacian pyramid as a compact image code. *Communications, IEEE Transactions on*, 31(4):532–540, Apr 1983.

[20] E. Candes, J. Romberg, and T. Tao. Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information. *Information Theory, IEEE Transactions on*, 52(2):489–509, Feb 2006.

[21] Y. K. Chan and V. C. Koo. An introduction to synthetic aperture radar (SAR). *Progress In Electromagnetics Research B*, 2:27–60, 2008.

[22] C.-I. Chang. *Hyperspectral data processing: algorithm design and analysis*. John Wiley & Sons, 2013.

[23] S. Chen, B. Kingsbury, L. Mangu, D. Povey, G. Saon, H. Soltau, and G. Zweig. Advances in speech transcription at IBM under the DARPA EARS program. *IEEE Trans. on Audio, Speech, and Language Processing*, 14(5):1596–1608, 2006.

[24] S. Chu, H.-K. Kuo, L. Mangu, Y. Liu, Y. Qin, Q. Shi, S. L. Zhang, and H. Aronowitz. Recent advances in the IBM GALE mandarin transcription system. In *Proc. Int'l Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, 2008.

[25] D. Cireşan, U. Meier, and J. Schmidhuber. Multi-column deep neural networks for image classification. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[26] A. Coates, H. Lee, and A. Y. Ng. An analysis of single-layer networks in unsupervised feature learning. In *Proc. 14th Int'l Conf. on Artificial Intelligence and Statistics*, 2011.

[27] B. Coüasnon, J. Camillerapp, and I. Leplumey. Access by content to handwritten archive documents: generic document recognition method and platform for annotations. *Int'l Journal of Document Analysis and Recognition (IJDAR)*, 9(2–4):223–242, 2007.

[28] B. Coüasnon and L. Pasquer. A real-world evaluation of a generic document recognition method applied to a military form of the 19th century. In *Proc. 6th Int'l Conf. on Document Analysis and Recognition*, 2001.

[29] N. Cvejic, A. Loza, D. Bull, and N. Canagarajah. A similarity metric for assessment of image fusion algorithms. *International Journal of Signal Processing*, 2(3):178–182, 2005.

[30] Defense Advanced Research Projects Agency (DARPA). DARPA-BAA-09-40 deep learning (DL) broad agency announcement, 2009.

[31] Defense Advanced Research Projects Agency (DARPA). DARPA-BAA-10-34 robust automatic transcription of speech (RATS) broad agency announcement, 2010.

[32] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A large-scale hierarchical image database. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 248–255, 2009.

[33] M. Desai and W. Jenkins. Convolution backprojection image reconstruction for spotlight mode synthetic aperture radar. *IEEE Transactions on Image Processing*, 1(4):505–517, October 1992.

[34] R. C. DiPietro. Extended factored space-time processing for airborne radar systems. In *Signals, Systems and Computers, 1992. 1992 Conference Record of The Twenty-Sixth Asilomar Conference on*, volume 1, pages 425–430, 1992.

[35] D. Doermann, J. Liang, and H. Li. Progress in camera-based document image analysis. In *Proc. 7th Int'l Conf. on Document Analysis and Recognition*, 2003.

[36] M. R. Endsley. Toward a theory of situation awareness in dynamic systems. *Human Factors*, 37(1):32–64, 1995.

[37] V. Farley, A. Vallières, A. Villemaire, M. Chamberland, P. Lagueux, and J. Giroux. Chemical agent detection and identification with a hyperspectral imaging infrared sensor. In *Proc. SPIE*, volume 6739, pages 673918–673918–12, 2007.

[38] N. Farrugia, F. Mamalet, S. Roux, F. Yang, and M. Paindavoine. Fast and robust face detection on a parallel optimized architecture implemented on fpga. *Circuits and Systems for Video Technology, IEEE Transactions on*, 19(4):597–602, April 2009.

[39] K. D. Forbus, J. Usher, and V. Chapman. Sketching for military courses of action diagrams. In *Proc. 8th Int'l Conf. on Intelligent User Interfaces*, 2003.

[40] S. K. Gaikwad, B. W. Gawali, and P. Yannawar. A review on speech recognition technique. *Int'l Journal of Computer Applications*, 10(3):16–24, 2010.

[41] G. Ginolhac, P. Forster, J. P. Ovarlez, and F. Pascal. Spatio-temporal adaptive detector in non-homogeneous and low-rank clutter. In *Proceedings of the 2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, ICASSP '09, pages 2045–2048, Washington, DC, USA, 2009. IEEE Computer Society.

[42] A. F. Goetz, G. Vane, J. E. Solomon, and B. N. Rock. Imaging spectrometry for earth remote sensing. *Science*, 228(4704):1147–1153, 1985.

[43] J. Goldstein, I. S. Reed, and L. Scharf. A multistage representation of the wiener filter based on orthogonal projections. *Information Theory, IEEE Transactions on*, 44(7):2943–2959, Nov 1998.

[44] L. A. Gorham and L. J. Moore. SAR image formation toolbox for MATLAB. In *Proceedings of SPIE, Algorithms for Synthetic Aperture Radar Imagery XVII*, volume 7699, 2010.

[45] J. Guerci and E. Baranoski. Knowledge-aided adaptive radar at darpa: an overview. *Signal Processing Magazine, IEEE*, 23(1):41–50, Jan 2006.

[46] A. Hafiane, K. Palaniappan, and G. Seetharaman. Uav-video registration using block-based features. In *Geoscience and Remote Sensing Symposium, 2008. IGARSS 2008. IEEE International*, volume 2, pages II–1104. IEEE, 2008.

[47] D. Hall and J. Llinas. An introduction to multisensor data fusion. *Proceedings of the IEEE*, 85(1):6–23, Jan 1997.

[48] G. E. Hinton, L. Deng, D. Yu, G. E. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29(6):82–97, 2012.

[49] M. Hossny, S. Nahavandi, and D. Creighton. Comments on 'information measure for performance of image fusion'. *Electronics Letters*, 44(18):1066–1067, August 2008.

[50] L. Hu and X. Xing. SAR target feature extraction and recognition based multilinear principal component analysis. In *Proc. Int'l Symp. on Optoelectronic Technology and Application 2014: Image Processing and Pattern Recognition*, 2014.

[51] H. Huang, L. Liu, and M. O. Ngadi. Recent developments in hyperspectral imaging for assessment of food quality and safety. *Sensors*, 14(4):7248–7276, 2014.

[52] Information Technology Laboratory (NIST). NIST special database 32 - multiple encounter dataset (meds). Technical report, NIST, Gaithersburg, MD, 2014.

[53] A. Jaimes and N. Sebe. Multimodal human computer interaction: A survey. In *Computer vision in human-computer interaction*, pages 1–15. Springer, 2005.

[54] V. Jain and E. G. Learned-Miller. Fddb: A benchmark for face detection in unconstrained settings. *UMass Amherst Technical Report*, 2010.

[55] S. Julier, Y. Baillot, M. Lanzagorta, D. Brown, and L. Rosenblum. BARS: Battlefield augmented reality system. In *Proc. NATO Symp. on Information Processing Techniques for Military Systems*, 2000.

[56] S. Kraut and L. Scharf. The cfar adaptive subspace detector is a scale-invariant glrt. *IEEE Transactions Signal Processing*, 47(9):2538–2541, 1999.

[57] S. Kraut, L. Scharf, and L. McWhorter. Adaptive subspace detectors. *IEEE Transactions Signal Processing*, 49(1):1–16, 2001.

[58] F. Kruse, A. Lefkoff, J. Boardman, K. Heidebrecht, A. Shapiro, P. Barloon, and A. Goetz. The spectral image processing system (sips)—interactive visualization and analysis of imaging spectrometer data. *Remote sensing of environment*, 44(2):145–163, 1993.

[59] D. Landgrebe. Hyperspectral image data analysis. *Signal Processing Magazine, IEEE*, 19(1):17–28, Jan 2002.

[60] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86:309–318, 1998.

[61] Y. LeCun, F. J. Huang, and L. Bottou. Learning methods for generic object recognition with invariance to pose and lighting. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2004.

[62] C.-Y. Lee, S. Xie, P. Gallagher, Z. Zhang, and Z. Tu. Deeply-supervised nets. In *Proc. of the 18th Int'l Conf. on Artificial Intelligence and Statistics (AISTATS)*, 2015.

[63] B. Leininger, J. Edwards, J. Antoniades, D. Chester, D. Haas, E. Liu, M. Stevens, C. Gershfield, M. Braun, J. D. Targove, S. Wein, P. Brewer, D. G. Madden, and K. H. Shafique. Autonomous real-time ground ubiquitous surveillance-imaging system (ARGUS-IS). In *Proc. SPIE*, volume 6981, pages 69810H–1–69810H–11, 2008.

[64] H. Li, B. Manjunath, and S. Mitra. Multisensor image fusion using the wavelet transform. *Graphical Models and Image Processing*, 57(3):235 – 245, 1995.

[65] H. Li, B. Manjunath, and S. Mitra. Multisensor image fusion using the wavelet transform. *Graphical Models and Image Processing*, 57(3):235 – 245, 1995.

[66] S. Li, J. T. Kwok, and Y. Wang. Multifocus image fusion using artificial neural networks. *Pattern Recognition Letters*, 23(8):985–997, 2002.

[67] P. Liang, G. Teodoro, H. Ling, E. Blasch, G. Chen, and L. Bai. Multiple kernel learning for vehicle detection in wide area motion imagery. In *Proc. 15th Int'l Conf. on Information Fusion (FUSION)*, 2012.

[68] Z. Liu, E. Blasch, Z. Xue, J. Zhao, R. Laganiere, and W. Wu. Objective assessment of multiresolution image fusion algorithms for context enhancement in night vision: A comparative study. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 34(1):94–109, Jan 2012.

[69] Z. Liu, D. S. Forsyth, and R. Laganiere. A feature-based metric for the quantitative evaluation of pixel-level image fusion. *Computer Vision and Image Understanding*, 109(1):56 – 68, 2008.

[70] G. Lu and B. Fei. Medical hyperspectral imaging: a review. *Journal of biomedical optics*, 19(1):010901–010901, 2014.

[71] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th international joint conference on Artificial intelligence - Volume 2*, IJCAI'81, pages 674–679, 1981.

[72] J. Mäkynen, H. Saari, C. Holmlund, R. Mannila, and T. Antila. Multi-and hyperspectral uav imaging system for forest and agriculture applications. In *SPIE Defense, Security, and Sensing*, pages 837409–837409. International Society for Optics and Photonics, 2012.

[73] S. Mallat. A theory for multiresolution signal decomposition: the wavelet representation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 11(7):674–693, Jul 1989.

[74] D. Manolakis, D. Marden, and G. A. Shaw. Hyperspectral image processing for automatic target detection applications. *Lincoln Laboratory Journal*, 14(1):79–116, 2003.

[75] W. Melvin. A STAP overview. *Aerospace and Electronic Systems Magazine, IEEE*, 19(1):19–35, 2004.

[76] O. Mendoza-Schrock, J. A. Patrick, and M. Garing. Exploring image registration techniques for layered sensing. In *Evolutionary and Bio-inspired Computation: Theory and Applications III*, volume 7347, pages 73470V–73470V–15, May 2009.

[77] J. H. Michels, M. Rangaswamy, and B. Himed. Performance of parametric and covariance based STAP tests in compound-gaussian clutter. *Digital Signal Processing*, 12(2–3):307 – 328, 2002.

[78] D. Munson Jr., J. O'Brien, and W. Jenkins. A tomographic formulation of spotlight-mode synthetic aperture radar. *Proceedings of the IEEE*, 71:917–925, August 1983.

[79] H. S. Neoh and A. Hazanchuk. Adaptive edge detection for real-time video processing using fpgas. *Global Signal Processing*, 7(3):2–3, 2004.

[80] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng. Reading digits in natural images with unsupervised feature learning. In *Proc. NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

[81] Y. Obuchi. Multiple-microphone robust speech recognition using decoder-based channel selection. In *Proc. ISCA Tutorial and Research Workshop on Statistical and Perceptual Audio Processing*, 2004.

[82] Y. Obuchi and R. M. Stern. Normalization of time-derivative parameters using histogram equalization. In *Proc. European Conf. on Speech Communication and Technology (INTERSPEECH)*, 2003.

[83] B. Pan, K. Li, and W. Tong. Fast, robust and accurate digital image correlation calculation without redundant computations. *Experimental Mechanics*, 53(7):1277–1289, 2013.

[84] J.-I. Park, S.-H. Park, and K.-T. Kim. New discrimination features for SAR automatic target recognition. *IEEE Geoscience and Remote Sensing Letters*, 10:476–480, 2013.

[85] D. B. Paul and J. M. Baker. The design for the wall street journal-based csr corpus. In *Proc. Workshop on Speech and Natural Language*, 1992.

[86] V. Petrovic and C. Xydeas. Gradient-based multiresolution image fusion. *Image Processing, IEEE Transactions on*, 13(2):228–237, Feb 2004.

[87] G. Piella. A general framework for multiresolution image fusion: from pixels to regions. *Information Fusion*, 4(4):259 – 280, 2003.

[88] S. Pigeon, C. Swail, D. Geoffrois, C. Bruckner, D. van Leeuwen, C. Teixeira, O. Orman, P. Collins, T. Anderson, J. Grieco, and M. Zissman. Use of speech and language technology in military environments. Technical Report TR-IST-037, North Atlantic Treaty Organisation (NATO), 2005.

[89] A. Plaza and C.-I. Chang. Impact of initialization on design of endmember extraction algorithms. *Geoscience and Remote Sensing, IEEE Transactions on*, 44(11):3397–3407, Nov 2006.

[90] A. Plaza, P. Martinez, R. Perez, and J. Plaza. A quantitative and comparative analysis of endmember extraction algorithms from hyperspectral data. *Geoscience and Remote Sensing, IEEE Transactions on*, 42(3):650–663, March 2004.

[91] A. Plyer, G. Le Besnerais, and F. Champagnat. Massively parallel lucas kanade optical flow for real-time video processing applications. *Journal of Real-Time Image Processing*, pages 1–18, 2014.

[92] R. Porter, A. M. Farser, and D. Hush. Wide-area motion imagery. *IEEE Signal Processing Magazine*, pages 56–65, September 2010.

[93] P. Price, W. M. Fisher, J. Bernstein, and D. S. Pallett. The DARPA 1000-word resource management database for continuous speech recognition. In *Proc. Int'l Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, 1998.

[94] G. Qu, D. Zhang, and P. Yan. Information measure for performance of image fusion. *Electronics Letters*, 38(7):313–315, Mar 2002.

[95] B. Rafert, R. G. Sellar, E. Holbert, J. H. Blatt, D. W. Tyler, S. E. Durham, and H. D. Newby. Hyperspectral imaging fourier transform spectrometers for astronomical and remote sensing observations. In *1994 Symposium on Astronomical Telescopes & Instrumentation for the 21st Century*, pages 338–349. International Society for Optics and Photonics, 1994.

[96] M. Rangaswamy. An overview of space-time adaptive processing for radar. In *Proceedings of the International Radar Conference*, pages 45–50, September 2003.

[97] B. D. Rigling and R. L. Moses. Taylor expansion of the differential range for monostatic SAR. *Aerospace and Electronic Systems, IEEE Transactions on*, 41(1):60–64, 2005.

[98] J. Roman, M. Rangaswamy, D. Davis, Q. Zhang, B. Himed, and J. Michels. Parametric adaptive matched filter for airborne radar applications. *Aerospace and Electronic Systems, IEEE Transactions on*, 36(2):677–692, Apr 2000.

[99] T. Ross, S. Worrell, V. Velten, J. Mossing, and M. Bryant. Standard SAR ATR evaluation experiments using the MSTAR public release dataset. In *Proc. SPIE Conf. on Algorithms for Synthetic Aperture Radar Imagery V*, 2012.

[100] G. Saon and J.-T. Chien. Large-vocabulary continuous speech recognition systems: A look at some recent advances. *IEEE Signal Processing Magazine*, 29(6):18–33, 2012.

[101] T. K. Sarkar, H. Wang, S. Park, R. Adve, J. Koh, K. Kim, Y. Zhang, M. Wicks, and R. Brown. A deterministic least-squares approach to space-time adaptive processing (STAP). *Antennas and Propagation, IEEE Transactions on*, 49(1):91–103, Jan 2001.

[102] R. A. Schowengerdt. *Remote sensing: models and methods for image processing*. Academic press, 2006.

[103] X. Shi, H. Ling, E. Blasch, and W. Hu. Context-driven moving vehicle detection in wide area motion imagery. In *Proc. 21st Int'l Conf. on Pattern Recognition (ICPR)*, 2012.

[104] M. I. Smith and J. P. Heather. A review of image fusion technology in 2005. In *Proc. SPIE*, volume 5782, pages 29–45, 2005.

[105] H. Soltau, G. Saon, B. Kingsbury, H.-K. J. Kuo, L. Mangu, D. Povey, and A. Emami. Advances in arabic speech transcription at IBM under the DARPA GALE program. *IEEE Trans. on Audio, Speech, and Language Processing*, 17(5):884–894, 2009.

[106] K. Swersky, J. Snoek, and R. P. Adams. Multi-task bayesian optimization. In *Proc. Advances in Neural Information Processing Systems (NIPS)*, 2013.

[107] N. R. Tallent, J. B. Manzano, N. A. Gawande, S. Kang, D. J. Kerbyson, A. Hoisie, and J. K. Cross. Algorithm and architecture independent benchmarking with SEAK. In *Proc. of the 30th IEEE Intl Parallel and Distributed Processing Symp.* IEEE Computer Society, May 2016.

[108] C. Tison, N. Pourthié, and J.-C. Souyris. Target recognition in SAR images with support vector machines (SVM). In *IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, pages 456–459, Barcelona, Spain, July 2007.

[109] K. Tombre and B. Lamiroy. Pattern recognition methods for querying and browsing technical documentation. *Lecture Note in Computer Science*, 5197:504–518, 2008.

[110] F. D. Van der Meer, H. M. Van der Werff, F. J. van Ruitenbeek, C. A. Hecker, W. H. Bakker, M. F. Noomen, M. van der Meijde, E. J. M. Carranza, J. B. de Smeth, and T. Woldai. Multi-and hyperspectral geologic remote sensing: A review. *International Journal of Applied Earth Observation and Geoinformation*, 14(1):112–128, 2012.

[111] P. Viola and M. J. Jones. Robust real-time face detection. *International journal of computer vision*, 57(2):137–154, 2004.

[112] T. Wan, N. Canagarajah, and A. Achim. Compressive image fusion. In *Image Processing, 2008. ICIP 2008. 15th IEEE International Conference on*, pages 1308–1311, Oct 2008.

[113] H. Wang and L. Cai. A localized adaptive mtd processor. *Aerospace and Electronic Systems, IEEE Transactions on*, 27(3):532–539, May 1991.

[114] J. Ward. Space-time adaptive processing for airborne radar. Technical Report F19628-95-C-0002, MIT Linclon Laboratory, December 1994.

[115] Z. Wei, W. Jie, and G. Jian. An efficient SAR target recognition algorithm based on contour and shape context. In *Proc. 3rd Int'l Asia-Pacific Conf. on Synthetic Aperture Radar (APSAR)*, 2011.

[116] M. Wicks, M. Rangaswamy, R. Adve, and T. Hale. Space-time adaptive processing: a knowledge-based perspective for airborne radar. *Signal Processing Magazine, IEEE*, 23(1):51–65, Jan 2006.

[117] D. T. Williamson, M. H. Draper, G. L. Calhoun, and T. P. Barry. Commercial speech recognition technology in the military domain: Results of two recent research efforts. *Int'l Journal of Speech Technology*, 8(1):9–16, 2005.

[118] M. E. Winter. N-FINDR: an algorithm for fast autonomous spectral endmember determination in hyperspectral data. In *SPIE's International Symposium on Optical Science, Engineering, and Instrumentation*, pages 266–275. International Society for Optics and Photonics, 1999.

[119] J. Woodard and E. Cupples. Selected military applications of automatic speech recognition technology. *IEEE Communications Magazine*, 21(9):35–41, 1983.

[120] X. Xing, K. Ji, H. Zou, and J. Sun. Sparse representation based SAR vehicle recognition along with aspect angle. *The Scientific World Journal*, 2014, 2014.

[121] C. Xydeas and V. Petrovic. Objective image fusion performance measure. *Electronics Letters*, 36(4):308–309, Feb 2000.

[122] J. Yang, X. Chen, J. Zhang, Y. Zhang, and A. Waibel. Automatic detection and translation of text from natural scenes. In *Proc. IEEE Int'l Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, 2002.

[123] M.-H. Yang, D. Kriegman, and N. Ahuja. Detecting faces in images: a survey. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(1):34–58, Jan 2002.

[124] P. W. Yuen and M. Richardson. An introduction to hyperspectral imaging and its application for security, surveillance and target acquisition. *The Imaging Science Journal*, 58(5):241–253, 2010.

[125] A. Zare and K. Ho. Endmember variability in hyperspectral analysis: Addressing spectral variability during spectral unmixing. *Signal Processing Magazine, IEEE*, 31(1):95–104, 2014.

[126] Y. R. Zeng, T. Shao, and E. Blasch. A fast-converging space-time adaptive processing algorithm for non-gaussian clutter suppression. *Digital Signal Processing*, 22(1):74 – 86, 2012.

[127] Z. Zhang and R. Blum. A categorization of multiscale-decomposition-based image fusion schemes with a performance study for a digital camera application. *Proceedings of the IEEE*, 87(8):1315–1326, Aug 1999.

[128] W. Zhao, R. Chellappa, P. J. Phillips, and A. Rosenfeld. Face recognition: A literature survey. *ACM computing surveys (CSUR)*, 35(4):399–458, 2003.

[129] Y. Zheng. Multi-scale fusion algorithm comparisons: Pyramid, dwt and iterative dwt. In *Information Fusion, 2009. FUSION '09. 12th International Conference on*, pages 1060–1067, July 2009.

[130] Y. Zheng, E. A. Essock, B. C. Hansen, and A. M. Haun. A new metric based on extended spatial frequency and its application to dwt based fusion algorithms. *Information Fusion*, 8(2):177 – 192, 2007. Special Issue on Image Fusion: Advances in the State of the Art.

[131] Y. Zhou. Multi-sensor image fusion. In *Image Processing, 1994. Proceedings. ICIP-94., IEEE International Conference*, volume 1, pages 193–197 vol.1, Nov 1994.

[132] M. Zortea and A. Plaza. A quantitative and comparative analysis of different implementations of N-FINDR: A fast endmember extraction algorithm. *Geoscience and Remote Sensing Letters, IEEE*, 6(4):787–791, 2009.

[133] M. Zortea and A. Plaza. Spatial preprocessing for endmember extraction. *Geoscience and Remote Sensing, IEEE Transactions on*, 47(8):2679–2693, 2009.