8-2005

Civilian Target Recognition using Hierarchical Fusion

Balasubramanian Lakshminarayanan
University of Tennessee - Knoxville

To the Graduate Council:

I am submitting herewith a thesis written by Balasubramanian Lakshminarayanan entitled "Civilian Target Recognition using Hierarchical Fusion." I have examined the final electronic copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Electrical Engineering.

Hairong Qi, Major Professor

We have read this thesis and recommend its acceptance:

Daniel B. Koch, Seong G. Kong

Accepted for the Council:
Carolyn R. Hodges

Vice Provost and Dean of the Graduate School

(Original signatures are on file with official student records.)

To the Graduate Council:

I am submitting herewith a thesis written by Balasubramanian Lakshminarayanan entitled "Civilian Target Recognition using Hierarchical Fusion". I have examined the final paper copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Electrical Engineering.

Hairong Qi

Major Professor

We have read this thesis
and recommend its acceptance:

Daniel B. Koch

Seong G. Kong

Accepted for the Council:

Anne Mayhew

Vice Chancellor and Dean of
Graduate Studies

(Original signatures are on file with the official student records.)

Civilian Target Recognition using Hierarchical Fusion

A Thesis
Presented for the
Master of Science Degree
The University of Tennessee, Knoxville

Balasubramanian Lakshminarayanan
August 2005

Acknowledgements

"Narayana yeti Samarpayami"

Amma and Appa, I am forever indebted to you for your patient ways of caring, understanding and support. Without your love and encouragement, I would have never climbed up to where I am. Thanks for always being there for me.

I would like to thank Dr.Hairong Qi for her valuable guidance and advice, and for being the best advisor anyone could wish for. Her teaching methods have been thorough and precise and have taught me to learn on my own. I thank her for the opportunity to work with her and for being ever so patient with me. Her constant motivation has helped me in a big way. I would also like to thank the members of my thesis committee, Dr. Daniel Koch and Dr. Seong Kong, for their time and inputs.

Additionally, I would like to thank my relatives and friends for being there and supporting me in times of need. Akka, Maneesh, Aarti, Karen, Meens, Su, Subha, Kiran, Chith and Chithi – thanks a lot! I wish to express my gratitude to the people around me in Knoxville – Ezhil "Jee", Srini "Pannadi", Giri and Padiy "Daddy" who have been more like brothers, and Priya, Maya, Chandra and Sangi who have been the best of friends. Without you guyz, I would have missed out on a lot of good food and good times!

Special thanks are due to the members of the AICIP lab. Thanks to Gaurav who's esoteric and dumbfounding thinking has made me wonder on the intricacies of the human brain. Thanks to Sankar who's unending zest for food and life never ceases to amaze me. Teja, Xiaoling, Hongtao, Yang, Yingyue, Lidan and Woye – hats off to you guyz for making a great team.

Abstract

The growth of computer vision technology has been marked by attempts to imitate human behavior to impart robustness and confidence to the decision making process of automated systems. Examples of disciplines in computer vision that have been targets of such efforts are Automatic Target Recognition (ATR) and fusion. ATR is the process of aided or unaided target detection and recognition using data from different sensors. Usually, it is synonymous with its military application of recognizing battlefield targets using imaging sensors. Fusion is the process of integrating information from different sources at the data or decision levels so as to provide a single robust decision as opposed to multiple individual results. This thesis combines these two research areas to provide improved classification accuracy in recognizing civilian targets. The results obtained reaffirm that fusion techniques tend to improve the recognition rates of ATR systems.

Previous work in ATR has mainly dealt with military targets and single level of data fusion. Expensive sensors and time-consuming algorithms are generally used to improve system performance. In this thesis, civilian target recognition, which is considered to be harder than military target recognition, is performed. Inexpensive sensors are used to keep the system cost low. In order to compensate for the reduced system ability, fusion is performed at two different levels of the ATR system – event level and sensor level. Only preliminary image processing and pattern recognition techniques have been used so as to maintain low operation times. High classification rates are obtained using data fusion techniques alone. Another contribution of this thesis is the provision of a single framework to perform all operations from target data acquisition to the final decision making.

The Sensor Fusion Testbed (SFTB) designed by Northrop Grumman Systems

has been used by the Night Vision & Electronic Sensors Directorate to obtain images of seven different types of civilian targets. Image segmentation is performed using background subtraction. The seven invariant moments are extracted from the segmented image and basic classification is performed using k Nearest Neighbor method. Cross-validation is used to provide a better idea of the classification ability of the system. Temporal fusion at the event level is performed using majority voting and sensor level fusion is done using Behavior-Knowledge Space method.

Two separate databases were used. The first database uses seven targets (2 cars, 2 SUVs, 2 trucks and 1 stake body light truck). Individual frame, temporal fusion and BKS fusion results are around 65%, 70% and 77% respectively. The second database has three targets (cars, SUVs and trucks) formed by combining classes from the first database. Higher classification accuracies are observed here. 75%, 90% and 95% recognition rates are obtained at frame, event and sensor levels. It can be seen that, on an average, recognition accuracy improves with increasing levels of fusion. Also, distance-based classification was performed to study the variation of system performace with the distance of the target from the cameras. The results are along expected lines and indicate the efficacy of fusion techniques for the ATR problem. Future work using more complex image processing and pattern recognition routines can further improve the classification performance of the system. The SFTB can be equipped with these algorithms and field-tested to check real-time performance.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Advances in science and technology have always been directed towards developing smarter and efficient systems. In this approach, humans have long been trying to mimic the Human Visual System (HVS) so as to enable machines to look at their environment and act accordingly. Insofar, it can be safely concluded that no machine has been built that can match the durability and dexterity of the HVS. Decades of research and billions of dollars have been spent on attempts to develop systems that can detect and recognize objects of interest amongst an irrelevant or uninteresting background. Automatic Target Recognition (ATR) is the process of aided or unaided target detection and recognition using data from different sensors. Usually, ATR algorithms work towards developing better image segmentation techniques or image features to improve the classification accuracy. ATR is a term generally associated with its military application of recognizing battlefield targets using imaging sensors.

With the development of cheaper sensors, many approaches of ATR now concentrate on using different sensors to monitor the same event. Information from these sensors is used to provide a single reliable decision. The inherent advantage of this method compared to the HVS is the ability to sense signals that cover a wider frequency range – the HVS is limited to the visual spectrum. The usage of different sensors also requires some technique to combine the various information sensed. The best example of an efficient fusion system is the human perception system which combines visual, auditory and sensory information to make robust

decisions. While trying to design ATR and fusion systems with the aim of matching human capabilities, it must be kept in mind that the human brain is a huge, parallel-connected neural network with years of training.

In this thesis, ATR is performed on civilian targets. Civilian targets are considered to be harder to classify than military targets. This is because their spectral content usually contains peaks at higher frequencies compared to military targets. Military targets are louder than civilian vehicles and hence the low frequencies tend to dominate. Commercial vehicles have a smaller engine and higher RPM rates, thereby causing high frequency audio signals. Also, commercial vehicles have better muffling which makes the low frequencies harder to observe. Thus, civilian target detection and recognition provide a harder classification problem than their military counterparts. The image processing algorithms that are used in this thesis are simple. High classification accuracy is achieved by using data fusion. Two levels of fusion, event-level and sensor-level, are performed and it is shown that each level of fusion improves the classification accuracy.

This chapter of the thesis introduces the reader to the topic of the thesis and explains the motivation behind the thesis, its objectives, contributions and the required background details.

1.1 Motivation

ATR is a multi-faceted problem with a variety of applications in the military and industrial areas. Military systems that perform targeting, detection, tracking or surveillance need ATR capabilities. ATR also reduces the workload of human operators in intense battleground situations. Since ATR is an automation of the human cognitive process, current approaches try to emulate human behavior for ATR. This provides insights into the powers of perception and recognition of humans and animals. Thus it can be seen that ATR is an important area of study with a lot of scope for research. However, ATR algorithms typically use non-portable and expensive sensors with computationally complex algorithms to boost their performance levels. This increases the system cost and processing time. Data and sensor fusion provide the ability to arrive at more reliable and

robust decisions at a cheaper cost and processing time. The redundancy and complementarity of sensors are desirable features.

This thesis combines ATR and data fusion for recognizing civilian targets. As mentioned earlier, ATR for military targets is a well-studied research area. ATR of civilian targets is however a relatively new field of study with many potential applications. The motivation of this thesis is to perform reliable recognition of civilian targets using easily accessible sensors. Reliability is achieved using data fusion techniques.

In collecting the dataset, the Sensor Fusion Testbed (SFTB) developed by the NVESD has been used. This testbed was developed in order to collect imaging and auditory information simultaneously so that a reliable dataset and ground truth information could be recorded. This dataset would then be used to test and develop algorithms for moving target ATR. Also, the development of the testbed establishes the necessary techniques and equipments needed to collect ground truth information of the targets.

1.2 Automatic Target Recognition

Automatic Target Recognition is the process of detecting and recognizing a target from the input sensory data. This technique is used to reduce human workload by attempting to replace the HVS in demanding situations like battlefield conditions. The goal is to develop ATR systems that can support lock-on-after-launch (LOAL) systems and fire-and-forget systems [4] that can operate consistently well.

Typically, ATR is used to perform tasks like image acquisition, target detection, tracking and recognition. In primitive ATR systems, only target acquisition is performed autonomously leaving the rest of the tracking and recognition to the human operator. This process is known as aided ATR or target cueing [2]. Depending on the complexity of the ATR system and/or algorithm, it can perform relatively simple tasks like cueing the target for the human operator or can perform target recognition itself. In the autonomous mode of operation, the ATR system is wholly responsible for identification of the target also. Current ATR systems are not capable of autonomous ATR. In the military and medicinal fields,

which are typical applications of ATR, tolerance for false detections is very low. It is quite obvious that human-in-loop systems will be used until autonomous ATR can be demonstrated as having consistently superior performance. Hence the failure rate of current autonomous ATR systems makes them unsuitable for practical usage.

In addition to classifying ATR systems on the basis of the level of human intervention as aided and autonomous, they can can also be classified based on their output values as binary and multivalued [1]. In binary ATR systems, the system answers to target detection scenarios with a yes or no i.e the system either classifies the region-of-interest (RoI) as a target or not a target. Multivalued ATR systems assign a number to the RoI to indicate the likelihood that it is a target. For example, the multivalued system assigns a probability ranging from 0.0 to 1.0 to the RoI to indicate if it is a target.

Though the HVS is the oldest and probably the most efficient sensor we know of, it has certain shortcomings. It is sensitive only to a certain frequency band (the visual band) and it can detect only illuminated, relatively close and clear targets. Also, humans are error-prone when continuous and repetitive tasks are at hand. ATR systems can overcome these limitations by using different sensing modalities and by automating the tasks of target detection and identification.

ATR systems can have one or more sensors. Typical sensors used for ATR are visual cameras, forward-looking infrared sensors (FLIR), RADAR sensors and LASERs. In recent years, sensors like Synthetic Aperture RADAR (SAR), Inverse Synthetic Aperture RADAR (ISAR), LASER RADAR (LADAR) and multispectral sensors have gained in importance. The number and type of sensors used depends on the ATR algorithm, processing ability and the application where the system is to be deployed. These sensors view the scene and detect possible instances of the target. An ATR system can be described using the block diagram shown in Figure 1.1. A typical ATR algorithm localizes the region of interest (RoI) and extracts features of the target from the input data provided by the sensors. These features are compared against those of the possible classes to which the target may belong. Based on previous learning experiences or a target database, the target is then classified into a particular class. The ATR system is trained to

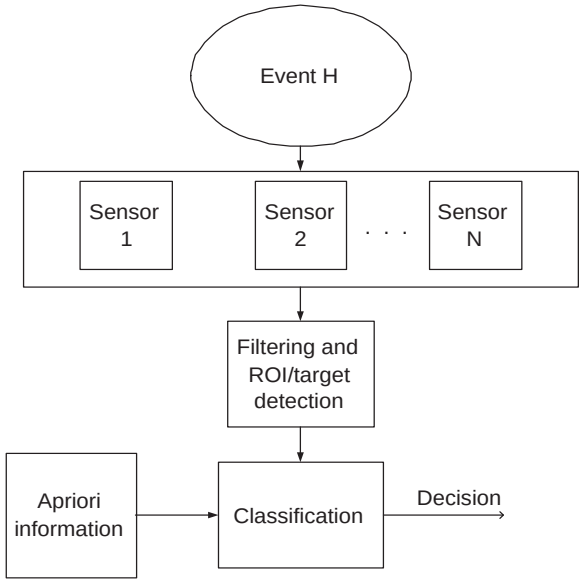


Figure 1.1: Process flow for ATR

recognize the targets using supervised or unsupervised training methods. Comparison of the target's feature vector with those of the training samples help in classifying and recognizing the target.

The performance of an ATR system depends on many associated technologies. These include hardware technologies (sensors and hardware evaluation and architecture), software technologies (algorithm development, software evaluation and architectures), theoretical concepts (statistical pattern recognition, neural networks, genetic algorithms, adaptive learning systems and model based vision), image processing techniques (image segmentation and feature extraction) and physical principles (detection theory, multiresolution processing and statistical techniques).

1.2.1 Requirements for ATR

Feature selection, extraction and matching for ATR systems is a complicated process. This is mainly due to the infinite variations that are possible in the target signatures. Typical variables include relative orientation of the target and sensor, image resolution, target camouflage, time of the day, season,terrain conditions and

vegetation. In addition, the target may vary from time to time; for example, the turrets of tanks may be angled differently and doors of vehicles may be opened or closed. Another major problem for ATR systems is the presence of clutter. Clutter refers to all the information content present in a scene that is not related to the target. This can be caused by noise in the atmosphere, weather conditions, sensing limitation of the sensor or sensor noise. In order to address all these problems, ATR algorithms have to meet certain requirements [8, 24]. Some of these are listed below.

1. Sensors with high resolution are needed to capture more information from the scene.

2. High speed processors were needed to work on all the information provided by the sensors. This need is now obsolete with the advent of high-speed chips and processors.

3. Collateral information, i.e. information from other sources, should be used for better performance.

4. Low false alarm rates are desired. For military applications, it is necessary that only the right targets are detected so that background or friendly objects are not fired upon.

5. High and reliable detection rates must be maintained. In military terms, the cost of a low detection rate would be the loss of an aircraft or tank.

6. Real-time operation is desirable. Hence the magnitude of calculations in the algorithm should be manageable for the hardware of the system.

7. The classification algorithm should be capable of incorporating new target information during operation. This ensures that the algorithm need not be trained again to accommodate new targets or different version of an old target.

8. The system must be capable of representing cases when it cannot confidently classify a target and alert the human operator.

9. The ATR system must be capable of identifying new targets.

10. The ATR system must work as independently of the clutter as possible.

1.2.2 Sensing Modalities for ATR

Sensors are the connecting blocks between the ATR system and the real world. They convert environmental and target parameters into data that can be used by the system. The sensors used by an ATR system are selected based on the resolution needed and the current application. For example, for aided ATR in which the ATR system has to locate possible target areas, low quality sensors may be sufficient. If the system is expected to identify the target and suggest possible matches, higher quality of images and sensors will be needed. The following is a list of sensors that are typically used for ATR. These sensors are usually used individually or in some combination.

- Forward Looking InfraRed (FLIR)
- Video cameras
- Radio Aided Detection and Ranging (RADAR)
- Sound Navigation and Ranging (SONAR)
- Light Amplification by Stimulated Emission of Radiation (LASER)
- LASER RADAR (LADAR)
- Synthetic Aperture Radar (SAR)
- Inverse Synthetic Aperture Radar (ISAR)
- Microwave/Millimeter wave (MMW)
- Acoustic
- Seismic
- Multispectral

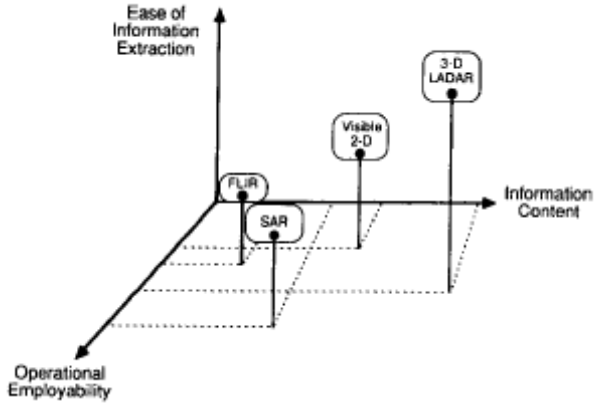


Figure 1.2: Operation of sensors [4]

- Hyperspectral

The sensors can be active sensors or passive sensors depending on their mode of operation. Passive sensors work by measuring signals emitted or reflected by the targets, like brightness or heat. Active sensors emit signals towards the target to calculate their features. Table 1.1 lists different sensors, their information content and the feature they sense [2, 4].

Sensors like LASERs, LADARs, MMW and acoustic sensors can be active sensors. FLIR and visible sensors are passive by nature. The resultant images obtained from them show the relative brightness or surface temperature of the objects in the scene. LASER sensors emit a coherent beam of light towards the target, and the reflected beam is captured and time of travel is measured. This gives features like range, velocity and angular resolution of the target. Similarly, RADAR beams can measure range and velocity using the time of travel and Doppler's Effect. Acoustic sensors use a high frequency beam of sound waves to achieve a similar purpose. The operation of different sensors with respect to information content, ease of operation and information extraction is shown in Figure 1.2.

The description and performance characteristics of sensors are discussed below [2, 14].

Table 1.1: Sensors and features sensed

Sensor	Type	Information content	Feature Sensed
Acoustic	Passive	Low	Range, velocity
SONAR	Active	Moderate	Range, velocity
Visible sensor	Passive	Low/Moderate	Color
FLIR	Passive	Moderate	Temperature
LASER	Active	Moderate	3D shape, range, velocity
RADAR	Active	Moderate/High	Motion
Seismic	Passive	Moderate	Vibration
MMW	Active	Moderate/High	Distance
High resolution imaging	Active/ Passive	High	Shape, color
Multi/hyper spectral imaging	Active/ Passive	High	Depending on spectrum covered

Visible Sensors These cover the 780-380nm range of the electromagnetic spectrum. The visible light reflected by the target and the environment is sensed. The image that is obtained from these sensors represents the brightness or intensity of the scene at every pixel. Though high resolution images can be obtained using visible sensors, their operation is dependent on the level of illumination present. Hence, they are suitable only for daytime operation. Also, the target must be clearly visible to the sensor for proper detection. Occlusion and inclement weather conditions affect the operation of the sensor adversely. They are generally used as low-cost sensors to sense easily visible targets. Typical uses include multispectral arrays, video cameras and LADARs.

Acoustic These sensors cover the 1-10KHz frequency range. The audible signal of the target is sensed if the sensor is a passive acoustic sensor. In case of active sensors, a short high-bandwidth pulse is sent towards the target and the reflected signal is studied to obtain range and velocity information (SONAR). These can be used during day or night times. The resolution of these sensors are poor and the sensing distances are short – a few meters in air and several hundred meters in water. The signal attenuation is high and depends on the distance, atmospheric conditions and frequency used.These sensors are usually used in SONAR systems, seismometers and acoustic detectors.

FLIR These sensors can detect signals between 300nm-1 micron. They detect the thermal signature of the scene. Since they do not depend on the light intensity of the scene, they can be used for day and night operations. However, it must be kept in mind that the signal level varies with the time of the day. High angular resolution can be obtained using these sensors. Their operation is limited by atmospheric conditions like rain and fog. Also, they have poor foliage penetration capabilities like visible sensors. When used from the sky, they can have an effective range of 10-15km. They are typically used in infrared cameras, multispectral arrays and focal plane arrays.

LASER Moderate to high resolution range data can be obtained using LASER

sensors. The sensors work by sending out a concentrated beam of light in different directions. The reflected beam is studied to find accurate range and texture information. These sensors can also be used for day and night time operations. Again, their operation is affected by the weather conditions.

RADAR The principle of operation of these sensors is the same as that of LASER sensors. These sensors use radio waves to study target information. MMW RADARs typically use frequencies between 30-300GHz to find range, velocity and intensity data. Though the resolution of the image is not high, these sensors can cover a large area, especially when used from the sky. SARs and ISARs are usually used from airplanes. These sensors also have good foliage penetration and adverse weather does not affect these sensors as much as others.

Multi/hyper spectral sensors These sensors use a combination of the above for achieving target information at different frequencies. For example, a multispectral sensor using visual, IR and RADAR frequencies can obtain brightness, thermal and range information. The imaging data obtained is of extremely high resolution. These sensors are gaining in importance with the rise of fusion systems used in ATR.

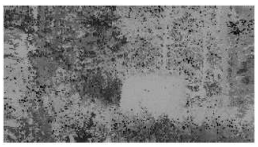
Examples of images sensed by a few sensors are shown in Figure 1.3.

1.2.3 Challenges in ATR

Though research in ATR has been carried out for well over two decades now, there are still no ATR systems that are deployable in the real field. This is because the ATR process is plagued with many problems. All the early papers discussing ATR talk about the practical concerns of implementing ATR systems [1, 2, 4, 8, 17].

The major problem is target and scene variability. There are many different possible combinations of target signatures for a single target. This can vary depending on the target orientation towards the sensor, distance, rotation, time of day, weather, illumination, terrain, vegetation, different models of the same target (different make or manufacturer), optional target equipments, aspect, clutter and

(a) Image in passive infrared region



(b) LASER RADAR image of same region as (a)

(c) 2D display of RADAR range data

(d) SAR image

Figure 1.3: Examples of images sensed by different sensors [17]

other such features. In case of military targets like tanks, further variation in target signature is caused by keeping the hatches open or closed and with various angles of the turret. It is practically impossible to train the ATR system with a dataset that represents all these variations and then deploy it. The total number of images that will be required for training is the product of the number of images required to represent each feature.

In addition to the non-availability of a representative database, it is often not clear as to what the ATR system is expected to achieve. There are no metrics to define the objective and performance of ATR systems. This problem can be thought of as one that is carried over from the image processing domain, in which the quality of the image has no objective metric measure but is often subjectively measured by the human observer looking at it. Another problem in ATR is the absence of robust algorithms that can provide high detection rates while maintaining low false alarm rates. This performance is affected by the presence of high clutter in typical ATR environments, such as battlefields. Also, typical ATR systems are incapable of identifying new targets or even indicating that the current target is new. ATR systems usually return the nearest match because distance measures in the feature space are usually used to classify the target. Hence the output of the system will be the target in the database that best matches the target under observation. This may prove to be a big setback since the new target may or may not be hostile, and a false alarm may be as costly as an undetected target.

The problems faced in the area of ATR are listed below, followed by a brief description.

1. *Target signature variability:* As mentioned above, the target signatures can vary a lot. The ATR system has to effectively capture all important variations and detect targets beyond such changes. Though humans deal with target variation easily, making ATR systems adapt to these changes is a very tough problem.

2. *Non-representative databases:* A database that represents the infinite variability of the target and background clutter cannot be generated practically. Even if such a database was generated, the training time would be

huge. Also, once such a system was deployed, introducing a new target or adding another feature to the database would usually require the system to be retrained (unless the system can be trained on-the-fly).

3. *Feature selection:* Choosing the features that are to be extracted must depend on the application, target environment and sensors used. The number of features to be used must be decided by the programmer. Usually, the more the features chosen, the better the classification accuracy.

4. *Number of samples:* The number of samples that are to be used in training the ATR system needs to be chosen carefully. Selecting too few samples will deteriorate the classification accuracy, while using too many could result in overfitting and large training time.

5. *Algorithms:* New algorithms have to be developed that can provide high detection rates without compromising on the false alarm rates. For this, the algorithm must be clutter independent and have hardware with high computational power. In this direction, new techniques like adaptive learning systems, knowledge based systems, neural networks and model based systems are being tested.

6. *Measurement metrics:* Proper metrics have to be defined to measure the performance of the ATR system. Also, the objective of the ATR system must be specified. This helps the programmer to have a fixed goal instead of a human measure that is difficult to translate into a program. Metrics for input image complexity will also be useful, since these can be used to define the expected performance of the system. Images that are more complex, in terms of clutter, cannot be expected to provide as good results as those that are simpler.

7. *New targets:* The algorithm must be capable of handling new targets and indicating to the human operator that a new target has been sighted. This problem arises when the classification is done solely on the basis of the distance of the features of the target from the features of the classes in the

database. As discussed, neither false alarms nor non-detections are good and in the actual application, both of them might prove to be too costly.

8. *Camouflaging:* In military applications, each target is controlled by an intelligent adversary who will try to evade attempts to be recognized. In addition to the already existing target variations, the ATR system will have to negotiate camouflaging attempts.

9. *Sensor limitations:* Usually, it is desirable to use ATR systems that can work with sensors that are easily available and simple. So, it is often the case that specific sensors cannot be tailor-made with the design objective in mind. Instead, available sensors are used and hence efficiency is compromised since the resolution and other image parameters cannot be chosen to the best ability.

10. *Architectures:* Since ATR requires a huge amount of processing power, memory and database, it is necessary to design smart hardware and software architectures for ATR. With the advent of VLSI technology, current chips are able to provide more processing power on a smaller wafer of silicon.

1.2.4 Performance Metrics

Performance metrics constitute an important area of concern for ATR systems. In image processing applications, there have not been objective means of measuring image quality. Similarly in the ATR field, there are no concrete metrics to measure the input image complexity or the performance of ATR systems. Usually, measures like the number of edge points, entropy, uniformity and structural measures have been used to characterize the input of an ATR system. However, these measures do not provide a knowledge of the image behavior and are not very helpful in characterizing the target and the clutter present in the image. There are some probabilities that are used to denote the effectiveness of the ATR algorithm. These are defined in terms of the ability of the system to detect, classify, recognize and identify the target. There are also additional metrics to define the performance of ATR algorithms [17]. These definitions are listed below.

1. *Probability of detection:* This is the probability of correctly detecting the presence of the target amidst a background containing clutter.

2. *Probability of classification:* This is the probability of correctly determining the class of the target. For example, this means the ability to detect if the target is wheeled or a tracked target.

3. *Probability of recognition:* This is one step ahead of classification. The target is recognized as belonging to a particular sub-class. For example, the tracked vehicle is then recognized as being a tank or an armored personnel carrier

4. *Probability of identification:* This is the probability of determining the identity of the target i.e. to classify it to the smallest sub-class level possible. For example, the make of the tank is identified in this stage.

5. *False alarm probability:* This is the probability of false detection. It is required in practical applications that this probability value remains low.

6. *Signal to Noise Ratio (SNR):* It is defined as the ratio of difference between the target and background intensities to the background intensity. If I_b represents the intensity of the background and I_t represents the intensity of the target, then the SNR is given by $(I_t - I_b)/I_b$

7. *Receiver Operator Curve (ROC):* It is a plot of true positives versus false positives i.e. a plot of detection rate versus the false alarm rate. This graph is plotted as a function of the Signal to Noise ratio.

8. *Confusion matrix:* The confusion matrix is a two dimensional array that lists the number of times a particular target was classified under various classes. By definition, the confusion matrix is always a square matrix. The diagonal entries of the confusion matrix indicate the correct classification values. For example, in a case containing possible targets as tanks, armored carriers and trucks, the confusion matrix lists the number of times a tank was classified as a tank, an armored carrier and a truck. In case of a two

target system, the confusion matrix lists the probabilities of true and false detections.

9. *Consistency:* This metric denotes the consistency of the classification process. For scenes with similar content, it measures how often the classifier classifies it under the same class. This gives an idea of the clutter dependence of the classifying algorithm since the difference between these scenes is caused by the noise of the atmosphere or the sensor.

It is interesting and is of practical importance to compare the performance of ATR systems and humans. The results in Table 1.2 are those of an experiment which was performed in low-clutter, forced choice environment [17]. The term forced choice indicates that nearest match was returned for any given target, that is, the humans were forced to make a choice. Though it is seen that the performance of the ATR systems is inferior compared to humans, the use of ATR systems is justified by the fact that humans cannot perform repetitive tasks efficiently over a long period of time. It is known that the best performance can be obtained by using aided ATR. This is justified by the data in the table which indicates that humans and ATR systems have similar detection rates, but the ATR system cannot classify the targets as accurately as humans. The last two columns give the probability of eight class and three class classification for ATR systems and humans.

Performance measurement is not entirely dependent on the information content of the image. For example, if the input scene has been captured from a high clutter

Table 1.2: Comparison of humans and ATR systems in target classification [17]

Classifier		TP	FP	P(8 class)	P(3 class)
ATR	Max prob.	0.869	13.323	0.353	0.732
ATR	Min prob.	0.604	3.532	0.268	0.541
ATR	Mean prob.	0.688	8.195	0.289	0.705
Human	Max Prob.	0.833	0.9	0.814	0.798
Human	Min Prob.	0.52	0.017	0.298	0.343
Human	Mean Prob.	0.683	0.234	0.586	0.663

Table 1.3: A new approach to ATR algorithm evaluation [1]

Ground Truth	Panel	ATR	Current practice	New practice
T	T	T	Detection	Detection
T	T	N	Miss	Miss
N	N	T	False detection	False detection
N	N	N	No change	No change
T	N	N	Miss	No change
N	T	T	False detection	Detection
T	N	T	Detection	Stop
N	T	N	No change	Stop

environment, the ATR system cannot be expected to perform as well as in a low clutter image. Hence, comparing the performance of the ATR system from these two scenes which have similar information content is not fair. Though the scenes have been captured for the same target, the information retrieved from the scenes differ. So, when the result of the ATR system is compared against the ground truth, it may seem that the consistency of the system is less. A new approach has been put forward to measure the performance of ATR systems in [1]. The following Table 1.3 shows this method of performance evaluation. T represents correct labelling of the target and N represents incorrect labelling of the target. In this method, the performance of the ATR systems is not compared against the ground truth. Instead, a panel of experts classifies the same image, and the performance of the ATR system is compared against that of the panel. Hence, the comparison is now against human performance which is what ATR systems seek to replace.

It is seen that though the ATR result is only required to conform with the result of the panel. In case the panel result and the ground truth are the same but the ATR result is different, some corrective measures have to be taken to change the output of the ATR system. However, when the ATR system and the ground truth are in agreement and the panel differs, no changes have to be made to the system. This ensures that the ATR system performs better than the panel. This approach is quite similar to the implementation of the Behavior-Knowledge Space (BKS) algorithm for data fusion. This algorithm will be discussed in later

chapters of this thesis.

1.2.5 Techniques for ATR

Early techniques for ATR were based on heuristic methods. Initially in the early 1980s, target detection was performed using the contrast box method. An arbitrary box was drawn around the RoI and the contrast of the RoI was compared to that of the background. The contrast box target acquisition technique [22] can be described as

$$C(i,j) = \frac{(\mu_t - \mu_b)^2 + \sigma_t^2}{\sigma_b} \tag{1.1}$$

where μ_t, σ_t^2 are the mean and variance within the box; μ_b, σ_b^2 are the mean and variance within the border.

Once target acquisition was performed, the next stage was to segment the RoI and classify it. This was performed using edge-detectors, broken edge connectors (using morphological operators), binarization, feature extraction of foreground and classification. The techniques that were used for classification included methods like Bayesian, k Nearest Neighbor or Parzen window methods. The performance of this generation of ATR algorithms was not very good, especially in the presence of clutter and detection rates did not cross 70% [17]. The low performance of these methods can be ascribed to the method of choosing the features and processing algorithms. Scene information, image formation physics and other intuitive information that is used in biological recognition systems were not used. Methods were chosen for individual processes, and the best of these techniques were put together without taking into consideration the application being studied. In the late 1980s and early 1990s, knowledge-based systems and template-matching systems began to be used. RoI detection and target classification were the two major stages of these techniques. Filtering techniques were used for the former and template matching was performed for the latter. This increased the detection rates to the 80% level in medium and low clutter environments [17] though the false alarm rate was still high. Recently, adaptive learning and model based approaches are being tested since they try to mimic human clas-

sification methods. Multisensor fusion techniques have been introduced to boost the classification rates to even higher levels.

One method used to study ATR performance and possibly improve it is using artificial imagery. A synthetic image is created and submitted as the input to an ATR system [1]. This helps the programmer to study the response of the system to a known input. The same image is also given to a panel of experts and their output is compared against that of the system. Unlike normal images, the response of the panel does not indicate the ground truth. Here, it is merely a question of which labelling is superior – the expert panel or the system. The advantage of this method is that images are not required to be provided from the field in order to train the system. Hence, huge cost savings is achieved here. In another technique, test imagery data is generated using real world images [22]. These images are manipulated to represent variations like different resolution, range, terrain, target aspects, etc. Bilinear interpolation technique was used to achieve these perturbations in the image. The ATR algorithm is expected to classify the target to the same class irrespective of the changes in the image.

The ATR algorithm classifies the feature vector of the target based on some *a-priori* knowledge. This knowledge is usually through supervised or unsupervised learning. In the learning process, the ATR system is fed different inputs and the system parameters are adjusted so that the correct output is obtained. Thus, at the end of the learning process, the system is expected to be modified enough from its initial configuration so that it can present correct outputs for similar inputs. Usually, the system is trained with the help of a domain engineer or knowledge engineer. There are three areas to which learning can contribute in ATR systems [24]. The first is in the initial acquisition of the domain theory. The second area to which learning can be applied is the usage of already known domain theory for new scenarios. This could include using the domain theory to detect targets in new weather conditions or different time of day than was present while developing the domain theory. This process is called *transfer*. The third area where learning can be applied is in the training of the system to learn a new feature. For example, adding a new feature that can help to detect the target even in low visibility. This is different from the transfer process since this involves

the addition of a new feature to the domain theory and not just adapting the already existing domain theory for a new scenario. Typically, in ATR systems, supervised learning occurs and a domain engineer or knowledge engineer is present when the system is being trained. It is preferred to use the contextual data and have adaptive algorithms that can learn from previous experiences. The major objectives of learning in ATR systems is to be able to detect and recognize the target consistently with the ability to accommodate new features and targets in real time and the ability to alert the human operator when a new target is detected.

Learning in ATR systems is usually implemented using Artificial neural networks (ANNs). These networks try to model the neural networks in the human brain. ANNs have various nodes called neurons that are densely interconnected at different layers. Each neuron has many inputs and one output. The neurons between different layers are connected using weighted links. The output of the any node, and the network itself, is a number between zero and one indicating the confidence of the classification. The node is associated with an activation function that is responsible for providing outputs based on the inputs received. A network may have more than one output neuron, one for each class of targets, and the classification is then done based on the neuron having the highest output value. During the learning or training process, inputs are provided to the ANN and the output is continuously monitored. The weights in the networks are continuously changed until the desired set of outputs is obtained for the inputs presented to the network. When this is done, the ANN is said to be trained and is ready to be used for target classification.

Learning in neural networks can be supervised or unsupervised. In the former, a human operator assists the network during the training stage and monitors the weights of the network until the desired outputs are obtained. Examples of supervised learning algorithms include the delta rule and the backpropagation rule. In case of unsupervised learning, the weight adjustments is done by the system itself. Examples of this learning include Kohonen learning. Feedforward and backpropagation neural networks are commonly used. The difference between them is that in the latter, there is a path from the output to the input providing

a feedback so that learning is faster. The advantage of using neural networks for learning is that they can perform unsupervised learning and lend themselves naturally to recognizing typical ATR targets. However, they have a long training time and hence cannot be retrained to accommodate new features and targets during run-time. Also, they are highly sensitive to noise and cannot identify new targets since they are minimum distance classifiers.

Explanation-based learning (EBL) [24] is a deductive learning process in which the system tries to learn from the current scenario. An EBL system is presented with four different inputs. The first is the *training sample* or its corresponding features. The second input is the *goal concept* or the target. The third is an *opearationality criterion* which describes which features are useful in identifying the target. The last input is the *domain theory* that describes the relation between the input and the output. After presenting these inputs to the system, it is allowed to learn and identify the important discriminants that allow it to correctly classify the data. In this method, the system can learn from different scenarios on its own and hence can react better to new situations. The disadvantage of EBL is that generating the training set is difficult. Also, it cannot identify a new target.

Adaptive algorithms for the image processing techniques like segmentation and feature extraction will be helpful in achieving better results. These blocks will help the system to learn parameters and concepts from the training and testing data provided. The ability to adapt adds robustness to the system. Another advantage of adaptive systems is that the system need not be retrained if it can adaptively learn on-the-fly.

Recently, model-based techniques are being tried for ATR [9]. Model-based techniques can be data-driven, goal-driven or model-driven while normal techniques are usually only data-driven. Model-based vision (MBV) paradigm of ATR falls in between the two other categories – Prescreen, Segment and Classify (PSC) and Matched Filter (MF). In PSC, no *a-priori* information is used and RoI detection is performed without any information on target shape. MF is the other extreme in which the image is compared against a template at all possible target-like shapes. MBV is the intermediate category in which PSC is performed to reduce the target area and then MF techniques are used for actual target de-

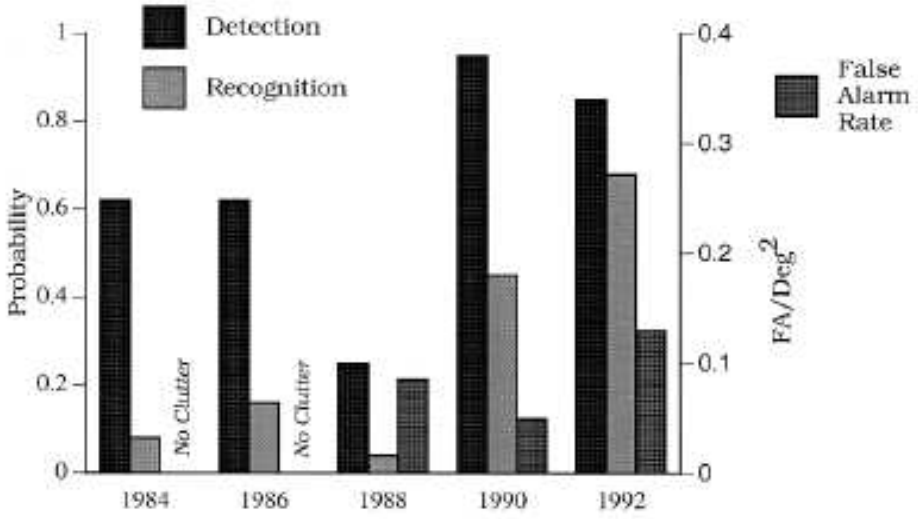


Figure 1.4: Performance of ATR over the years [17]

tection. It can be seen that MBV is the category under which most ATR systems fall. The idea of model-based systems is not only to model the target, but also form models of clutter, noise, sensors, background, heat flow, atmospheric physics and countermeasures.

1.2.6 Growth of ATR

Over the years, with the advance in technology and algorithm development, the detection, recognition and classification rates in ATR have improved. The following graphs follow the trend of ATR in different ares. Figure 1.4 gives an indication of the performance of ATR algorithms over the years. The false alarm rate for the first two years is zero, since a no clutter environment was used. The projected growth of processor computation rate against the years for single and parallel processors is shown in Figure 1.5. Figure 1.6 shows the feasible progression of ATR technology.

Other new developments in the ATR field include fusion of data from different sensors. As previously discussed, collateral information helps in detecting targets better. Hyperspectral and multispectral sensors provide information over different

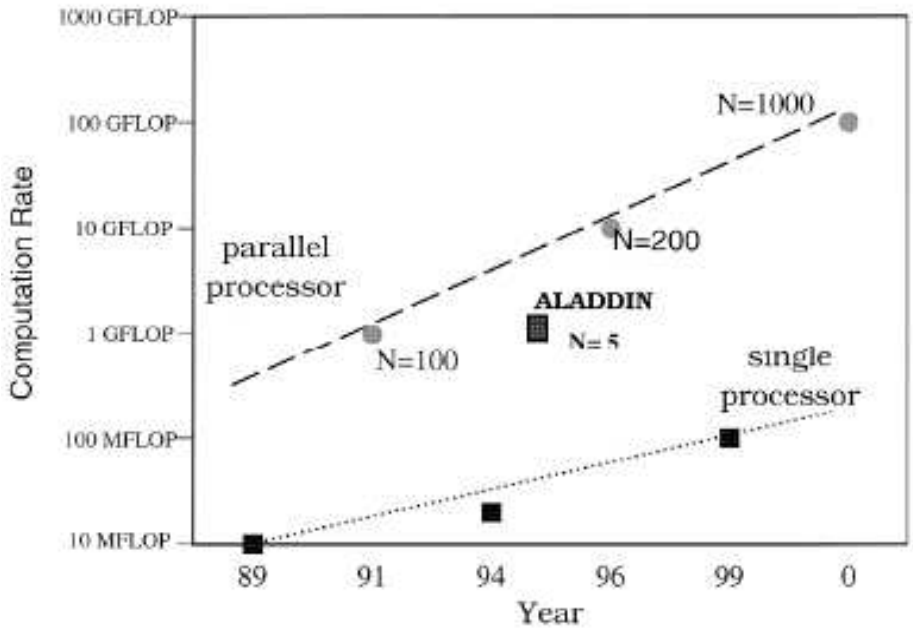


Figure 1.5: Processor computation rate versus chronology [17]

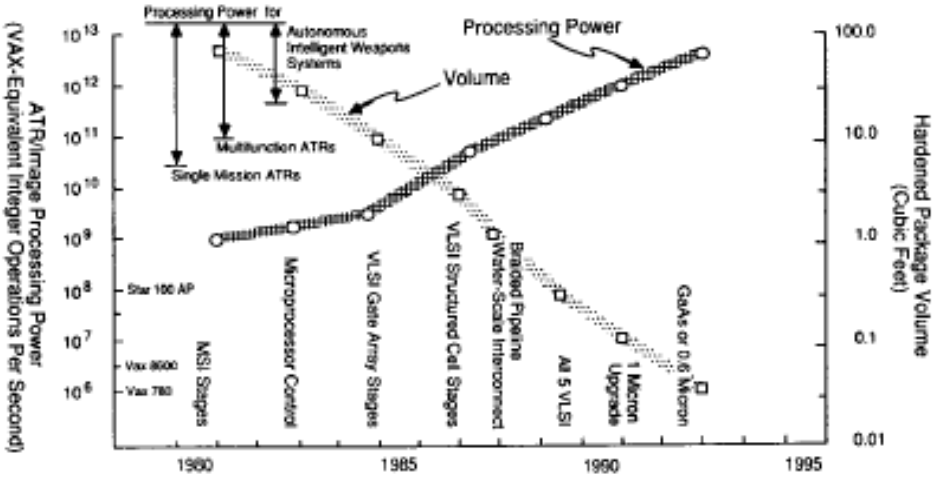


Figure 1.6: Progression of ATR processing technology [4]

spectral bands, hence different materials can be sensed based on the wavelength that they are sensitive to. In such cases, the emphasis is on identifying the spectral bands that are useful and reducing the amount of redundant information transmitted from the sensors. Data fusion is also performed to accommodate information from different sensors. Fusion can be performed at various levels, like pixel fusion and decision fusion. As mentioned in this report, current performance of ATR systems is still far from human performance. Detection can be performed accurately for low and medium clutter environments. However, new targets and scenarios continue to be a problem to ATR systems. Utilizing new sensors and information may hold the key to improving ATR system performance.

1.3 Data Fusion

Data fusion is another instance of imitating biological behavior to incorporate intelligence in automated systems. Human recognition of many objects and events is typically based on fusing visual, auditory and perceptible information. This enables robust decision making. Animals also regularly perform tasks based on information from multiple sensors. For example, multi-sensor integration has been recorded in pit vipers and rattlesnakes [21]. Infrared information is sensed by the pit organ and visual information is sensed by the eyes of a rattlesnake. The optic tectum of the rattlesnake is responsive to both these data and some neurons respond to different combinations of infrared and visual data.

Data fusion combines diverse techniques like statistical signal processing, pattern recognition, artificial intelligence and information theory to derive better decisions than stand-alone sensors, though exposed to the same target data. Multi-sensor fusion is differentiated from the more general aspect of multi-sensor integration by Luo in [21]. The latter involves the integration of multiple sensors at different levels of the system architecture. Fusion is seen as mathematical or statistical issues that are involved in the actual combination or fusion of information from multiple sensors.

In systems with multiple sensors, fusion is a natural method of combining information or decisions of the different sensors. With the advent of sensor net-

works and distributed processing paradigms, data fusion techniques are now being widely implemented to improve the classification of different processes. This section introduces the concept of data fusion and explains the various issues that have to be dealt with in fusing information in a sensor network.

1.3.1 Distributed Sensor Networks

Microprocessor and silicon technologies have been rapidly improving in accordance with Moore's Law. This has triggered an era of cheaper sensors with more processing capability. From the days of tracking events using single and high-performance sensors, it is now seen that using multiple sensors, albeit with lesser processing capability, can help in obtaining better detection and classification rates. The major reasons for the spurt in distributed sensor networks (DSNs) are the relatively low cost of sensors, inherent redundancy capabilities, presence of high speed communication networks and higher processing power [21].

DSNs can be setup using two different paradigms – centralized and decentralized. In centralized DSNs, each sensor communicates the information that it has sensed to a central processor. The fusion of disparate information is performed here. Decentralized DSNs perform local operations on the sensed data at each sensor. The results of the processing are sent to the next sensor until it reaches the *fusion center*. Thus, intelligence is present at every node of the network. The centralized processing scheme, though simple in implementation, has many disadvantages. It may not perform well in cases when the coverage area of the network is too large, especially if sensors have non-overlapping areas of coverage. Also, it places a heavy demand on the bandwidth of the communication network since all the raw data has to be transferred to the central processor. The decentralized DSNs have lower bandwidth requirements, faster response times, reduced cost and increased reliability. Hence, the current trend is to use distributed processing paradigms in sensor networks.

One of the important issues in setting up the sensor network is to decide on the network topology. This defines the positioning of sensors and the flow of information between them. Typical topologies that are used are serial, parallel with

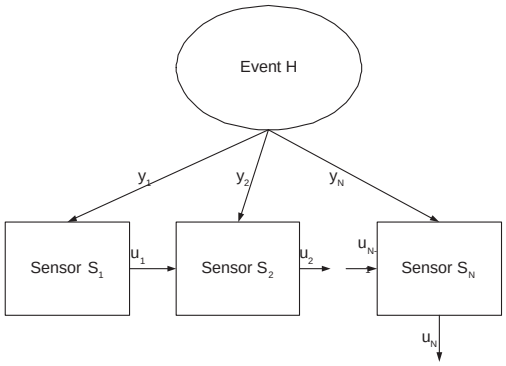
fusion center, parallel without fusion center and tree topologies. Consider a sensor network in which the event H is sensed by N sensors. The input to each sensor $S_1, S_2, \ldots, S_N$ is represented by the variables $y_1, y_2, \ldots, y_N$ respectively. The output of every sensor is given as $u_1, u2, \ldots\ u_N$. Figure 1.7 illustrates the different serial and parallel network topologies used in sensor networks. The SFTB is used only for data collection and hence is a network that uses centralized processing. However, addition of some local processing at each node can quickly convert the SFTB to a decentralized network with a parallel topology with a fusion center.

1.3.2 Types of Sensors

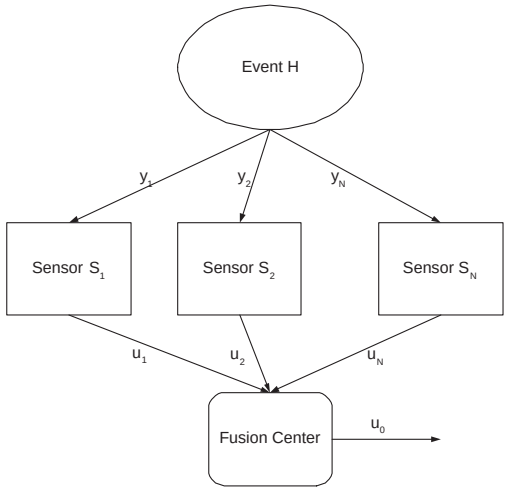
To better understand issues related to data fusion from different sensors, it is necessary to define the different kinds of sensors [3]. Firstly, the concept of *abstract sensor* is introduced. This is defined as the sensing aspect of the sensor and is removed from the idea of the physical sensing device which are called *concrete sensors*. This abstraction helps in identifying the theoretical limits of the sensors and their sensitivity without being restricted by hardware limitations and particular kind of sensors. Based on the interaction of different sensors in the network, sensors can be divided into different types.

Complementary sensors These sensors provide a better and complete picture of the sensing area when taken together. They do not depend on each other directly. An example of complementary sensors would be a network of cameras in which each camera covers a different area. Each camera provides visual information of some area, but when taken together, they give a more complete and generalized picture of the entire region. Fusion of information from complementary sensors is simple since it usually involves appending data from different sensors.

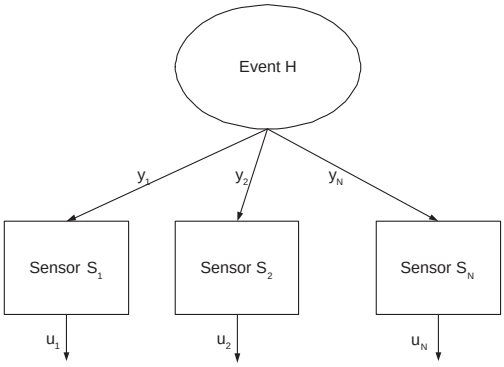
Competitive sensors In this case, each sensor provides information about the same event. Since they are providing information about the same phenomenon, the sensors are in competition with each other as to which sensor's decision must be believed. These sensors need not necessarily sense the same



(a) Serial topology



(b) Parallel topology with fusion center



(c) Parallel topology without fusion center

Figure 1.7: Serial and parallel network topologies in sensor networks

information, i.e. the network can consist of visual, infrared and RADAR sensors. The important criterion is that they derive their information from the same event.

Cooperative sensors The network consists of sensors which provide information in such a way that when individual data of each sensor is taken together, it provides information unavailable to the individual sensors. Complementary sensors can extract final decisions, though less reliably, from the data of a single sensor. In the previous example, video information from a single camera can still be used for surveillance. However, in cooperative sensors, final decision cannot be made from the sensed data of any single sensor. For example, consider a sensor network consisting of pressure sensors along a line. Each sensor provides information about the pressure at each point, but the network can derive information about pressure changes along the line.

Individual sensors Those sensor networks that do not match any of the above categories are termed individual sensors. The information sensed by each sensor might not be fused together in the strictest sense. Sensed data that have no relevance to each other can be obtained and stored together. This is treated as a separate case since this frequently occurs in practise.

1.3.3 Benefits and Limitations

Data fusion provides many qualitative and quantitative benefits when performed properly [3, 14]. These benefits are listed below.

1. The problem of inaccuracy of single sensors is successfully mitigated. The classification and decision making of the system is limited by the accuracy of the information sensed by the sensors. Since more than one sensor is used, the inaccuracy of the sensor makes a lesser impact.

2. Using multiple sensors also improves the sensitivity of the system to noise. The fusion of decisions from different sensors enables robust performance even in the presence of noise.

3. Reliability of the system is improved since the decisions are more robust and accurate. When more than one sensor infer a similar decision, the chances of the final decision being wrong are lesser. The ambiguity in decision making is removed, or at least reduced.

4. Multi-sensor fusion reduces the cost of the system since cheaper sensors can be used to replace single, high-performance and costlier sensors.

5. Extended coverage of the region can be obtained using multiple sensors. This increases the surveillance area and detection chances.

6. Presence of multiple sensors ensures graceful degradation of the system. The sensing system will not die out suddenly and unexpectedly, and can continue to operate even when one or two sensors in the network fail.

7. Using many sensors improves the survivability of the system since it is more resistant to chances of failure due to enemy action or natural phenomena. The observability of single sensors is limited.

Multi-sensor networks have these advantages at a price. Their very nature implies that information from multiple sensors have to be combined to arrive at the final conclusion. This translates to an increased need for computational power and intelligence for decision making. Also, the information overload in the system has to be successfully dealt with in order to perform effective sensor fusion. The system architecture must provide means to manage the excessive information. Sensor fusion need not necessarily provide better results than single sensor systems. Though many sensors are used, this does not imply improved sensibility, since the limitations in sensing exist in every sensor. Using many sensors cannot substitute into using a single, robust and error-free sensor. Errors in sensing cannot be negotiated by fusion since the information provided to the fusion algorithm itself is flawed. Again, there is no perfect data fusion algorithm that can perform optimally under all situations.

1.3.4 Applications

Data fusion architectures and algorithms are used in a variety of applications which include industrial, medicinal, military, remote sensing, aeronautical and other areas [3, 14, 21]. In general, any signal processing application can use data fusion techniques. Military applications include command and control of battle management systems, target detection and tracking applications, object recognition systems, border control, surveillance and strategic warning systems. Law enforcement agencies can also use fusion algorithms for issues like traffic control, border surveillance and transportation applications. Remote sensing using multi-spectral and hyper-spectral sensors use data fusion techniques to locate and identify mineral deposits and study environmental conditions. In addition, fusion has important applications in aeronautical, manufacturing and industrial areas for processes like material handling, part fabrication, inspection and assembly lines. Automatic fault diagnosis and identification and obstacle location can also be performed by such systems. Medicinal applications of data fusion algorithms include diagnosis of diseases, location of tumors and physical condition evaluation using sensors placed on, in or around the body.

1.3.5 Hierarchy and Levels of Data Fusion

The input to sensor systems consists of raw sensed information and the final output is usually a specific estimate of the identity or location details of the target or event. The transition of data within the system from raw input to the processed output is interesting to study. It gives an idea of the techniques and inference levels that are used in the system architecture. This section describes the different hierarchical inference levels and the Joint Directors of Laboratories (JDL) data fusion model [13, 14].

At the lowest inference level, the raw sensor data is used to estimate or detect the presence of an entity or target. This is the first step in trying to obtain the final desired output. The second level of inference would be to determine the position or velocity details of the target using multiple data. This estimate is usually given in terms of six vectors, three for position and three for velocity

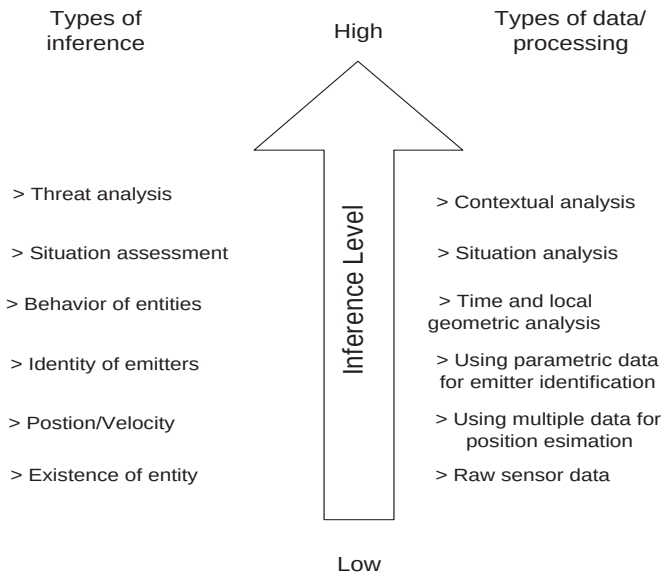


Figure 1.8: Hierarchy of data fusion

information of the target. This is the minimum requirement to represent the state of the target or event that is used to predict the future state. At the next level, the identity of the target is determined. This involves estimating the target class or sub-class. Pattern recognition techniques and statistical methods are used to convert the parametric data into the target class estimate. Higher inference levels involve assessment of the situation, behavior of the entity and threat analysis which usually use heuristic methods like templating or expert systems [14]. Figure 1.8 shows the different inference levels and the types of data and processes used for them.

Different kinds of models can be used to represent and study the data fusion process. These include functional models, architectural models, process models, formal models and mathematical models. One of the important models that is used is the Joint Directors of Laboratories (JDL) data fusion group's model of data fusion. This was developed to reduce the confusion in terminology used by different groups of data fusion system developers. This is a functional model, which means that it shows the primary functions, relevant databases and the interconnectivity between blocks to explain data fusion. It incorporates the levels

one, two and three of the data fusion group's terminology. Figure 1.9 shows the JDL data fusion model. The figure on the top is the first version of the data fusion model. The block diagram at the bottom is the revised JDL data fusion model.

Both the JDL fusion models are almost similar. The objective of the revision in the model was to provide a useful categorization to represent different problems and to maintain the terminology. The different levels of processing in the first data fusion model as explained as follows [14].

Preprocessing Information overload is a common problem in data fusion systems. Preliminary filtering provides an way to control the flow of data into the fusion model. Data can be grouped into categories based on the sensor type, target signature or location so that subsequent processing of data is simpler.

Level 1 Processing This is the object assessment level. Data from different sensors is fused together to obtain the details of the target like position, velocity and identity. This uses pattern recognition processes and statistical methods. This level of data processing consists of four distinct processes – data alignment, data association, tracking and identification.

Level 2 Processing This seeks a higher level of inference from level one processing. This level deals with situation refinement. Reasoning methods and heuristic techniques are used to determine the relationships among the different entities found on scene and interpret the meaning of the observations.

Level 3 Processing Threat refinement is performed at this stage. The current observations and situations are projected into the future to find possible threats. The fused data is observed from the point of view of the adversary. This helps in identifying possible threats much more easily. This is an inferential process.

Level 4 Processing This is a meta-process level. As can be observed from Figure 1.9, this block lies partially within and partially outside the data fusion domain. This is a process refinement level that monitors the processes to improve the results.

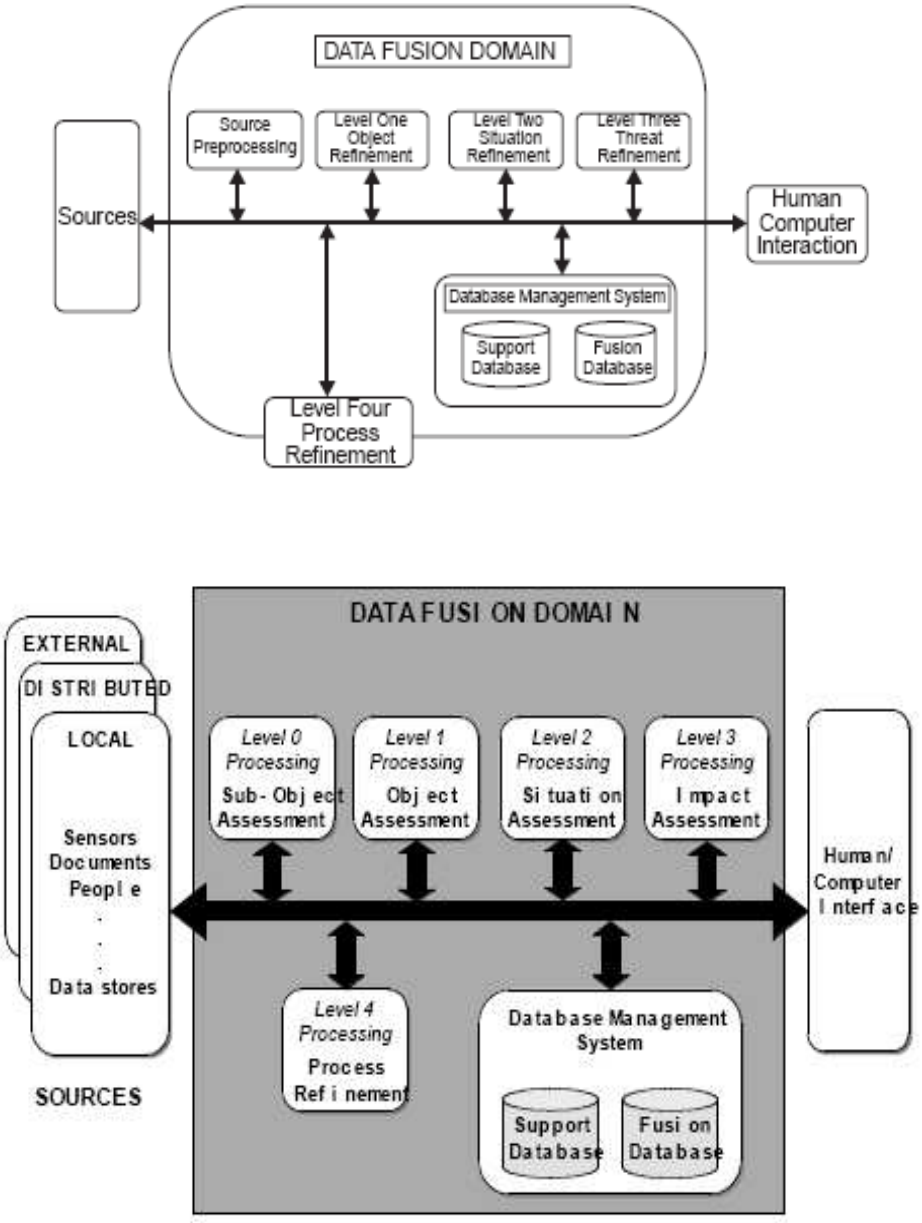


Figure 1.9: JDL data fusion model [13]

Level 5 Processing This stage is termed as cognitive refinement. This level was included to incorporate the interaction between the human-in-the-loop and the fusion system.

Database management systems The creation and maintenance of the support and fusion databases is performed by these systems. The results of different levels of fusion are recorded in these databases.

1.4 Thesis Contribution

This thesis is primarily motivated towards providing increased classification for the civilian target recognition problem. Towards this goal, the recently developed fusion techniques are used. The thesis thoroughly reviews the existing methods and issues related to ATR and fusion. The important contribution of this thesis is that it uses decision fusion methods at different architectural levels to improve the accuracy of the ATR model. The image processing routines used in this thesis are basic and simple. Advanced methods are now available that can provide increased pattern recognition capabilities. However, the final classification accuracy is raised to high levels using fusion techniques. The presence of the fusion hierarchy also compensates for using inexpensive infrared and color cameras in place of more sophisticated sensors. The targets of interest in this thesis are not military targets, but civilian vehicles. Successful classification of civilian targets can open a wide variety of applications for ATR in our daily lives. Another important contribution is the provision of a single modular framework for detecting, segmenting and identifying different targets. Complex algorithms for image segmentation, feature extraction and data fusion can easily be incorporated merely by writing new routines and including them in the main program. The system works in a cost-effective and computationally simple manner. The classification process uses little computation time (less than 0.5 seconds when all the frames are present for classification) and this makes the system capable of real-time performance.

1.5 Thesis Outline

This thesis is organized as follows. The basics of ATR and data fusion have already been covered in this chapter. The hardware and software architecture that are used for ATR are presented and discussed in Chapter two. This chapter describes in detail the SFTB and the positioning of nodes to capture images of the targets. The method of data storage and operating conditions are discussed. The software architecture shows the data flow from the sensors to the final decision making stages. For any ATR system, image processing is very important since this comprises the "eyes" of the system. The image processing routines and techniques are presented in Chapter three. The outputs of different image processing operations are shown. Once frame-level classification results are obtained, the next process that is performed is data fusion. This comprises Chapter four. Temporal and BKS fusion are explained in detail and the fusion algorithms are discussed here. Chapter five consists of the results of the different experiments that were performed and discusses these results. Finally, the thesis ends with the conclusions and scope for future work in Chapter six.

Chapter 2

Data Acquisition

In the previous chapter of this thesis, the research problem and background were introduced. The motivation of the thesis was explained and the objectives were stated. To achieve these objectives, a hardware and software architecture must be designed and implemented to convert real-world phenomena and data into a form that can be used for automated classification. This architecture will contain different units to perform functions like sensing, data acquisition, feature extraction, feature vector database formation, image processing techniques and pattern recognition techniques. A general overview of the different processes involved in creating the database from the scene is shown in Figure 2.1. Image data is acquired for this work using the SFTB. The description and setup of the testbed, node placement issues, target details, conditions of data capture and formation of the feature vector database are explained in this chapter.

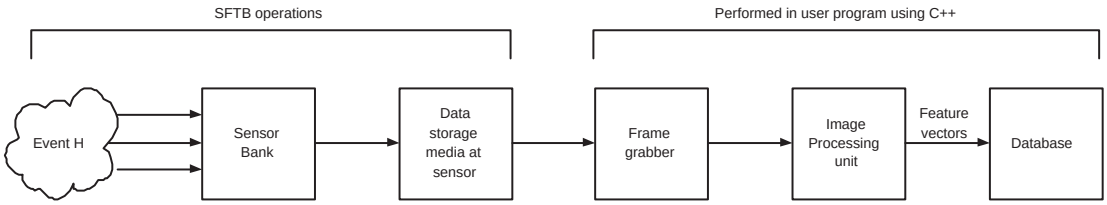


Figure 2.1: Database creation from the scene

2.1 Sensor Fusion Testbed

The database for this research work was collected using the Sensor Fusion Test-Bed (SFTB). The Night Vision and Electronics Sensors Directorate (NVESD) selected Northrop Grumman Mission Systems to develop the SFTB. The testbed contains acoustic and imaging sensors that capture target information. Testbeds are strategically deployed so that the sensors can be placed along the roads on which the targets will travel. The information thus sensed is converted into a database of feature vectors which is used for classification. The objectives of creating the SFTB are two-fold. Firstly, it creates a framework to collect a set of acoustical and imaging data for testing and developing moving target Automatic Target Recognition (ATR) data fusion algorithms. Another important objective of developing the SFTB is to develop the means to record and collect the ground truth information.

The architectural and functional details of the SFTB are discussed in [6]. The system consists of three nodes and one base station. The SFTB has two modes of operation – an attended acquisition system or an autonomous data collection system. The latter mode was used for the data used in this thesis. The SFTB also has a software suite that contains a framework to perform ATR and target tracking. The Multiple Signal Classification (MUSIC) algorithm is used to determine the direction of arrival of the target and is a built-in function provided with the SFTB. New algorithms can be easily added to the system since each application is responsible for its own operation and integration time is low.

2.1.1 Nodes

The testbed contains three data collection nodes. Each node has an imaging sensor and a acoustic sensor array consisting of seven acoustic microphones in a hexagonal pattern. Among the three nodes, two of them (nodes 1 and 3) have uncooled IR cameras and the other (node 2) has a color camera. Although node 2 has a color camera, the images that are obtained from this node are grayscale images. The objective of using the non imaging sensor in the nodes is to detect the target. Once a target is detected using the acoustic data, the direction of

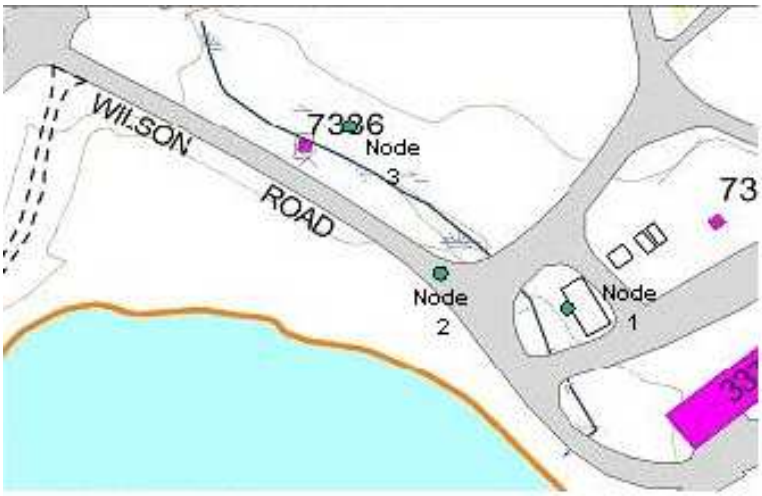


Figure 2.2: Position of different nodes [7]

approach is estimated using the MUSIC algorithm, and the imaging sensors can be turned on. The three nodes are deployed in the testing area so as to enable roadside monitoring of the targets. The nodes are all stationary and point towards the south-east corner of the map shown in Figure 2.2.

The nodes are in constant communication with the base station which controls their operation. The data collected at the nodes correspond to four types of information from the scenario – acoustic, infrared, grayscale and meteorological. The acoustic, infrared and grayscale information constitute the target information captured and meteorological information is used to generate the ground truth information. The following data collection equipments are used. The first three components are used to collect target details while the last two are used to record the ground truth information.

- Knowles (Emkay) BL-1994 ceramic microphone and Burr-Brown PGA103 programmable gain amplifier (preamp for the microphone)

- Indigo Alpha uncooled micro bolometer FPA camera
  The spectral sensitivity of this camera extends from 7.5 to 13.5 microns. The camera has a horizontal FoV of 40 degrees and can operate in a temperature range of 0 to 35 degrees Celsius. The image are captured at a frame rate of

30Hz and have a size of 160x120 pixels. Each pixel is represented using 16 bits where the most significant nibble is set to zero.

- Pulnix TMC-7DSP color video camera
  Images are captured by this camera at a frame rate of 30Hz. The image size is 160x120 pixels, which is also the size of the images captured by the infrared camera. However, each pixel is represented using 8 bits per color channel. So, the color image has 24 bits per pixel. The grayscale image has only 8 bits per pixel.
- Garmin global positioning units
- Texas Weather Instruments weather station

These sensors are illustrated in Figure 2.3. In this research work, only images from nodes 2 and 3 have been used. This is because node 1 is positioned further away from the road so that the target area in the images captured by node 1 is very small. Image segmentation of images from node 1 does not yield substantial target details to aid in recognition. The audio signals are not used.

2.1.2 Targets

Seven non-military vehicles are used in this experiment. These are listed below.

- *Target 1* Honda CRX (Car)
- *Target 2* Chevy Cavalier (Car)
- *Target 3* Toyota Pickup (Light truck)
- *Target 4* GMC Pickup (Light truck)
- *Target 5* Xterra (SUV)
- *Target 6* Toyota 4runner (SUV)
- *Target 7* Stake Body (Truck)

Figure 2.4 shows examples of frames captured by nodes 2 (grayscale image) and 3 (infrared image) for the seven different targets.

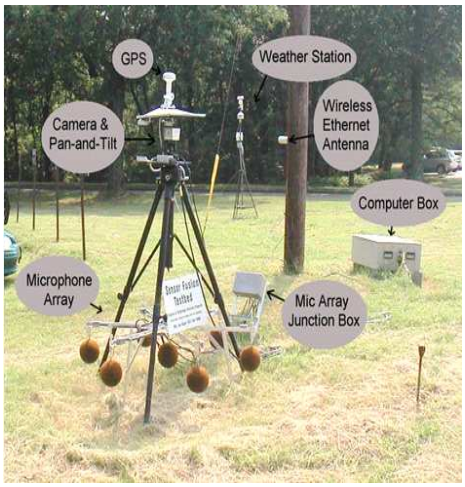
(a) Pulnix color video camera

(b) Indigo Alpha infrared camera

(c) Knowles microphone array

(d) Garmin global position unit

(e) Texas Weather Instruments weather station



(f) Node with sensors

Figure 2.3: Images of different sensors and node [6]

Figure 2.4: Examples of frames captured using infrared (left column) and color video (right column) cameras for target 1 (top) to target 7 (bottom)

2.1.3 Scenarios

There are three different scenarios present in the final database. These scenarios are labeled scenarios 1, 6 and 25. Each scenario is a variation in the motion of the target. The scenarios are described as follows.

In scenario 1, the target moves from the far end of the field-of-view (FoV) of the nodes towards the near end (south-east to north-west in the map). The vehicle travels at a constant speed and stops outside the FoV of the nodes.

In scenario 6, the target again moves from the far end of the FoV of the nodes towards the near end (south-east to north-west). In this case, the target stops within the FoV of the nodes for a brief interval of time (manual count of 10 seconds) and resumes its motion and finally stops outside the FoV of the sensor.

In scenario 25, the target moves in the opposite direction i.e. the target moves from the near end of the FoV towards the far end of the FoV of the nodes (north-west to south-east). The target stops within the FoV of the nodes and resumes its motion after a manual count of 10 seconds.

2.1.4 Operating Conditions

For facilitating data capture, the following operating conditions were maintained [7].

1. The targets move on gravel or asphalt roads.
2. Targets are fully exposed unless obstructed by roadside vegetation or other vehicles.
3. The license plate numbers on the targets are covered and are not readable.
4. The targets move at the same speed within the FoV of the sensors. If the target stops within the FoV, then it does so with a constant acceleration and deceleration, and stops within the FoV for a manual count of ten.
5. Each target travels at a constant speed of 5, 10, 15 or 20 mph.
6. The targets start and stop outside the FoV of the sensors.

7. The target motion and the number of vehicles that appear on the scene are dependent on the scenario.

8. The sensors are stationary.

9. The data collection is carried out during daytime only.

10. Data is collected for a basic time of three minutes.

2.2 Data Capture

Each node contains a local data repository in which it stores the sensed data. The data is collected from the nodes at the end of the data collection day and composed into a central database. All the data is stored in an ASCII or binary format. Infrared imagery, color imagery and acoustic signature files are stored in the binary format. These files may consist of one or more records of binary data and each block of data is preceded by a header. The ASCII format is used to store the weather information.

The infrared and color imagery data are stored in the Automatic Target Recognition Working Group Raster Format (ARF) [27]. This is a universal format that is easily readable by different applications running on different machines. The ARF format is portable, extensible, supports eleven image formats and is easy to work with. It supports multiple frame files and a wide variety of integer and floating point image formats. It uses the External Data Representation (XDR) format to represent data with the exception of one and two byte integer pixels. Basically each ARF image consists of a main header, possibly some sub-headers and footers and one or more image frames. This is illustrated in Figure 2.5

The main header contains information like ARF version number, number of rows and columns, image type, number of frames, image offset (number of bytes used by header and subheaders) and subheader flags. Each subheader contains the following information – ARF information (image source, capture rate, capture time, sensor name, sensor FoV, etc.), ARF colomap, ARF multiband information and so on. The image data follows the header information.

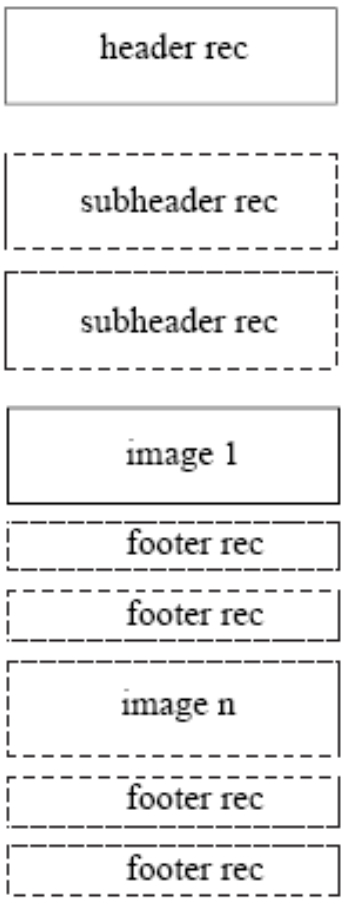


Figure 2.5: Content of an ARF file

For database creation, the individual frames have to recovered from the .arf files and converted into a machine readable format. In order to accomplish this, the ImageJ software is used. This converts the .arf files into a group of raw files. These raw files are read into the C++ program and are used for feature extraction and database creation. Since the filename of each .arf file is based only on the node ID, node number, scenario number and target number, it is possible to read in frames of interest into the C++ program when the user feeds in these information to the program.

2.3 Database Creation

The nodes 2 and 3 capture the images of targets when the targets are within their FoV. Suitable frames are selected from the images captured so that substantial target details are observed and reliable feature extraction can be performed.

Initially, the use of instantaneous frames were studied i.e. frames captured by both nodes at the same instant of time were used. However, the node positioning is such that the target area of images from node 3 is quite negligible at the times when node 2 observes the target well. Similarly, when the target is clearly visible to node 3, node 2 does not see the target. Hence, the idea of simultaneous frame fusion was dropped.

Since the idea was to fuse the results of nodes 2 and 3, it was decided to perform temporal fusion at these nodes individually and fuse the temporal results at the sensor level. Therefore, at the event level, frames captured by one node would be fused locally to obtain a reliable result for that node. When both nodes had individually decided on the target, these decisions were fused together (multimodality or multisensor fusion) to arrive at the final decision.

In order to achieve this, frames were selected from each node such that the target was as close to the camera as possible. From the speed of the targets and the positioning of the nodes, it was decided that thirty frames would be selected in which the target was as close to the node as possible. The frames are spaced out in time (one in five or one in two frames are grabbed) so as to obtain temporal data. These frames were subjected to image processing techniques so

as to extract features from the target in the image. These feature vectors were then used for classification and the classification accuracy was calculated. The following flowchart in Figure. 2.6 describes the database creation process. This process is followed at both the nodes.

The number of columns in the database depends on the number of feature vectors extracted. In our case, there are seven feature vectors (invariant moments). In addition to the features, the target number and scenario number are appended at the end of each row. Hence, nine columns are present in the database. The number of rows is fixed and depends on the number of nodes, targets and scenarios. This can be explained as given below.

- Number of nodes = 2
- Number of scenarios = 3
- Number of targets = 7
- Number of frames per target = 30

Hence, for every scenario, there are 7x30 = 210 frames in the database.
for every node, there are 210x3 = 630 frames in the database.
There are two databases – one for each node and each containing 630 frames.

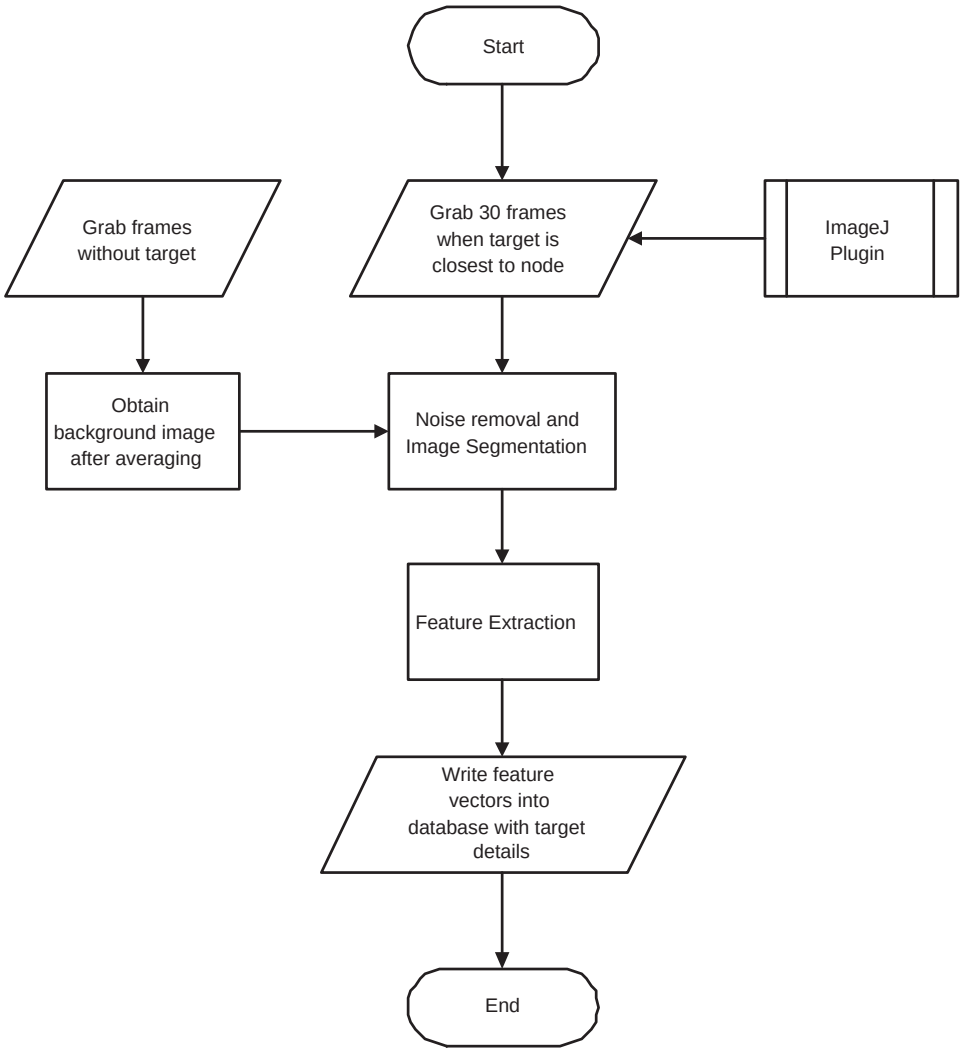


Figure 2.6: Flowchart describing the formation of the feature dataset

Chapter 3

Image Processing and Pattern Recognition

ATR is a term synonymous with image processing. Current ATR systems use imaging sensors to obtain real-world data and hence image processing and pattern recognition techniques are an integral part of ATR systems. Every recognition system consists of filtering, segmentation, feature extraction and classification modules. These modules are discussed in this chapter. In this research work, simple image processing algorithms have been implemented. The classification accuracy of a single image frame is not significantly high. Increased recognition rates are obtained using data fusion techniques.

3.1 Image Preprocessing

The data captured by any sensor is usually corrupted by a variety of noises – atmospheric noise, instrument noise, quantization error and others. These noises can be removed using low pass filters by smoothening the high frequency noise. There are different kinds of low pass filters that are used for noise removal which can be broadly classified as spectral and spatial filters. The former work in the frequency domain and the latter work in the spatial domain.

In order to remove noise from the images obtained from the SFTB, the median filter and averaging filter were used. Both are spatial filters. The averaging filter

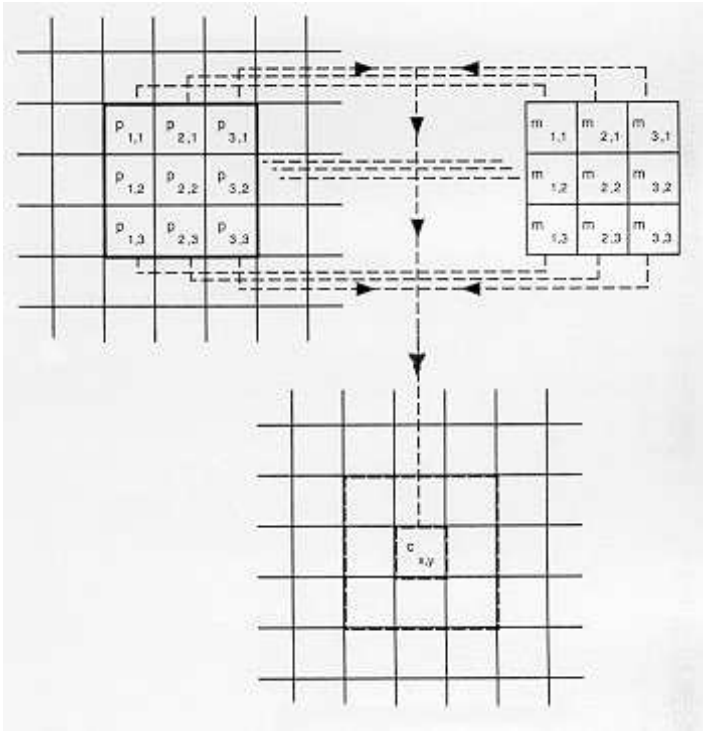


Figure 3.1: Applying a low pass filter to an image [25]

is linear and uses a filter mask for operation. The median filter is a non-linear filter. A section of the image of the same size as the filter mask is convolved with the mask to obtain the low-pass filtered value for each pixel. This is shown in Figure 3.1. The kernel on the left is the image and the one on the right is the filter mask. The weighted sum of the overlapping elements of these kernels gives the value of the output pixel.

The averaging filter is described by Equation 3.1. The numerator of this equation is a general expression representing all linear filters. When this is divided by the sum of the mask elements, it gives the averaging filter. In this filter, each output pixel value is found as the average of the sum of the overlapping pixel values. That is, each output pixel value is the average of a small neighborhood of pixels in the original image. This removes sharp transitions in the image by blurring and makes the original image less noisy. However, the averaging filter does not work well in all cases, and the blur introduced in the image is undesirable.

$$g(x,y) = \frac{\sum_{s=-a}^{a}\sum_{t=-b}^{b} w(s,t) f(x+s,y+t)}{\sum_{s=-a}^{a}\sum_{t=-b}^{b} w(s,t)} \tag{3.1}$$

where $g(x,y)$ is the output image; $f(x,y)$ is the input image of size M,N; $w(s,t)$ is the filter mask of size m,n; $a = (m-1)/2, b = (n-1)/2$; $x = 0, 1, \ldots, M-1$ and $y = 0, 1, \ldots, N-1$.

The median filter performs filtering by choosing the median pixel value in a small neighborhood of the original image and assigning this median value as the output pixel value. The output pixel value is not a weighted average, but is simply the median value of the pixels in its neighborhood. The advantage of using a median filter is that it preserves edges and does not introduce as much of a blur as the averaging filter.

Figure 3.2 shows the output results of averaging and median filters on input images obtained from the SFTB. Filter masks of size 3x3 were used for the averaging and median filters. It can be seen that the median filter can effectively remove the black strips that are present in the input infrared image. The averaging filter tries to blur this strip but is not effective in removing it. Also, the blurring of the averaging filter is clearly seen.

3.2 Target Segmentation

Once the noise has been removed from the input image, the next step is to isolate the region of interest (RoI) from the clutter. This is done using image segmentation techniques. This process divides the image into constituent parts and the RoI is alone considered for further image processing modules. Segmentation is usually based on similarities or the discontinuities in the image. For example, thresholding and texture based segmentation are based on identifying similar regions while edge-based segmentation and active contours work on discontinuities present in the image. Connected component analysis, region growing and watershed methods are hybrid methods of segmentation. In our case, segmenting the images obtained from the nodes should result in only the targets being present in the image. When this is accomplished, the features of the target can be ex-

(a) IR input image

(b) Grayscale input image

(c) IR averaging filter output

(d) Grayscale averaging filter output

(e) IR median filter output

(f) Grayscale median filter output

Figure 3.2: Output images of averaging and median filter for target 4, scenario 25

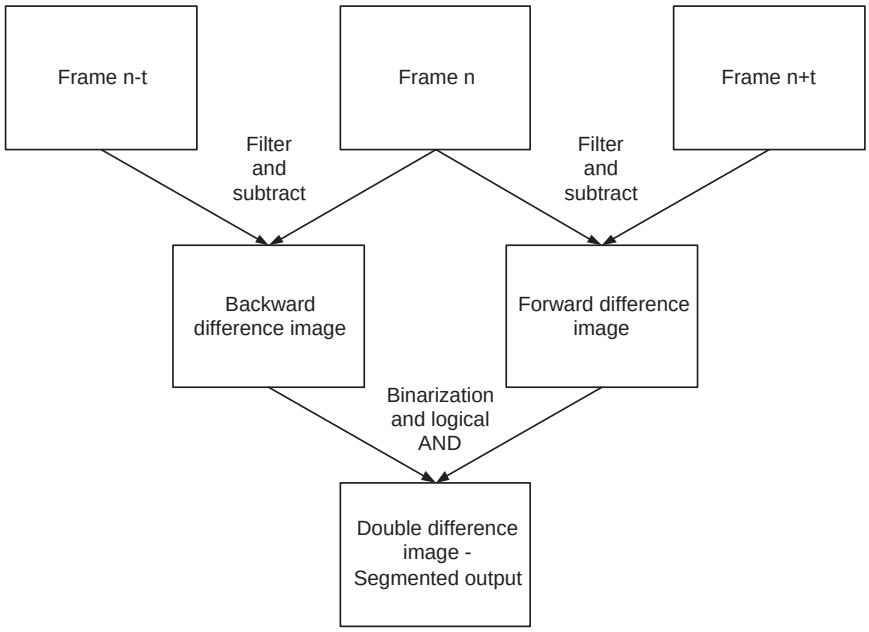


Figure 3.3: Double difference image – Motion based segmentation

tracted and used for classification. Segmentation is a very important part of any image processing application since its accuracy affects the further analysis of the image and ultimately decides on the efficiency of the system. Different segmentation techniques like motion-based segmentation and background subtraction were implemented. The best segmentation algorithm was chosen and included in the final program. Since the software architecture is modular, it is easy to substitute another segmentation routine in the process so that classification accuracy can be improved. The different algorithms and their results are discussed in the following parts of this section.

For motion-based segmentation, double difference images were used [29]. Three image frames are used to obtain two difference images. These difference images emphasize the regions where movement is observed since they show the difference of two successive frames. The AND operation is performed on these two difference images to obtain the final segmentation result, which is called the *double difference image*. This process is illustrated in Figure 3.3.

Another simple segmentation technique is background subtraction. This can

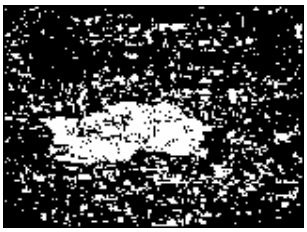
be used in situations where the camera is stationary and the RoI moves against a fairly constant background. The background image is captured and stored when the target is not present in the FoV of the camera. The segmented result is obtained by subtracting an image containing the target from the background image. Since the background is fairly non-changing, the difference image will consist of the target alone. In order for this method to work well, the background image is updated frequently so as to reflect the current illumination and clutter conditions. Another approach would be to normalize the background and the obtained images to compensate for these variations. Practically, the background method seldom works since it is hard to maintain a constant background. This method can be used in assembly lines and conveyor belts where the background can be easily formulated. Using filtering techniques to remove the irregularities caused by small background changes can help in improving the segmented image. In our case, the background subtraction and double difference segmentation routines were followed by a neighborhood operation to improve the segmentation results and remove random noise pixels. This method is similar to morphological filtering. The 3x3 neighborhood of every foreground pixel of the segmented image is checked. Only those pixels with a preset number of foreground pixels in its neighborhood were classified as foreground pixels in the final image. This removed stray white specks and noise considerably. Figure 3.4 shows an example of the results obtained after segmentation and neighborhood operation. It can be seen that background segmentation provides a better output than the double difference segmentation technique. Hence the background segmentation algorithm was used for developing the feature vector database.

3.3 Feature Extraction

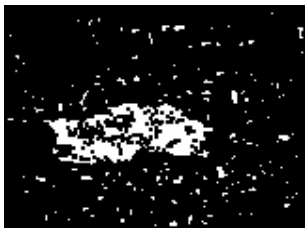
In order for the ATR system to recognize targets, it is necessary to characterize them in a way that the system would understand. For this, mathematical methods are used to extract the features of the targets. These features describe the targets uniquely and distinguish different targets from each other. Target characteristics like 2D and 3D shape, color and texture can be used to describe the targets. Shape-

(a) Input image

(b) Background image



(c) Motion based segmentation result



(d) (c) after neighborhood operation

(e) Background subtraction result

(f) (e) after neighborhood operation

Figure 3.4: Segmentation results for target 4, scenario 25

based classification uses a set of descriptors that describe target features like scalar descriptors (perimeter, area, thinness, compactness), fourier descriptors, invariant moments and boundary chain encoding. The features that are chosen are usually application-dependent.

Scalar descriptors consist of a single real number that describe a particular aspect of the target shape. The perimeter (P) counts the number of boundary pixels of the shape and the area (A) measures the number of pixels that make up the target. The thinness of a shape can be mathematically defined as given in Equation 3.2.

$$\begin{aligned} ThinnessA &= \frac{P^2}{A - 4\pi} \\ ThinnessB &= \frac{P}{A} \end{aligned} \tag{3.2}$$

Statistical features like amplitude and shape statistics can be derived from the image signature. The signature of a shape is a single dimensional representation of the boundary of the shape [11]. This is easier to describe than the two dimensional boundary of the shape. An example of deriving the signature is to record the distance of all perimeter pixels from the centroid of the shape. The amplitude and shape statistics can be calculated using this set of distances. These features define the mean, standard deviation, skewness and kurtosis of the shape. Amplitude and shape statistics are global features of the shape and they do not provide any local information. The amplitude statistics are given in Equation 3.3 and the shape statistics are shown in Equation 3.4.

$$
\begin{aligned}
\mu_{amp} &= \frac{\sum_{i=1}^{N} F(i)}{N} \\
\sigma_{amp} &= \sqrt{\frac{\sum_{i=1}^{N} (F(i) - \mu_{amp})^2}{N}} \\
\gamma_{amp} &= \frac{\sum_{i=1}^{N} \frac{(F(i)-\mu_{amp})^3}{\sigma_{amp}}}{N} \\
\beta_{amp} &= \frac{\sum_{i=1}^{N} \frac{(F(i)-\mu_{amp})^4}{\sigma_{amp}}}{N} - 3
\end{aligned} \tag{3.3}
$$

where $F(i)$ is the relative fluorescence at the ith time points; N is the number of time points.

$$
\begin{aligned}
\mu_{shape} &= \frac{\sum_{i=1}^{N} F(i)}{N} \\
\theta_{shape} &= \sqrt{\frac{\sum_{i=1}^{N} (i - \mu_{shape})^2 F(i)}{S}} \\
\gamma_{shape} &= \frac{\sum_{i=1}^{N} \frac{(i-\mu_{shape})^3}{\theta_{shape}} F(i)}{S} \\
\beta_{shape} &= \frac{\sum_{i=1}^{N} \frac{(i-\mu_{shape})}{\theta^4_{shape}} F(i)}{S} - 3
\end{aligned} \tag{3.4}
$$

where $F(i)$ is the relative fluorescence at the ith point; N is the number of time points; $S = \sum_{i=1}^{N} F(i)$.

The features that have been used to generate the database are Hu's moments. Hu developed the mathematical foundation for invariant moments in 1962 and explained their relevance to shape recognition. There are seven moments that are derived from the central moments. These moments are invariant to translation, rotation and scaling of the shape [19]. Translation invariance is achieved by normalizing the image with respect to the center of gravity of the image by using central moments. Size invariance is achieved by normalizing the image using algebraic invariants. Due to their invariant nature, they are called Hu's invariant

moments.

The moments are computed from the information of the shape boundary and the region encompassed by the shape. The following set of equations explain the procedure to calculate the invariant moments for an image [10]. Consider a two dimensional image plane, f, then the moment of order $(p+q)$ is defined as Equation 3.5.

$$m_{pq} = \sum_{x=1}^{rows} \sum_{y=1}^{columns} x^p y^q f(x,y) \tag{3.5}$$

Based on the moments calculation, the central moments are given as in Equation 3.6.

$$\mu_{pq} = \sum_{x=1}^{rows} \sum_{y=1}^{columns} (x - m_h)^p (y - m_v)^q f(x,y) \tag{3.6}$$

where $m_h = \frac{m_{10}}{m_{00}}$ is center of gravity in the horizontal direction; $m_v = \frac{m_{01}}{m_{00}}$ is center of gravity in the vertical direction. These central moments are translation invariant since they are normalized with respect to the centroid of the image. The moments can also be normalized with respect to scale using the following expression.

$$\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}^{\gamma}}$$

where $\gamma = \frac{p+q}{2} + 1$.

The set of equations in Equation 3.7 is used to calculate the seven invariant moments that are used to characterize the shape of an object in the image.

$$
\begin{aligned}
\phi_1 &= \eta_{20} + \eta_{02} \\
\phi_2 &= (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2 \\
\phi_3 &= (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2 \\
\phi_4 &= (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2 \\
\phi_5 &= (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 \\
&\quad -3(\eta_{21} + \eta_{03})^2] + (3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03}) \\
&\quad [3(\eta_{30} - 3\eta_{12})^2 - (\eta_{21} + \eta_{03})^2] \\
\phi_6 &= (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] \\
&\quad +4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03}) \\
\phi_7 &= (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 \\
&\quad -3(\eta_{21} + \eta_{03})^2] + (3\eta_{12} - \eta_{30})(\eta_{21} + \eta_{03}) \\
&\quad [3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2]
\end{aligned}
\tag{3.7}
$$

These seven moments are the features that have been used to classify the targets in the images captured by the SFTB. It is to be expected that as the target moves towards or away from the nodes, its orientation and scale will change. Hence, the moments provide the means to characterize them even when these changes are present and the moments are the features best suited for representing the target shape. The moments, target number and node number are written into a database file from which the features are later used for classification.

3.4 Classification

Many classification techniques are based on human approaches to learning. Neural networks and genetic algorithms are examples of methods designed to imitate human learning behavior. Studies of biological learning systems have shown that any predictive-learning system consists of two main phases [18]. The first part is called *induction* which involves learning and estimating dependencies in the

system from the inputs. Induction tries to estimate the input-output behavior of the system from a limited number of observation points (training data) so that this can be modelled and applied to future inputs (testing data). The formulation of this function usually involves the use of *a-priori* information. The second part is *deduction* which uses these estimated dependencies to predict the future output.

There are two different types of inductive learning systems - *supervised* and *unsupervised* [18]. Supervised learning assumes the presence of a teacher or a learning model which will provide the outputs of the system to the training data. This output is compared against that of the learning system, and subsequent changes are made in the system to make these outputs as similar as possible. It can be seen that the system is a closed-loop feedback system. Using a set of training data and constant changes, the system is made to follow the output of the teacher as closely as possible. Once this is accomplished, the training of the learning system is said to be complete. The performance of the system is usually computed in terms of the mean squared error function or the sum of squared error function over the training samples. Now, real-world or testing data can be fed to the system, and the output of the system is expected to be close to the actual output. In case of unsupervised learning systems, there is no feedback or a teacher. The inputs are fed to the learning machine and the system is made to look for regularities in the input data and form internal representations for recognizing similar input samples. Examples of unsupervised learning systems include cluster analysis and some neural networks.

Learning in inductive systems involves the use of density functions and models. These are used to predict the behavior of the inputs and estimate the future outputs. Depending on the availability of *a-priori* knowledge, these methods can be divided into *parametric* and *non-parametric* density estimations [26]. In parametric estimation, the statistics are partially known or assumed. Usually, *a-priori* and class conditional density are known. The common parametric forms rarely fit the densities that are actually encountered in practice. Also, classical parametric densities are unimodal while practical problems are usually multimodal densities. Examples are Maximum-Likelihood estimate (MLE), Minimum Mean Square Error (MMSE) estimate, Maximum a Priori (MAP) estimate and Bayesian

learning inference. In non-parametric techniques, data statistics are unknown. These methods estimate the density function directly from the data. There is no assumption that the density function has a particular functional form and are hence more general methods. However, these require a large amount of data for training. Examples include Parzen window and k Nearest Neighbor (kNN) techniques.

The kNN algorithm is used to classify frames obtained from the SFTB. For the 30 frames that are obtained for each target, 3 frames are used for testing and the remaining 27 are using for training. Thus, for every scenario, out of 210 feature vectors, there are 21 testing vectors and 189 training vectors. And in the database, there are 630 feature vectors with 567 training vectors and 63 testing vectors.

For individual frame classification of a scenario, let us consider $x_1, x_2, \ldots, x_{630}$ to be the feature vectors. The first 63 vectors $x_1, x_2, \ldots, x_{63}$ are used for testing and the remaining are used for testing. In order to obtain the classification result for x_1, a similarity or difference measure is computed between x_1 and all the training vectors. This measure is usually a distance measure like the Euclidean distance. Let these distances be $d_{1,64}, d_{1,65}, \ldots, d_{1,630}$. The k shortest distances (nearest neighbors) are chosen and the target class that appears most often in these k neighbors is assigned to x_1. In the 2D scale, if $d_{1,63}, d_{1,64}, \ldots, d_{1,630}$ were points plotted on the xy-axis, a cell would be centered at x_1 and allowed to be grown until it encompasses k nearest points. The most well-represented class among these neighbors is assigned to x_1. The value of k is usually taken to be $\sqrt{n}$, where n represents the number of training vectors.

The aim of classification is to predict how well the classifier would work when it is given an input it has not seen before. Hence, it is desirable to keep aside a subset of the training data solely for testing without exposing the classifier to it. Thus, this subset of data can be used as 'new data' for testing. This is the idea behind *cross-validation* techniques. The simplest kind of cross-validation is called *holdout method* in which a set of data is kept aside and used only for testing. This method is not preferred since it reduces the amount of training data available and its results depend on which data is used for testing and training. An improvement

over this method is the *K-fold cross-validation*. In this technique, the dataset is divided into k subsets. For every run of the classification routine, one subset is held separately for classification. Each run is repeated with a separate subset for testing, and a new run is carried out until each subset has been for testing once and only once. Each data point gets to be in the testing set once and in the training set $k-1$ times. The classification accuracy is averaged over the k runs. The disadvantage of this method is that it needs k times as much computation and time. In this thesis work, K-fold cross-validation is used to provide a better representation of the classification accuracy.

Chapter 4

Data Fusion

The process of combining decisions from multiple classifiers to obtain one robust final results has been given different names in the technical literature [15, 20] – classifier fusion, mixture of experts, consensus aggregation, classifier ensembles, divide-and-conquer classifiers, combination of multiple experts, etc. Fusion of individual decisions involves the use of mathematical methods like Bayes method, Dempster Shafer techniques and other probabilistic techniques or the use of ad hoc methods like majority voting. In either case, the fusion technique is based on some mathematical foundations and assumptions. For example, probabilistic techniques usually assume that the classifiers use mutually independent subset of features or commit independent classification errors. This chapter presents the mathematics behind fusion and explains some of the widely used fusion techniques. Behavior-Knowledge Space (BKS) fusion algorithm is explained in detail and this technique is used in performing sensor-level fusion in this thesis work.

4.1 Techniques for Fusion

The mathematical notations and expressions that are used here are based on [20]. Let $x \in \Re^n$ be a feature vector. The vector can be obtained from any one of the c classes of targets present. Every mapping that satisfies the condition

$$D : \Re^n \rightarrow [0, 1]^c - \{0\}$$

is called a classifier. In other words, the classifier converts a real number or feature vector into a value between zero and one. The output of the classifier is called *class label* and is denoted as $\mu_D^i(x) = [\mu_D^1(x), \ldots, \mu_D^c(x)]^T, \mu_D^i(x) \in [0, 1]$. Each class label consists of numbers that represent the *a-posteriori* probabilities for each target class. Based on the nature of the class labels, classifiers can be classified under the following types.

1. *Crisp classifiers* are those that assign a specific class label or target class to the output decision variable. Mathematically,

$$\mu_D^i(x) \in \{0, 1\}, \sum_{i=1}^{c} \mu_D^i(x) = 1, \forall x \in \Re^n$$

2. *Fuzzy classifiers* are those that assign a probability value to every class that denotes the chance of that class matching the target. For these classifiers,

$$\mu_D^i(x) \in [0, 1], \sum_{i=1}^{c} \mu_D^i(x) = 1, \forall x \in \Re^n$$

3. *Possibilistic classifiers* can be described as

$$\mu_D^i(x) \in \{0, 1\}, \sum_{i=1}^{c} \mu_D^i(x) > 0, \forall x \in \Re^n$$

In order to convert any decision into a crisp label, which is called *hardening*, the maximum membership rule or majority voting rule can be used. This assigns the output to that class which has the maximum probability value. This rule is given in Equation 4.1.

$$D(x) = k \Leftrightarrow \mu_D^k(x) = max_{i=1,\ldots,c}\{\mu_D^i(x)\} \tag{4.1}$$

For the case of fusing decisions from L classifiers denoted as $\{D_1, D_2, \ldots, D_L\}$ to obtain the fused result $\hat{D}(x)$, the *decision profile* (DP) is formed. Each row of the DP is a class label and hence the DP has L rows. Every ith column of

the DP has the probabilities of the ith class of every classifier. The DP can be represented as

$$DP(x) = \begin{bmatrix} d_{1,1}(x) & \dots & d_{1,j}(x) & \dots & d_{1,C}(x) \\ \dots & \dots & \dots & \dots & \dots \\ d_{i,1}(x) & \dots & d_{i,j}(x) & \dots & d_{i,C}(x) \\ \dots & \dots & \dots & \dots & \dots \\ d_{L,1}(x) & \dots & d_{L,j}(x) & \dots & d_{L,C}(x) \end{bmatrix}$$

Class-conscious and *Class-indifferent* data fusion methods can be defined based on the method by which they calculate the support or belief for each class from the DP. Methods that calculate the support for the ith class using only the ith column of the DP are class-conscious methods. They take into account only the probabilities of that class. Examples of such methods include averaging, minimum, maximum and product operators. Class-indifferent methods use the entire DP to calculate the support or belief for each class. For these methods, a second level of fusion is used to classify the elements of the DP to give the final class label $\hat{D}(x)$. Examples of this method include neural networks, Fisher linear discriminant analysis, Dempster Shafer analysis, etc. The former method uses the context of the DP but uses only a part of the entire information which is available. The latter method uses all the available information but fails to use context-relevant information, which might be a costly loss. A method that does not fall into either extreme category uses *decision templates* [20]. Some methods require the presence of crisp class labels for fusion. Examples of such methods include majority voting, naive Bayes classification and BKS fusion. Fusion methods can also be classified based on the level at which information combination occurs. Data-level, feature-level and decision-level fusion processes are possible.

Majority voting is the process of hardening the class labels to obtain crisp labels. In this method, each classifier is allowed to "vote" for a target based on its probability values. The class that obtains the most number of votes is declared to be the target class. In case more than one class obtain the same number of votes, the tie is broken by randomly choosing a target class from the tied classes. This method does not require any training and can be implemented as such using only the decisions of individual classifiers. Temporal fusion of frames obtained from the

SFTB is performed using majority voting. Individual frame classification results are obtained using the kNN technique. The decisions of all the testing frames of that particular event are taken together and majority voting is performed on these labels. The output of the majority voting gives the temporal fusion result.

Classical inference is one of the basic statistical techniques that can be used for data fusion [14]. To derive the final output, the training data is observed and an empirical probabilistic model is formed that can be used to predict the outputs of future states. Empirical probability calculates the probability of occurrence of an event as the limit of the result of an individual experiment run infinitely many times. Thus, it can be seen that this method cannot be used for single event cases. There are two possible hypotheses – null hypothesis H_0 which states that the observed data is caused by the event E, and the alternate hypothesis H_1 which states that the observed data is not caused by the event E. The decision on how to select the null or alternate hypothesis is based on some decision rules like Maximum a posteriori (MAP) rule, Maximum Likelihood Estimate (MLE), Neyman-Pearson rule or Bayes rule. MAP rule concludes that the hypothesis H_0 is true if the *a-posteriori* probability of H_0 given y is greater than that of H_1 given y. MLE decides on H_0 if $p(y|H_0) > p(y|H_1)$. For the Bayes selection rule, a cost function is established that helps to choose between H_0 and H_1. An example of a cost function is

$$C = C_{00}P(H_0)P_a + C_{01}P(H_0)P_b + C_{10}P(H_1)P_cP(H_1)P_d$$

where $P(H_0)$ and $P(H_1)$ are the *a-priori* probabilities of the hypotheses H_0 and H_1, P_a and P_c are detection probabilities and P_b and P_d are false alarm probabilities. C_{ij} are randomly chosen constants and the Bayes method attempts to find the hypothesis that reduces the cost function C.

Bayes inference is based on the conditional probability density calculation as given in the famous Bayes rule. Consider an event E which can be described using the mutually exclusive and exhaustive hypotheses $H_1, H_2, \ldots, H_j$. The statement of Bayes theorem can be described using Equation 4.2.

$$
\begin{aligned}
P(H_i|E) &= \frac{P(E|H_i)P(H_i)}{\sum_i P(E|H_i)P(H_i)} \\
\sum_i P(H_i) &= 1
\end{aligned} \tag{4.2}
$$

where $P(H_i|E)$ is the *a-posteriori* probability that H_i is true given the event E; $P(H_i)$ is the *a-priori* probability that the hypothesis H_i is true; $P(E|H_i)$ is the probability of the event E occurring when it is known that H_i is true.

Bayes rule gives the probability that a given hypothesis is true when a known event has occurred. Classical inference only gives a probability of relating an observation to an event given as assumed hypothesis. Also, the Bayes inference allows us to incorporate the *a-priori* knowledge that the given hypothesis is true. However, implementing the Bayes rule has some disadvantages. The definition of the prior probabilities is an issue. One of the major issues with the Bayesian inference method is the necessity for independence of the hypotheses and events. Each hypotheses must be mutually exclusive and also exhaustive.

4.2 BKS Fusion

The BKS method was developed by the Concordia OCR Group [15] for the recognition of handwritten characters. This technique avoids the independence assumption by deriving its information from a "knowledge" space which concurrently records the decision of each classifier on each training sample. The behavior of the classifiers is recorded in the knowledge space, and hence the method is called Behavior-Knowledge Space.

The terminology used in this section is defined here. e_k represents each classifier or expert k, where $k = 1, 2, \ldots, K$ and K is the total number of experts available. The total number of classes that are present in the dataset is given by the variable M and the data for each class is given by the mutually exclusive and exhaustive set of patterns $C_1, C_2, \ldots, C_M$. $\Lambda = \{1, 2, \ldots, M\}$ is the set of all possible classes. The input feature vector to the classification system is denoted

as x and the output of the expert k is represented as $e_k = j_k$. This means that the kth classifier has assigned the input vector x to the class j_k, where $j_k \in \Lambda$. The objective of the classification module is to find what combination of the individual classifier results will produce the best final classification decision.

The BKS is a K-dimensional space, with each classifier's decision forming one dimension. Each classifier can produce $M + 1$ distinct outputs, M for each one of the target classes and another output value for rejecting classification. This extra output value can be used to identify new classes. Each unit in the BKS accumulates the number of samples for each class at the intersection of the decision of each classifier. Each unit consists of three different types of information. These include the total number of incoming samples, the best representative class and the total number of incoming samples per class.

The BKS fusion algorithm operates in two stages. These are termed *knowledge modeling* and *decision making*.

4.2.1 Knowledge Modeling

In the first state, a BKS lookup table is generated by exposing the training data to each individual classifier and recording their individual classification results. Let $BKS(e(1), e(2), \ldots, e(K))$ be a unit of the BKS which records the decisions of classifiers one to K. Let the total number of incoming samples for class m in $BKS(e(1), e(2), \ldots, e(K))$ be given as $n_{e(1),\ldots,e(K)}(m)$. The toal number of incoming samples in $BKS(e(1), e(2), \ldots, e(K))$ is given as

$$T_{e(1),\ldots,e(K)} = \sum_{m=1}^{M} n_{e(1),\ldots,e(K)}(m)$$

and the best representative class for $BKS(e(1), e(2), \ldots, e(K))$ is given as

$$R_{e(1),\ldots,e(K)} = \{j | n_{e(1),\ldots,e(K)}(j) = max_{1 \leq m \leq M} n_{e(1),\ldots,e(K)}\}$$

Using these values, the lookup table of the BKS is computed. The lookup table has three columns. The first column serves as an index and is represented by the set of temporal fusion results. Each row of the lookup table contains these

Table 4.1: Example of BKS lookup table

Class combination	Class concentration	Cell label
1,1	6/1/2	1
1,2	4/9/6	2
2,1	8/8/6	1,2
2,2	0/0/0	random

entries - number of times that index has been encountered for every class and a single class label which is the most encountered class among all training samples. An example of a possible BKS lookup table for a two-class, two-classifier system with 50 training samples is shown in Table 4.1.

To generate the lookup table for BKS, the classification processes for frame-level and temporal fusion are performed. An event is chosen, and testing and training sets are laid out for this event. Frame-level classification results are combined to form the temporal fusion results. The event level fusion results are obtained for both nodes – the infrared and color camera nodes. The set of temporal fusion result from each node is used to index the lookup table. Thus, each node serves as a separate classifier in our case.

4.2.2 Decision Making

Once the lookup table has been populated, the system is now ready to perform BKS fusion on the testing dataset. For any set of input frames for both nodes, frame-level and event-level results are obtained as before. The temporal fusion results of both nodes are used to find the row in the BKS lookup table that is of interest. The last column of that row contains the classification result of the sensor-level fusion. The final decision rule can be stated mathematically as

$$E(x) = \left\{ \begin{array}{ll} R_{e(1),\ldots,e(K)}, & when\ T_{e(1),\ldots,e(K)} > 0\ and\ \frac{n_{e(1),\ldots,e(K)}(R_{e(1),\ldots,e(K)})}{T_{e(1),\ldots,e(K)}} \geq \lambda \\ M+1, & otherwise \end{array} \right\}$$

where λ is a threshold such that $0 \leq \lambda \leq 1$. This factor controls the reliability of the final decision and is usually set heuristically. It can be seen that the decision modeling stage of the BKS needs to be performed only once over a set of training samples. For every set of testing vectors that are based on the same training database, the decision making stage has to be run repeatedly to obtain their fusion results.

4.2.3 Discussion

The BKS fusion algorithm performs better than other techniques like majority voting, classical inference and naive Bayes method. Also, it is not bound by the independence assumption unlike most probabilistic methods and it has some good properties like automatic thresholding and optimality [15]. However, the BKS fusion algorithm has its limitations. It needs a huge amount of training data so that the lookup table is well populated and is representative of the actual scenario. If only a few samples are considered for computing the lookup table, and failing careful selection of samples, the desirable qualities of the BKS method cannot be guaranteed. With the issue of a large training set comes the problem of overfitting. Since a large database is required for proper calculation of the lookup table, the BKS algorithm is prone to overfitting. Hence, it may perform well on the training data while not meeting expectation when run on the testing data. Four fusion methods were tested in [15]. These were majority voting, Bayesian classification, Dempster Shafer fusion and BKS fusion. It was observed that BKS outperforms the other techniques except in cases of low substitution rates.

Chapter 5

Results and Discussions

The previous chapters have explained the premises and implementation of this research work. Data acquisition from the SFTB, image processing and feature extraction, formation of the database of feature vectors and the classification processes involved have been discussed. In this chapter, the different experiments that have been carried out are put forth and their results are presented and explained. The classification routines were coded in MATLAB while the image processing algorithms and database generation were implemented using GNU's C++ compiler. The use of different techniques and algorithms are justified using these results.

5.1 Classification Levels and Databases

In the classification of the civilian target images obtained using the SFTB, three different levels of classification and two different databases have been used. In addition to this, frames were chosen in different ways – successively and intermittently. It was seen that for a realistic classification scenario, the frames must not be successive since nearby frames have high correlation. Hence using one frame for testing and its neighboring frames for training would always provide a high classification accuracy, although this would not be reflective of the practical case. Therefore intermitted frames were used to generate the database entries.

The three different levels of classification that were performed are frame-level

classification, temporal fusion classification and BKS fusion classification. Temporal fusion and BKS fusion can be mapped to the event level and sensor level fusion. It is expected that as the level of decision making goes higher, the classification accuracy and the reliability will increase. The results obtained prove this trend. The two different databases that have been used for classification differ in their target classes and the number of samples per target that they contain. In the first database, the seven targets were treated as individual classes and classification results were obtained (7 class case). In this case, as discussed earlier, each target has 30 frames per event. In the second database, cars were combined to form class 1, SUVs were combined to form class 2 and light trucks were combined to form class 3 (3 class case). Target 7 (Stake body truck) was not included in the second database. So, there are three classes present in total and for each of the three classes present, 60 frames are available per event. This increases the number of training samples present and classification is expected to be easier since similar classes have been grouped together. This grouping helps in differentiating between similar looking cars or SUVs since only the vehicle type has to be differentiated. As expected, the results were better for the second database. The first database performs target recognition while the second performs target identification.

For each level of classification, a different classification technique is used. At the frame level, kNN is used to classify the testing data and the process is repeated to achieve cross-validation. Temporal fusion results are obtained by fusing frame-level results using the majority voting technique. BKS fusion results are obtained using temporal fusion results and the BKS lookup table. The total number of feature vectors, training vectors and testing vectors in the two different databases are listed below.

First database – 7 class (Target Recognition)

Frame-level

Number of samples per target per event	= 30
Number of training samples per target per event	= 3
Number of testing samples per target per event	= 27

Event-level

Number of samples per target per scenario = 210
Number of training samples per target per scenario = 189
Number of testing samples per target per scenario = 21

Sensor-level

Number of samples per sensor = 630
Number of training samples per sensor = 567
Number of testing samples per sensor = 63

Second database – 3 class (Target Identification)

Frame-level

Number of samples per target per event = 60
Number of training samples per target per event = 5
Number of testing samples per target per event = 55

Event-level

Number of samples per target per scenario = 180
Number of training samples per target per scenario = 15
Number of testing samples per target per scenario = 165

Sensor-level

Number of samples per sensor = 540
Number of training samples per sensor = 45
Number of testing samples per sensor = 495

At the individual frame level, the features of the input testing frame are compared against those of all the training frames. The distances are calculated and the 'k' nearest neighbors are studied to identify the target class of the testing frame. At the event level, all the testing frames of the same target captured by the same node are used to form a single decision. The individual frame decisions are combined using majority voting method. At the sensor level, the corresponding temporal fusion results of both the nodes are used to form one final result. The different classification levels are illustrated with respect to the first database

(7 class) in Figure. 5.1. The process holds good for the second database (3 class) with a change in the number of training and testing vectors.

5.2 Training and Testing

In general, the classification program can be divided into two parts based on the implementation of the BKS algorithm. The first part of the program deals with the *knowledge modeling* stage of BKS. In this part, the dataset is used to construct the BKS lookup table using the classification results and the ground truth. Once the lookup table is obtained, the second part of the program is executed. This part is called the *decision making* stage. In this stage, random testing vectors are given as input to the classification algorithm and the fused results are obtained using the lookup table. These stages of the BKS algorithm have been discussed earlier.

For the knowledge modeling stage, the infrared and color video datasets are split into testing and training vectors as mentioned in Section 5.1. K-fold cross-validation is performed on the database. The number of runs of the classification algorithm needed for cross-validation is equal to the ratio of total number of samples in the database to the number of testing samples in one run. Thus, the number of runs for the 7 class and 3 class databases are 10 and 12 respectively. This ensures that all frames are used for obtaining the lookup table entries. Once the lookup table is populated using the above method, the program is ready to classify any given set of test vectors. The testing vectors for each run is not chosen randomly. If five vectors are to be chosen for testing from a total of 60 vectors, five equally spaced vectors are taken from the database, with the index of the first vector equalling the run number. Therefore, for the first run, vectors 1, 13, 25, 37, and 49 are chosen for testing while for the second run, vectors 2, 14, 26, 38 and 50 are chosen for testing. The last run would consist of vectors 12, 24, 36, 48 and 60. A similar process is followed to chose 3 training vectors per run from a total of 30 vectors.

To model the decision making stage, a set of testing vectors is chosen randomly from the database. The number of vectors chosen depends on the database used

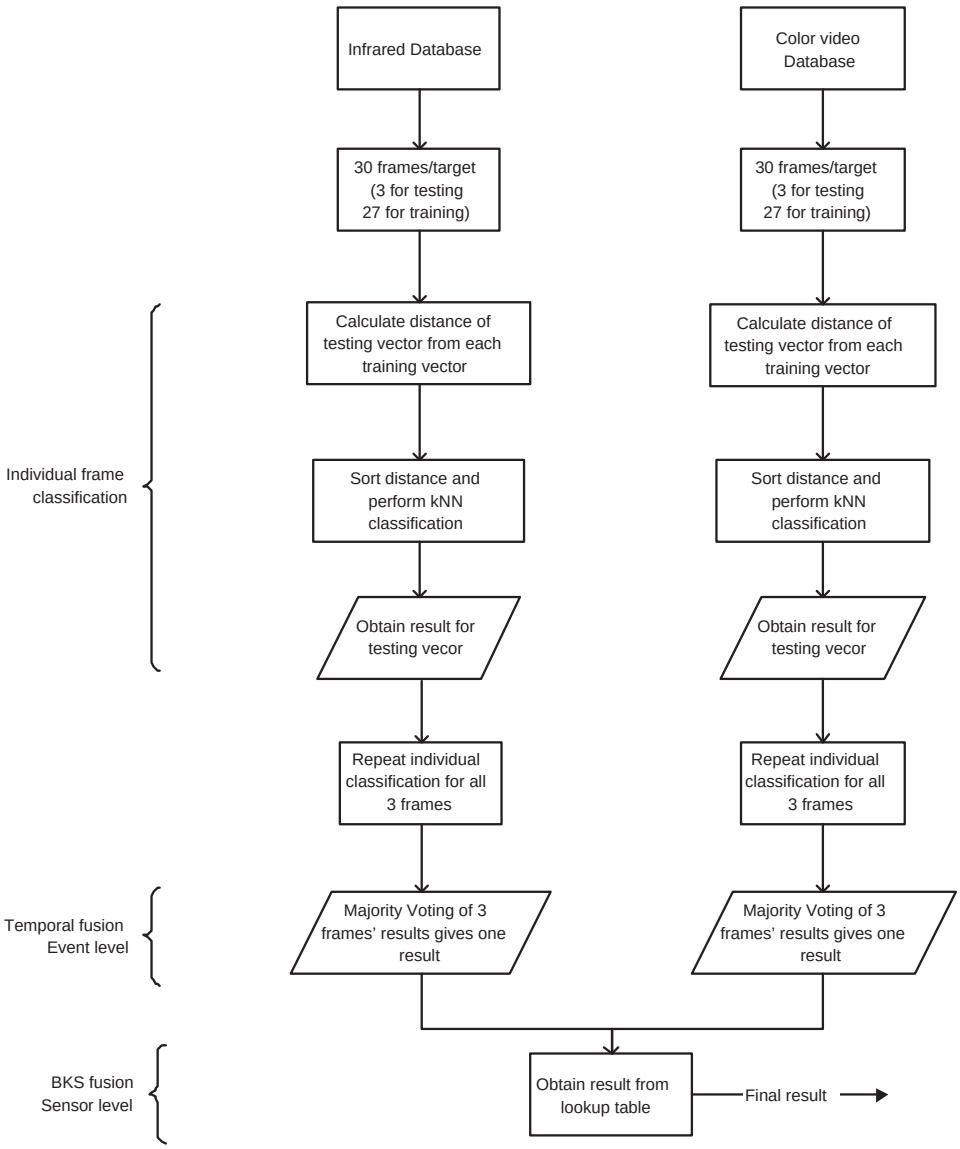


Figure 5.1: Classification levels with respect to the first database (7 class)

since this will match the number of testing vectors that were used for each run during the formation of the lookup table. The difference in the testing vector set used for lookup table formation and testing vector set used for classification is that the latter is not carefully chosen but are random. Hence, the probability of using the same set of vectors for lookup table formation and final classification is less. It is important to note that the number of training and testing samples per node or per event does not apply in the decision making stage. Out of the entire database, one event (scenario and target) is chosen randomly. A set of testing vectors is chosen randomly for this event from both nodes. The three levels of classification accuracies that are obtained as described here.

Individual frame classification These testing vectors are classified and their individual classification accuracy is obtained by comparing the classification result against the ground truth.

Temporal fusion For every event, all the testing vectors are fused temporally using majority voting. This result is compared against the ground truth data to obtain the classification result at this level of fusion.

BKS fusion The temporal fusion results from both the nodes are taken and the index vector for the lookup table is formed. The last column for this index vector in the lookup table contains the classification result. This result is compared against the ground truth to obtain the classification accuracy for the sensor fusion stage.

For the first database (7 class), this process gives six classification results at the frame level (three per node), two results at the event level (one per node) and one classification result at the sensor level. Within the program, the decision making loop is run 50 times to get a better idea of the classification accuracy, i.e. the classification accuracy is averaged over 50 sets of testing vectors. The program itself was executed 20 times so as to extract a mean and standard deviation for the classification accuracy.

5.3 Classification Results

The results of the three levels of classification are presented in this section. As mentioned earlier, the classification algorithm was run twenty times to obtain twenty sets of accuracies at the frame, event and sensor levels. Examples of the confusion matrices obtained for the 3 classification levels of the 7 class and 3 class databases are shown in Tables 5.1 to 5.6. The confusion matrices for the 7 class and 3 class case are shown graphically for $k = 11$ in Figures 5.2 and 5.3 and class-wise results are given in Figures 5.4 and 5.5. The mean and standard deviation of these twenty classification values were calculated. These values have been tabulated in Tables 5.7 and 5.8 and shown graphically in Figures 5.6 and 5.7. Each row entry of these tables corresponds to the results obtained for each of the twenty runs of the classification routine. From these tables, it can be inferred that the second database has better classification accuracy than the first. This is because it is easier to differentiate cars, SUVs and light trucks from each other than differentiate different types of cars, SUVs and light trucks. Also, it can be seen that the classification accuracy at the event level is greater than that at the frame level and the classification accuracy at the sensor level is greater than that at the event level i.e. temporal fusion performs better than frame based classification and BKS fusion performs better than temporal fusion. Hence, it can be substantiated that fusion improves target classification and the higher the level of fusion, the better the accuracy.

In addition to classification accuracy, the time taken for feature extraction and classification was also recorded. The time taken for feature extraction is calculated by averaging thirty runs of the algorithm on different input frames. It has been explained that the classification algorithm runs fifty times and averages the results obtained during all those runs. The time taken to obtain the BKS fusion result for each of these fifty runs was recorded and averaged. These timings are given below.

Feature extraction: This was performed on a Pentium III machine running on Red Hat Linux using the GNU C++ compiler.

1. Time taken to extract features and write into database file after segmenta-

Table 5.1: Confusion matrix - 7 class; individual frame classification; k=11; Classification accuracy=67.67%

	T1	T2	T3	T4	T5	T6	T7
T1	48	4	12	2	3	1	5
T2	1	44	0	1	0	2	0
T3	4	0	45	1	1	5	3
T4	0	5	0	34	0	1	2
T5	6	2	3	2	12	7	5
T6	0	5	6	1	2	13	2
T7	1	0	0	1	0	1	7

Table 5.2: Confusion matrix - 7 class; temporal fusion classification; k=11; Classification accuracy=74.00%

	T1	T2	T3	T4	T5	T6	T7
T1	19	2	5	0	1	1	3
T2	0	17	0	1	0	2	0
T3	0	0	16	0	0	1	1
T4	0	1	0	13	0	1	1
T5	1	0	0	0	4	2	1
T6	0	0	1	0	1	3	0
T7	0	0	0	0	0	0	2

Table 5.3: Confusion matrix - 7 class; BKS fusion classification; k=11; Classification accuracy=86.00%

	T1	T2	T3	T4	T5	T6	T7
T1	9	0	1	0	0	1	0
T2	0	9	0	0	0	1	0
T3	0	0	9	0	0	0	0
T4	0	0	0	7	0	0	0
T5	1	0	0	0	3	0	1
T6	0	0	1	0	0	3	0
T7	0	1	0	0	0	0	3

Table 5.4: Confusion matrix - 3 class; individual frame classification; k=11; Classification accuracy=75.00%

	Car	SUV	Light truck
Car	165	19	14
SUV	17	129	15
Light truck	28	32	81

Table 5.5: Confusion matrix - 3 class; temporal fusion classification; k=11; Classification accuracy=88.00%

	Car	SUV	Light truck
Car	38	2	1
SUV	2	31	2
Light truck	2	3	19

Table 5.6: Confusion matrix - 3 class; BKS fusion classification; k=11; Classification accuracy=96.00%

	Car	SUV	Light truck
Car	21	1	1
SUV	0	17	0
Light truck	0	0	10

Table 5.7: Classification accuracy (%) for 7 class case with mean and standard deviation of accuracy, k=11

Frame Level *Individual frame*	Event Level *Temporal fusion*	Sensor Level *BKS fusion*
73.00	79.00	82.00
61.33	64.00	66.00
69.00	79.00	84.00
66.33	72.00	82.00
67.67	74.00	84.00
64.67	72.00	84.00
60.67	66.00	82.00
59.00	65.00	70.00
65.67	72.00	86.00
67.33	73.00	80.00
63.67	67.00	74.00
63.67	65.00	76.00
62.67	67.00	74.00
62.33	66.00	76.00
66.67	67.00	68.00
65.33	73.00	84.00
65.67	74.00	78.00
59.67	64.00	72.00
64.33	68.00	78.00
72.67	75.00	74.00
MEAN		
65.0675	70.1000	77.7000
STANDARD DEVIATION		
3.7707	4.7782	5.8858

Table 5.8: Classification accuracy (%) for 3 class case with mean and standard deviation of accuracy, k=11

Frame Level *Individual frame*	Event Level *Temporal fusion*	Sensor Level *BKS fusion*
75.00	90.00	94.00
73.60	87.00	88.00
77.00	91.00	98.00
74.00	93.00	94.00
74.20	89.00	94.00
76.40	90.00	94.00
77.80	95.00	96.00
74.40	89.00	90.00
74.80	88.00	90.00
75.00	92.00	96.00
76.60	90.00	90.00
77.00	90.00	96.00
74.80	89.00	98.00
73.80	92.00	94.00
75.20	91.00	100.00
76.20	88.00	96.00
73.80	86.00	96.00
75.00	87.00	98.00
78.80	92.00	98.00
76.40	91.00	96.00
MEAN		
75.49	90.00	94.80
STANDARD DEVIATION		
1.4574	2.2243	3.2053

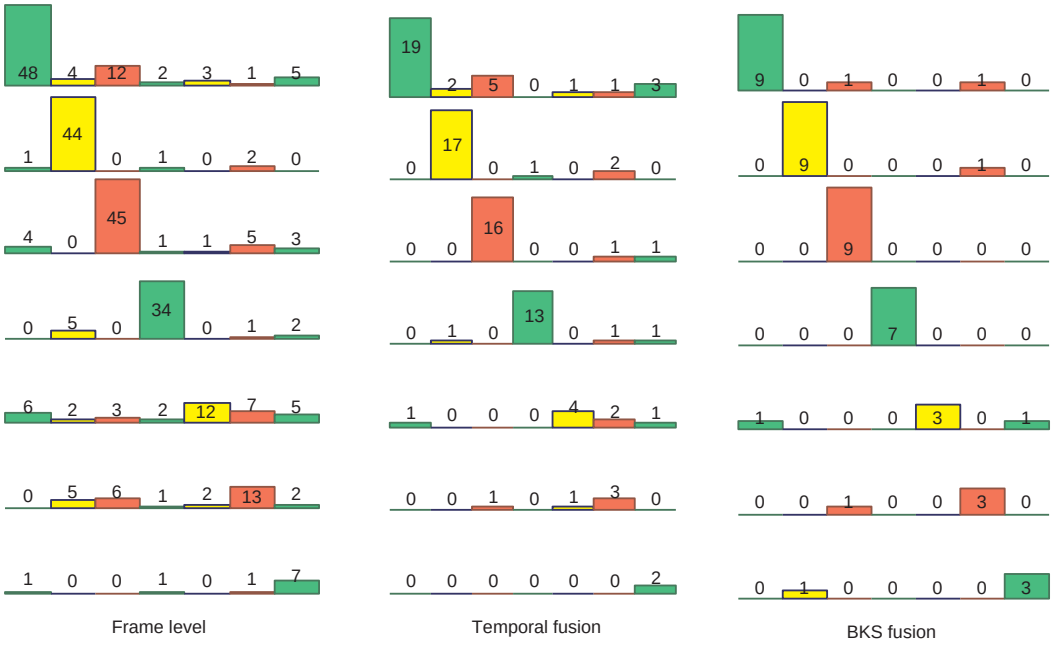


Figure 5.2: Confusion matrix for 7 class case, k=11

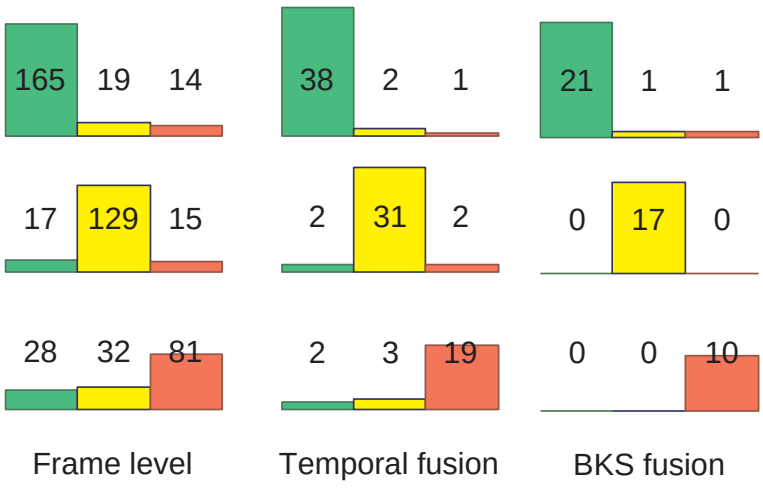


Figure 5.3: Confusion matrix for 3 class case, k=11

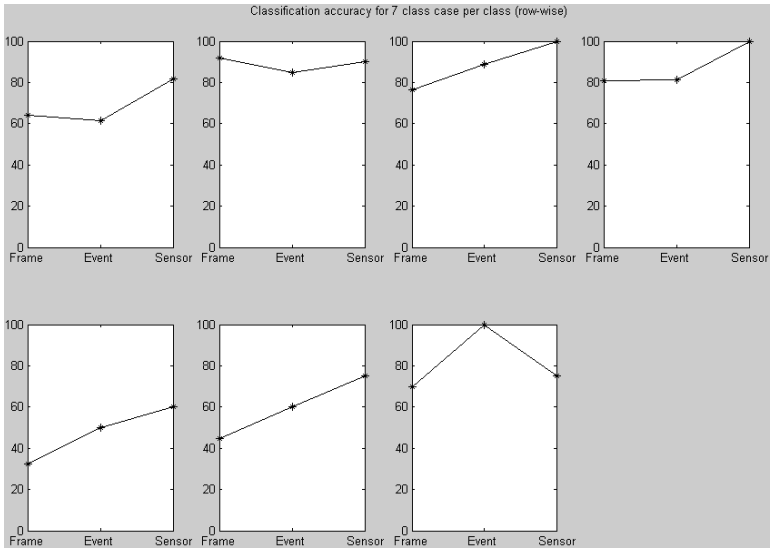


Figure 5.4: Class-wise classification accuracy for 7 class case, k=11

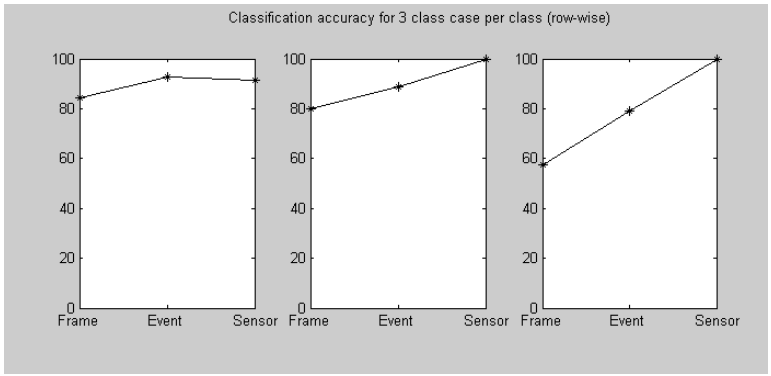


Figure 5.5: Class-wise classification accuracy for 3 class case, k=11

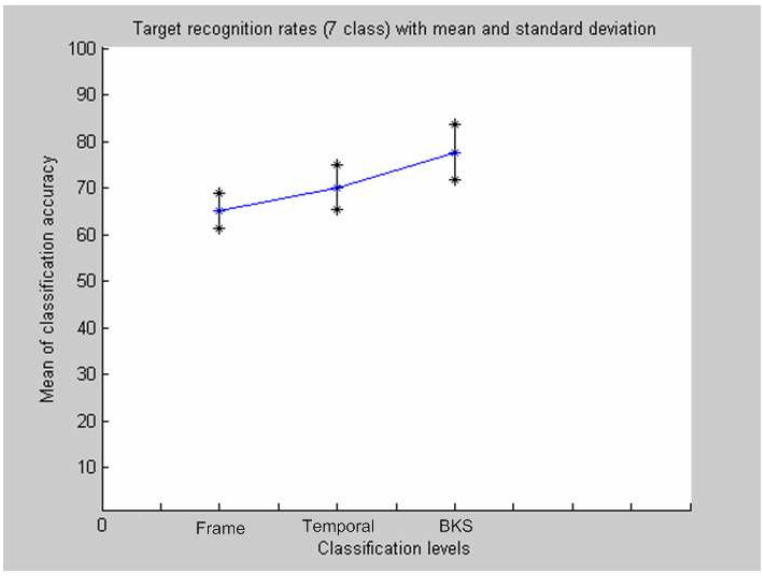


Figure 5.6: Target recognition mean and standard deviation for different classification levels, k=11

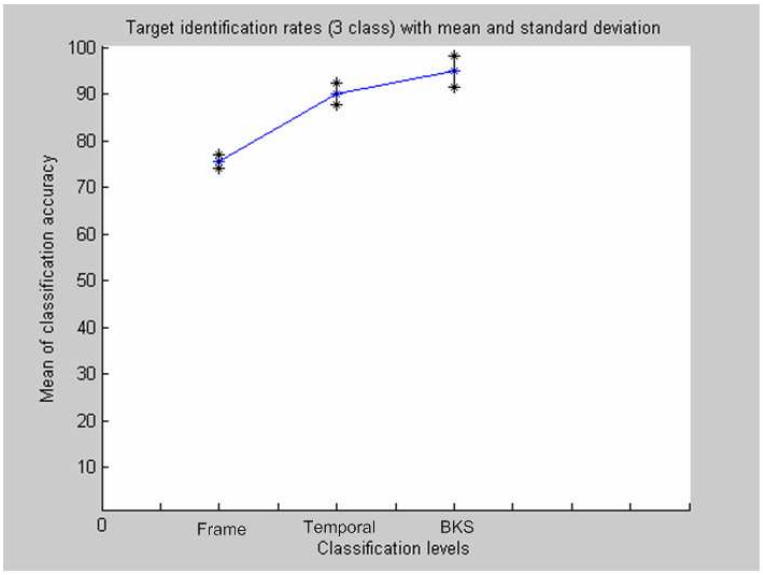


Figure 5.7: Target identification mean and standard deviation for different classification levels, k=11

tion, averaged over 30 runs is .253333 seconds

2. Time taken to normalize a feature vector database containing 60 feature vectors is 0.03 seconds

Classification: This was performed on a Pentium M 1.6GHz machine with 512Mb RAM running on Microsoft Windows XP. The program was written and executed using MATLAB.

1. Time taken to obtain BKS fusion (sensor-level) result averaged over 50 runs for the 7 class case with k=11 is 0.0439 seconds

2. Time taken to obtain BKS fusion (sensor-level) result averaged over 50 runs for the 3 class case with k=11 is 0.0993 seconds

Image preprocessing, segmentation and feature extraction is done in about 250 milliseconds. Normalizing the database is needed only for training and not required when the system has been deployed. The time taken for classification is well below 50 milliseconds for the 7 class case, and below 100 milliseconds for the 3 class case. The second database exhibits higher classification time due to the presence of increased training vectors compared to the first database. In either case, the time taken to classify the feature vector is significantly less and is well-suited for real-time operation of the fusion algorithm. Hence, this thesis work demonstrates that fusion techniques for the civilian ATR problem provides high classification accuracy at very high processing speeds suitable for real-time operation.

5.4 Distance-based Classification

In order to study the effect of distance on classification accuracy, the classification of civilian targets was performed based on the distance of the target from the sensor nodes. Previously, once the classifier was trained and the lookup table for the BKS algorithm was formed, testing was performed using a random set of vectors from the database. For distance-based classification, the training and

lookup table formation is the same. For testing, random images are not taken. Instead, three groups of images are used for every event – images when target is close to the camera, moderately far and very far.

For the 7-class classification, each image group consists of 3 images (temporal fusion is performed on 3 images per event) and each group of images is separated by around 12 frames (approximately 2.5 meters).

For the 3-class classification, each image group consists of 5 images (temporal fusion is performed on 5 images per event) and each group of images is separated by around 8 frames (approximately 3 meters).

Two different values of k (number of nearest neighbors) were used – k=9 and k=11. It is expected that when the target is at a moderate distance from the cameras, the classification accuracy would be highest. Also, the increase in accuracy from individual frame-based classification to temporal and BKS fusion is expected. These trends are generally observed in the results. Very near and far targets have lesser classification accuracy compared to near targets. However, it can also be seen that the accuracy is much lesser than the one obtained using random vectors for classification (which was discussed in the previous report). When random vectors were used for testing, the classification accuracies ranged from 70% (for individual frame classification) to 95% (for BKS fusion). There is one possible reason for this. When the testing vectors are chosen randomly, they are usually chosen well-spaced apart. Hence, the remaining vectors in the database provide a good means for distance-calculation and comparison. However, when a group of vectors from one part of the database is chosen for testing, as in this case, there are no nearby vectors left in the database for comparison. Therefore, good classification accuracy is hard to obtain. As the size of the group increases, the number of nearby frames for comparison and accuracy will fall. For example, if frames 3, 4 and 5 are used for testing (size of group is 3), then the frames left behind for comparison which give a good representation of the testing frames are frames 1, 2, 6 and 7. If frames 2, 3, 4, 5 and 6 are used for testing (size of group is 5), then the frames left behind for proper comparison are reduced to only frames 1 and 7. Tables 5.9 to 5.12 show the classification accuracies obtained. Figures 5.8 and 5.9 show the graphical version of Tables 5.10 and 5.12.

Table 5.9: 7 class classification accuracy(%), k=9

Distance	Frame	Temporal Fusion	BKS Fusion
Very near	53.97	57.14	57.14
Near	59.52	59.52	61.90
Far	35.71	35.71	42.86

Table 5.10: 7 class classification accuracy(%), k=11

Distance	Frame	Temporal Fusion	BKS Fusion
Very near	50.79	52.38	52.38
Near	60.32	59.52	61.90
Far	35.71	33.33	57.14

Table 5.11: 3 class classification accuracy(%), k=9

Distance	Frame	Temporal Fusion	BKS Fusion
Very near	61.11	66.67	66.67
Near	60.00	61.11	66.67
Far	51.11	44.44	44.44

Table 5.12: 3 class classification accuracy(%), k=11

Distance	Frame	Temporal Fusion	BKS Fusion
Very near	60.00	66.67	66.67
Near	61.11	61.11	77.78
Far	50.00	44.44	66.67

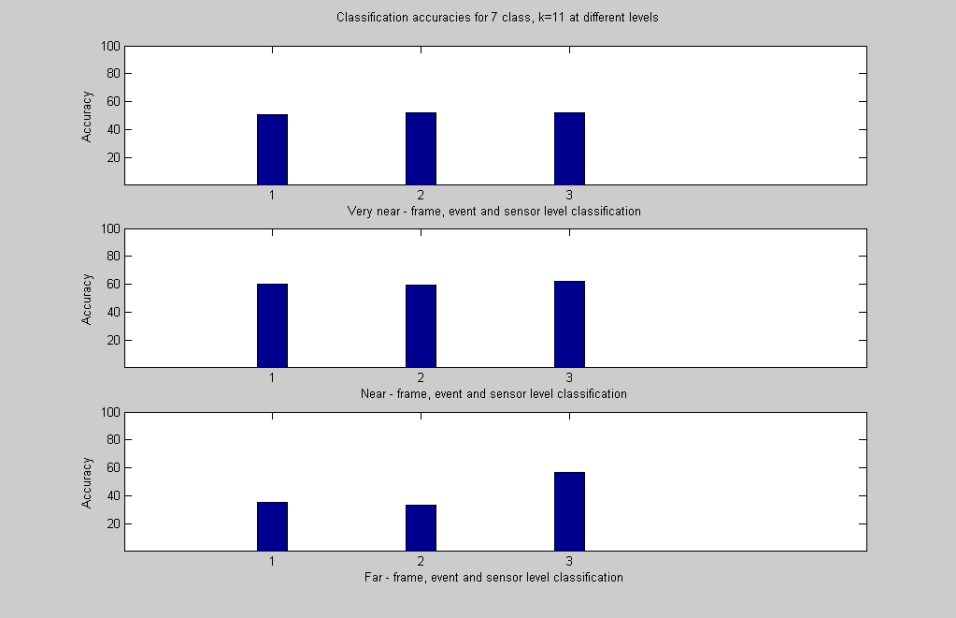


Figure 5.8: Classification accuracy for 7 class case, k=11

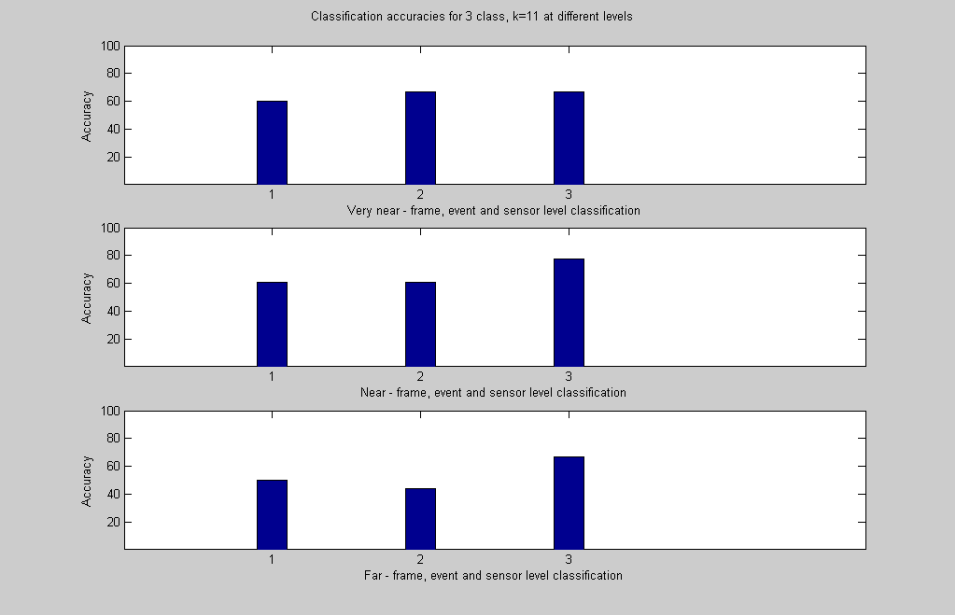


Figure 5.9: Classification accuracy for 3 class case, k=11

In addition to the classification accuracy, it is often convenient and handy to study misclassification also. This can be done using the confusion matrix. The rows represent the ground truth and the columns indicate classification results obtained. VN, N and F stand for 'Very Near', 'Near' and 'Far' cases.

Confusion matrices of the above classifications for the seven class case and a 'k' value of 9 are displayed in Tables 5.13 to 5.21. {T1,T2...,T7} indicate the seven targets – Honda CRX, Chevy Cavalier, Toyota Pickup, GMC Pickup, Xterra, Toyota Forerunner and Stake body light truck.

Confusion matrices of the above classifications for the three class case and a 'k' value of 9 are displayed in Tables 5.22 to 5.24. T1, T2 and T3 indicate cars, SUVs and light truck targets respectively. For example, in the first confusion matrix for the 'Very Near' targets, cars have been classified as cars 17 times, SUVs 8 times and light trucks 5 times. Thus the classification accuracy would be 17/(17+8+5) = 56.67%.

Table 5.13: Confusion matrix - 7 class; individual frame classification; distance very near; k=9

	T1	T2	T3	T4	T5	T6	T7
T1	9	0	7	0	3	0	3
T2	1	14	0	0	0	1	0
T3	3	0	9	1	0	1	1
T4	1	3	0	14	2	0	0
T5	2	0	1	2	10	6	4
T6	1	1	0	1	2	5	6
T7	1	0	1	0	1	5	4

Table 5.14: Confusion matrix - 7 class; individual frame classification; distance near; k=9

	T1	T2	T3	T4	T5	T6	T7
T1	12	0	4	0	1	0	6
T2	0	15	0	1	0	0	0
T3	1	0	12	1	5	2	0
T4	0	0	0	14	0	0	3
T5	5	2	1	2	5	4	4
T6	0	1	1	0	4	12	0
T7	0	0	0	0	3	0	5

Table 5.15: Confusion matrix - 7 class; individual frame classification; distance far; k=9

	T1	T2	T3	T4	T5	T6	T7
T1	11	1	0	5	5	0	0
T2	0	11	0	0	0	1	1
T3	4	0	15	0	4	2	5
T4	0	4	0	9	3	3	2
T5	2	1	2	1	4	3	0
T6	0	1	1	1	2	6	3
T7	1	0	0	2	0	3	7

Table 5.16: Confusion matrix - 7 class; temporal fusion classification; distance very near; k=9

	T1	T2	T3	T4	T5	T6	T7
T1	4	0	3	0	1	0	1
T2	1	5	0	0	0	0	0
T3	1	0	3	0	0	0	0
T4	0	1	0	6	1	0	0
T5	0	0	0	0	3	2	2
T6	0	0	0	0	1	2	2
T7	0	0	0	0	0	2	1

Table 5.17: Confusion matrix - 7 class; temporal fusion classification; distance near; k=9

	T1	T2	T3	T4	T5	T6	T7
T1	4	0	2	0	0	0	2
T2	0	5	0	1	0	0	0
T3	0	0	4	0	2	1	0
T4	0	0	0	5	0	0	1
T5	2	1	0	0	1	1	1
T6	0	0	0	0	2	4	0
T7	0	0	0	0	1	0	2

Table 5.18: Confusion matrix - 7 class; temporal fusion classification; distance far; k=9

	T1	T2	T3	T4	T5	T6	T7
T1	4	0	0	3	2	0	0
T2	0	4	0	0	0	0	0
T3	1	0	6	0	1	1	2
T4	0	2	0	3	1	1	1
T5	1	0	0	0	2	1	0
T6	0	0	0	0	0	2	1
T7	0	0	0	0	0	1	2

Table 5.19: Confusion matrix - 7 class; BKS fusion classification; distance very near; k=9

	T1	T2	T3	T4	T5	T6	T7
T1	1	0	1	0	1	0	1
T2	1	2	0	0	0	0	0
T3	0	0	2	0	0	0	0
T4	0	0	0	3	0	0	0
T5	0	1	0	0	1	1	1
T6	0	0	0	0	0	1	0
T7	1	0	0	0	1	1	1

Table 5.20: Confusion matrix - 7 class; BKS fusion classification; distance near; k=9

	T1	T2	T3	T4	T5	T6	T7
T1	1	0	0	0	1	0	0
T2	0	2	0	0	0	0	0
T3	0	0	3	0	1	0	0
T4	0	0	0	3	0	0	0
T5	2	0	0	0	0	0	2
T6	0	1	0	0	1	3	0
T7	0	0	0	0	0	0	1

Table 5.21: Confusion matrix - 7 class; BKS fusion classification; distance far; k=9

	T1	T2	T3	T4	T5	T6	T7
T1	1	0	0	0	0	0	1
T2	0	1	0	0	0	0	0
T3	1	0	3	0	0	0	0
T4	0	0	0	3	1	0	0
T5	1	2	0	0	2	0	0
T6	0	0	0	0	0	2	1
T7	0	0	0	0	0	1	1

Table 5.22: Confusion matrix - 3 class; individual frame classification; distances very near, near and far; k=9

VN	T1	T2	T3	**N**	T1	T2	T3	**F**	T1	T2	T3
T1	17	8	5	T1	16	6	0	T1	15	2	7
T2	5	15	3	T2	2	17	8	T2	5	17	10
T3	8	7	22	T3	12	7	22	T3	10	11	13

Table 5.23: Confusion matrix - 3 class; temporal fusion classification; distances very near, near and far; k=9

VN	T1	T2	T3	**N**	T1	T2	T3	**F**	T1	T2	T3
T1	4	2	1	T1	3	1	0	T1	3	1	1
T2	1	3	0	T2	1	4	2	T2	1	3	3
T3	1	1	5	T3	2	1	4	T3	2	2	2

Table 5.24: Confusion matrix - 3 class; BKS fusion classification; distances very near, near and far; k=9

VN	T1	T2	T3	**N**	T1	T2	T3	**F**	T1	T2	T3
T1	2	1	1	T1	2	0	0	T1	2	0	1
T2	1	2	0	T2	1	3	1	T2	1	3	1
T3	0	0	2	T3	0	0	2	T3	0	0	1

Chapter 6

Conclusions

ATR is an important application of computer vision. It uses techniques from a variety of fields like image processing, pattern recognition, statistical methods and data fusion techniques and is based on different hardware technologies. This field has been researched for well over two decades now, and though there is no definite answer to the problem of clutter and effective target recognition, the results are improving with technology. One of the recent advances in the ATR field has been to use multiple sensors to look at the target data and fuse the data or results from these multiple sensors. This has been shown to improve the detection accuracies and reduce the system cost by using inexpensive hardware.

This thesis has researched the ATR process as applied to civilian targets. Civilian targets are generally considered harder to classify than their military counterparts due to the nature of their power spectral density. In order to underline the growing importance of fusion in ATR applications and reduce computation time and system cost, this thesis work has mainly used basic and simple image processing techniques for target segmentation and classification. The high overall classification accuracy is obtained by performing decision-level fusion at two stages. First, the individual frame classification results are fused over time using the simple majority voting technique. This simple fusion technique shows a marked increase, on an average, in the target identification ability of the system. To further boost the accuracy, sensor level fusion is performed using the BKS algorithm. The increase in performance is clearly demonstrated in the results

presented. From the classification rates that have been tabulated earlier, it can be seen that temporal fusion results are better than individual frame classification, and sensor fusion results are better than the temporal fusion results. Thus, the superiority of data fusion techniques and the increase in performance with the level of fusion have been successfully demonstrated. In addition to the improved classification rates, introduction of fusion into the recognition system also reduces the cost of the ATR system by removing the need for complex and costly sensors, and highly computational and iterative algorithms. The time taken by the system for classification is recorded. It is observed that the classification time is small enough implement the algorithms in real time. Distance-based classification was also performed to study the effect of distance on the classification ability of the system.

As course of future work in this area, advanced image processing algorithms can be added to the framework that has been provided. This should improve the system performance rates to even higher levels. Also, different fusion techniques can be tried and tested. The best combination of algorithms can be coded together and stored in the SFTB so that the testbed can act as an effective target recognition network. The current classification system can be integrated in the SFTB and field-tested for real-time requirements. This would create a customizable and effective ATR system that can provide immediate and accurate classification.

Bibliography

Bibliography

[1] Kenneth Augustyn. A New Approach to Automatic Target Recognition. *IEEE Transactions on Aerospace and Electronic Systems*, 28(1), January 1992.

[2] Bir Bhanu and Terry L. Jones. Image Understanding Research For Automatic Target Recognition. *IEEE AES Systems Magazine*, October 1993.

[3] Richard R. Brooks and S.S. Iyengar. *Multi-Sensor Fusion: Fundamentals and Applications with Software*. Prentice Hall, 1998.

[4] W. M. Brown and C. W. Swonger. A Prospectus for Automatic Target Recognition. *IEEE Transactions on Aerospace and Electronic Systems*, 25(3), May 1989.

[5] R. Cucchiara and M. Piccardi. Vehicle Detection under Day and Night Illumination. In *Proceedings of ISCS-IIA99, Special Session on Vehicle Surveillance*.

[6] Night Vision & Electronics Sensors Directorate. Sensor fusion testbed description.

[7] Night Vision & Electronics Sensors Directorate. Sensor fusion testbed local site data collection test plan.

[8] Guest Editorial. Introduction to the Special Issue in Automatic Target Detection and Recognition. *IEEE Transactions on Image Processing*, 6(1), January 1997.

[9] Jorge Eduardo Pérez-Jácome F. *Automatic Target Recognition Systems with Emphasis in Model-Based Approaches*. PhD thesis, Georgia Institute of Technology, November 1996.

[10] Jan Flusser. On the Independence of Rotation Invariant Moments. *Pattern Recognition*, pages 1405–1410, 2000.

[11] Rafael C. Gonzalez and Richard E. Woods. *Digital Image Processing*. Pearson Education, 2003.

[12] Stephen Grossberg, Harold Hawkins, and Allen Waxman. Introduction: 1995 Special Issue Automatic Target Recognition. *Neural Networks, Elsevier Science*, 8(7/8), 1995.

[13] David L. Hall and James Llinas. *Handbook of Multisensor Data Fusion*. CRC Press, 2001.

[14] David L. Hall and Sonya A.H. McMullen. *Mathematical Techniques in Multisensor Data Fusion*. Artech House, second edition, 2004.

[15] Y.S. Huang and C.Y. Suen. A Method of Combining Multiple Experts for the Recognition of Unconstrained Handwritten Numerals. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(1), January 1995.

[16] Anil K. Jain. *Fundamentals of Digital Image Processing*. Prentice Hall.

[17] Rudolf G. Buser James A. Ratches, C. P. Walters and B. D. Guenther. Aided and Automatic Target Recognition Based Upon Sensory Inputs from Image Forming Systems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(9), September 1997.

[18] Mehmed Kantardzic. *Data Mining: Concepts, Models, Methods and Algorithms*. John Wiley & Sons, 2003.

[19] Laura Keyes and Adam Winstanley. Topographic object recognition through shape. Technical report, Ordnance Survey, Southampton, March 2001.

[20] James C. Bezdek Ludmila I. Kuncheva and Robert P.W. Duin. Decision Templates for Multiple Classifier Fusion: an experimental comparison. *Pattern Recognition*, 34(5), 2001.

[21] Ren C. Luo and Michael G. Kay. Multisensor Integration and Fusion on Intelligent Systems. *IEEE Transactions on Systems, Man and Cybernetics*, 19(5), September/October 1989.

[22] Maqsood A. Mohd. Sensitivity Analysis of ATR Algorithms to Scene Distortions: The Key to Performance Evaluation of Automatic Target Recognition Systems. IEEE, 1994.

[23] Andrew W. Moore. Cross-validation for detecting and preventing overfitting. Class notes, Carnegie Mellon University, `http://www.cs.cmu.edu/~awm/tutorials`.

[24] Robin R. Murphy and Brent Taylor. A Survey of Machine Learning Techiniques for Automatic Target Recognition. Submitted to IEEE Transactions on Pattern Analysis and Machine Intelligence.

[25] Frederic I. Parke. Class notes on neighborhood operations. Class notes - Texas A&M University, 2000.

[26] Peter E. Hart Richard O. Duda and David G. Stork. *Pattern Classification*. Wiley, second edition, Oct 2000.

[27] Mark Smith. *ARF Specification*. E-OIR Measurements, Inc. developed under contract to NVESD, 2 edition.

[28] Hairong Qi Xiaoling Wang and S. Sitharama Iyengar. Collaborative Multi-Modality Target Classification in Distributed Sensor Networks. In *Proceedings of the Fifth International Conference on Information Fusion*, July 2002.

[29] Kameda Yoshinari and Minoh Michihiko. A Human Motion Estimation Method Using 3-Successive Video Frames. IMEL, Kyoti University.

[30] Chunrui Zhang and M. Y. Siyal. A New Segmentation Technique for Classification of Moving Vehicles. In *Proceedings of the IEEE Vehicular Technology Conference*, May 2000.

Vita

Balasubramanian "Bala" Lakshminarayanan was born in the Kerala state of India on the 10th of February in 1982. He completed his elementary and secondary education in Madras under the Central Board for Secondary Education. He went on to obtain his Bachelor's degree in Electronics and Communication Engineering from the University of Madras in 2003. During this study period, he studied and implemented the International Telecommunication Union's (ITU-T) standard G-729A for compressing and decompressing audio signals for Voice over Internet Protocol (VoIP) applications. Due to his keen interest in signal processing, he undertook a course on Digital Signal Processors and Applications conducted by Analog Devices and Indian Institute of Technology-Madras. In addition, he underwent in-plant training at Videsh Sanchar Nigam Limited (VSNL), Chennai, a national communications provider and gateway. Bala joined The University of Tennessee, Knoxville for his graduate study in Fall 2003 and started working under Dr.Hairong Qi in the Advanced Imaging and Collaborative Information Processing (AICIP) lab from January 2004. His major topics of interest are image processing, pattern recognition and data fusion techniques. He studied the implementation of data fusion techniques to address the issue of Automatic Target Recognition of civilian targets. This thesis work is a compilation of his work towards this topic.