Affine Transform Motion Compensation For intermodal Cargo Identification

A Thesis
Presented to
The Academic Faculty

By

Jonathan Page Siplon

In Partial Fulfillment
of the Requirements for the Degree
Master of Science in Electrical and Computer Engineering

Georgia Institute of Technology

August 2005

Affine Transform Motion Compensation
For Intermodal Cargo Identification

Approved by:

Dr. Joel Jackson, Advisor
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Ashraf Saad
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Ghassan Al-Regib
School of Electrical and Computer Engineering
Georgia Institute of Technology

May 20, 2005

For Jan, Thomas and Lily

ACKNOWLEDGEMENT

I would first like to thank my advisor Dr. Joel Jackson for all his great advice, fun conversations and selflessly unwavering support; as well as my reading committee, for working with me to help make this a reality. I am also deeply indebted to Dr. David Frost and Mrs. Jill Parks for both believing in me and providing an academic atmosphere in which to learn, work, grow and excel.

I also cannot say thank you enough times to my core support team at home consisting of my loving wife Jan and my incredible son Thomas. Jan, this is the light at the end of the tunnel I kept promising... Really!

Thanks also to everyone who gave me encouragement, advice and/or the occasional kick in the pants to keep me going in the right direction. You know who you are!

TABLE OF CONTENTS

Acknowledgement iv

List of Tables vii

List of Figures viii

Glossary xi

Summary xv

Chapter 1 Introduction 1
- 1.1 The Maritime Shipping Industry: An Overview 1
  - 1.1.1 Georgia's Role 4
- 1.2 Seaport Security 7
  - 1.2.1 Cargo Containers 8
  - 1.2.2 Supply Chain Integrity 8

Chapter 2 Optical Character Recognition 10
- 2.1 What is Optical Character Recognition 10
- 2.2 Methodologies of the OCR Process 11
  - 2.2.1 Pre-Processing 11
  - 2.2.2 Training and Recognition Techniques 22
  - 2.2.3 Post-Processing 29
- 2.3 Commercial Applications of OCR 30

Chapter 3 Seaport Optimization 34
- 3.1 Operations Overview 34
  - 3.1.1 Water-side (Berth) Interface 36
  - 3.1.2 Ship Operations 37
  - 3.1.3 Yard Operations 38
  - 3.1.4 Land Side Interface 39
- 3.2 Challenges 41
  - 3.2.1 Overview 41
  - 3.2.2 Physical Container Limitations 42
  - 3.2.3 Identification Codes 42
  - 3.2.4 Environmental Factors 43
  - 3.2.5 The Current Process (motion compensation application) 44
- 3.3 Container Configuration 45
  - 3.3.1 Container Identification Codes: ISO 6346:1995 45

TABLE OF CONTENTS (continued)

Chapter 4 Motion Compensation 52
4.1 Theory of Motion Estimation and Compensation 52
4.2 Motion Estimation and Compensation Techniques 54
4.2.1 Translational Motion Compensation Methods 54
4.2.2 Higher-Order Motion Compensation Methods 66
4.2.3 Motion Compensated Temporal Filtering (MCTF) 74

Chapter 5 Results 76
5.1 Procedure 76
5.1.1 Overview 76
5.1.2 Translational Block Matching 79
5.1.3 Affine Transformation 81
5.2 Resulting Data and Images 82
5.2.1 Container Sequence 216_B 83
5.2.2 Container Sequence 034 87
5.2.3 Container Sequence 035 91
5.2.4 Container Sequence 206 95
5.2.5 Container Sequence 216 99
5.2.6 Container Sequence 08 103

Chapter 6 Conclusion 107

Appendix A: Matlab Code 109

References 121

LIST OF TABLES

Table 1.1	Top 10 Volume Ports: 2002	3
Table 3.1	ISO Container Identification Codes: Example Photographs	51
Table 5.1	Summary table of MAD values from residual images	84

LIST OF FIGURES

Figure 2.1	SAIC's Commercial OCR System	31
Figure 2.2	Commercial OCR for Railroads	32
Figure 2.3	Camco's OCR Solution	32
Figure 2.4	HTSOL's Truck Gate OCR System	33
Figure 2.5	Typical Camera Configuration	33
Figure 3.1	Map Included with "Truckers Guide to GPA"	35
Figure 3.2	Aerial Photo of the GPA Garden City Terminal	36
Figure 3.3	Three of the Seventeen Quay Cranes at GPA	36
Figure 3.4	Container Being Loaded Onto a Jockey Truck	37
Figure 3.5	RTG Moving a Container	38
Figure 3.6	Aerial Photo of the GPA Container Yard	39
Figure 3.7	Preferred ISO Code Horizontal Layout	48
Figure 3.8	Alternate ISO Code Horizontal Grouping Layout	48
Figure 3.9	Preferred ISO Code Vertical Layout	49
Figure 3.10	Alternate (Multiple Column) Vertical Layout	49
Figure 3.11	Location of Mandatory and Optional ISO Markings	50
Figure 4.1	Illustration of aperture problem	53
Figure 4.2	Block matching approach	59
Figure 4.3	Hierarchical block matching scheme	65
Figure 4.4	Mesh designs used in higher order motion model schemes	67
Figure 5.1	Block diagram of process	78
Figure 5.2	Original container image frames 215 and 216	83
Figure 5.3	Isolated section of original frame 216_B	83
Figure 5.4	Estimated frame 216_B using block matching	84

LIST OF FIGURES (continued)

Figure 5.5	Estimated frame 216_B using affine transformation	84
Figure 5.6	Residual image of block match estimate of frame 216_B	85
Figure 5.7	Residual image of affine estimate of frame 216_B	86
Figure 5.8	Original container image frames 033 and 034	87
Figure 5.9	Isolated section of original frame 034	87
Figure 5.10	Estimated frame 034 using block matching	88
Figure 5.11	Estimated frame 034 using affine transformation	88
Figure 5.12	Residual image of block match estimate of frame 034	89
Figure 5.13	Residual image of affine estimate of frame 034	90
Figure 5.14	Original container image frames 034 and 035	91
Figure 5.15	Isolated section of original frame 035	91
Figure 5.16	Estimated frame 035 using block matching	92
Figure 5.17	Estimated frame 035 using affine transformation	92
Figure 5.18	Residual image of block match estimate of frame 035	93
Figure 5.19	Residual image of affine estimate of frame 035	94
Figure 5.20	Original container image frames 205 and 206	95
Figure 5.21	Isolated section of original frame 206	95
Figure 5.22	Estimated frame 206 using block matching	96
Figure 5.23	Estimated frame 206 using affine transformation	96
Figure 5.24	Residual image of block match estimate of frame 206	97
Figure 5.25	Residual image of affine estimate of frame 206	98
Figure 5.26	Original container image frames 215 and 216	99
Figure 5.27	Isolated section of original frame 216	99
Figure 5.28	Estimated frame 216 using block matching	100

LIST OF FIGURES (continued)

Figure 5.29 Estimated frame 216 using affine transformation 100

Figure 5.30 Residual image of block match estimate of frame 216 101

Figure 5.31 Residual image of affine estimate of frame 216 102

Figure 5.32 Original container image frames 07 and 08 103

Figure 5.33 Isolated section of original frame 08 103

Figure 5.34 Estimated frame 08 using block matching 104

Figure 5.35 Estimated frame 08 using affine transformation 104

Figure 5.36 Residual image of block match estimate of frame 08 105

Figure 5.37 Residual image of affine estimate of frame 08 106

GLOSSARY

Berth: The area allotted to accommodate a vessel alongside a wharf, or the area in which a vessel swings when at anchor.

Break-bulk: Non-containerized general cargo. Examples include iron, steel, machinery, linerboard, wood pulp and yachts.

Chart Datum: A plan below which the tide will seldom fall. The Canadian Hydrographic Service has adopted the plane of Lowest Normal Tides (LNT) as chart datum. To find the depth of water, the height of tide must be added to the depth shown on the chart. Tidal heights preceded by a (-) must be subtracted from the charted depth. Note: United States tidal datum is Mean Low Water and can differ from Canadian datum by as much as 1.50 meters.

Charter Party: The contract of hire for a ship.

Consolidated Freight Station or Container Freight Station (CFS): Location on terminal grounds where stuffing and stripping of containers is conducted.

Container: A 20, 35, 40 or 45 foot box which can be handled interchangeably among trucks, railcars, barges and ocean going vessels.

Container on Flat Car (COFC): A container placed directly on a railroad flatcar without chassis.

Customs Broker: Performs duties related to documentation, cargo clearance, coordination of inland and ocean transportation, dockside inspection of cargo, etc. Employed by the importer.

Deadweight Tonnage (DWT): Maximum weight including cargo, ballast, and stores that can be loaded into a vessel.

Displacement: The total weight of the vessel, i.e. the amount of water the vessel will displace.

Dock: A structure built along or at an angle form a navigable waterway so that vessels may lie alongside to load and discharge cargo.

Dry Bulk: Minerals or grains stored in loose piles moving without mark or count. Examples are potash, salts, sugar and aggregate.

Electronic Data Interchange (EDI): The exchange of information through an electronic format. Used by many various parties for hundreds of types of data.

GLOSSARY (continued)

Feeder Service: Ocean transport system involving use of centralized ports to assemble and disseminate cargo to and from ports within a geographic area. Commodities are transported between major ports then transferred to feeder vessels for further transport to a number of additional ports.

Freight Forwarder: Consultant in logistics and international traffic. The forwarding agent assists the exporter in finding the most economic and efficient methods of transporting and storing cargo.

Gantry (Quay) Crane: Track-mounted crane utilized in the loading and unloading of break-bulk cargo, containers and heavy lifts.

Gross Tonnage: The capacity of the vessel calculated by an approved formula. The gross tonnage is a figure representative of the volume of the enclosed space in cubic meters (1969 Tonnage Convention).

Hopper Car: A freight car used for handling dry bulks, with an operable top and one or more openings on the bottom through which the cargo is dumped.

Interchange: Point of entry/exit for trucks delivering and picking up containerized cargo. Point where pickups and deposits of containers in storage area or yard are assigned.

Intermodal: Relating to cargo which can be handled interchangeably among different transportation modes, i.e. truck, rail, ocean and air.

LCL: Less than container load.

Length Overall (LOA): Linear measurement of a vessel from bow to stern.

Lift-On/Lift-Off (LO/LO): Cargo handling technique involving transfer of commodities to and from the ship using shore side cranes or ship's gear.

Liner Service: Sailings between specified ports on a regularly scheduled basis.

Liquid Bulk: Cargo which is transported and stored in liquid form, other than in a drum or similar vessel.

Longshoremen: Individuals who perform services under the direction of a terminal operator or stevedoring company such as operating equipment, rigging cargo or administrative tasks associated with the loading or unloading of a vessel.

Long Ton: A long ton equals 2,240 pounds or 1,016 kilograms.

Marshaling Yard: Any open area for either assembly of cargo for export or placement of imported cargo awaiting inland transport.

GLOSSARY (continued)

Mean Low Water (MLW): Lowest average level water reaches on an outgoing tide.

Mean High Water (MHW): Highest average level water reaches on an outgoing tide.

Tonne or Metric Tonne: 2,204.6 pounds.

Motor Ship (MS) or Motor Vessel (MV): A ship propelled by internal-combustion engines.

On-Dock Rail: Direct shipside rail service. Includes the ability to load and unload containers / break-bulk directly from rail car to vessel.

On-Terminal Rail: Railroad service and track access provided within a designated terminal area.

Reefer: Refrigerated cargo, whether break-bulk or containerized. Also refers to a ship's capability to handle such cargo, and storage areas, containers, etc., used to store and transport them.

Roll-on/Roll-off (RO/RO): Transportation mode utilizing ramp equipped vessels where wheeled equipment and cargo on flatbeds can be driven on or off.

Rubber-Tired Gantry (RTG): Traveling crane used for the movement and positioning of containers in a container field. RTG's may also be used for loading and unloading containers from rail cars.

Short Ton: A short ton equals 2,000 pounds or 907 kilograms.

Steamship Line: Organization that operates ocean carriers/vessels to transport cargo.

Stevedore: Agency retained by the vessel operator or agent to determine the method cargo is to be loaded or discharged and to provide the necessary equipment and labor to execute the handling and supervise the actual handling process.

Shipper: Organization responsible for the packaging and shipping of a commodity.

Stripping: The process of removing cargo from a container, or the process for pumping the last of the cargo from a liquid bulk carrier.

Stuffing: The process of packing a container with loose cargo prior to inland or ocean shipment.

Top-lift: A piece of equipment similar to a forklift that lifts cargo, usually containers, from above rather than below. Top-lifts are used to handle containers in the storage yard to and from storage stacks, trucks and railcars.

GLOSSARY (continued)

Trailer on Flat Car (TOFC): A container placed on a chassis which is in turn placed on a railroad flat car.

Tramp ship: Any ship other than a liner (see above) that generally operates a charter party and can call at any port for cargo.

Transit Shed: Located dockside, these buildings are used for temporary storage of commodities just before export and immediately following import.

Twenty Foot Equivalent Unit (TEU): A unit of measurement equal to the space occupied by a standard twenty foot container. Used in stating the capacity of container vessel or storage area. One 40 ft. Container (FEU) is equal to two TEU's.

SUMMARY

The volume of cargo flowing through today's transportation system is growing at an ever increasing rate. Recent studies show that 90% of all international cargo that enters the United States flows through our vast seaport system. When this cargo enters the US, time is of the essence to quickly obtain and verify its identity, screen it against an ever increasingly wide variety of security concerns, and ultimately correctly direct the cargo towards its final destination.

Over the past few years, new port and container security initiatives and regulations have generated huge interest in the need for accurate real-time identification and tracking of incoming and outgoing traffic of vehicles and cargo. On the contrary, the manually intensive identification and tracking processes, typically employed today, are inherently both inefficient and inadequate, and can be seen as a possible enabling factor for potential threats to our ports and therefore our national security. The contradiction between current and required processes coupled to the correlation with accelerated growth in container traffic, has clearly identified the need for a solution.

One heavily researched option is the utilization of video based systems implementing Optical Character Recognition (OCR) processes for automatically extracting the unique container identification code to expedite the flow of cargo through various points in the seaport. The actual current process of how this occurs along with the opportunities and challenges for adding such a technological solution will be investigated in great detail.

This thesis will investigate the feasibility of application of motion compensation algorithms as an enhancement to OCR systems specifically designed to address the challenges of OCR of cargo containers in a seaport environment. This motion compensation could offer a cost effective alternative to the sophisticated hardware systems currently being offered to US ports.

CHAPTER 1

INTRODUCTION

1.1 The Maritime Shipping Industry: An Overview

The United States is the largest trading nation in the world for both imports and exports, accounting for nearly 20 percent of world trade in goods. Over the last ten years the value of America's international trade in goods has nearly doubled, and rapid growth is expected in the next decade as well. The number of enterprises involved in America's international trade is substantial. In 2002, approximately 202,800 U.S. importers received goods from more than 178,200 foreign exporters via liner shipping. The combined value of U.S. exports and imports of goods in 2003 was approximately $1.98 trillion dollars. Of that amount, approximately $807.1 billion was international waterborne trade arriving at or departing from U.S. ports.

The majority of this cargo is containerized – that is, it is carried in sealed metal containers from point of origin to destination. In 2003, two-thirds or $491.2 billion, of all cargo was transported in a containerized fashion. This total averages to approximately $1.34 billion worth of containerized goods moving through domestic ports each day.

These containers usually come in standard sizes, typically 20', 40', 45' and 53' in length, and may include various specialized technologies, such as refrigeration units, or internal hanger systems for carrying clothing. Containers serve as both packing crate and in-transit warehouse for virtually every type of cargo moving in international commerce. The standard measure of the volume of containerized cargo is a TEU (Twenty-foot Equivalent Unit). For

example, one forty-foot long container of cargo, the most common size in U.S. trades, would be counted as two TEUs of cargo.

The worldwide fleet of marine containers in circulation in mid-2003 is estimated to be about 10.8 million containers with overall capacity of almost 17 million TEUs. There are more than 4 million containers in use at any given time in the U.S. trades. One forty-foot long container (2 TEUs) can hold 20,000 toy dolls, or 6,600 dresses (on hangers), or 3,600 men's suits (on hangers), or approximately 500 computer monitors.

In 2003, approximately 21.3 million TEUs of containerized cargo were imported into or exported from the U.S. on roughly 1,040 different individual containerships making about 17,250 total port calls. Ships longer than three football fields, and specially designed for efficient stacking of containers, can carry more than 6,000 TEUs of cargo on each voyage. The majority of the containerized cargo in the U.S. moves from inland point to inland point via a multi-modal network linking vessels, seaports, trucks and trains. At the heart of this service network is the planning, tracking and delivery of cargo and state-of-the-art information systems needed to provide certainty and reliability to the hundreds of thousands of American companies engaged in international waterborne commerce.

Container cargo arriving from and departing to America's foreign markets passes across the docks of a select group of U.S. seaports with terminals specially equipped to efficiently handle thousands of containers a day – moving them rapidly on and off ships, onto and off of truck chassis and rail cars, and into and out of port areas.

In 2003, the top 25 U.S. container ports handled 98% of the total volume of our containerized waterborne cargo, with over 83% moving through the top 10 ports and 61%

through the top 5 ports. In fact, the port complex of LA/Long Beach in Southern California alone accounted for over 36% of all containerized U.S. imports and exports in 2003.

Table 1.1 Top 10 Volume Ports: 2002

U.S. Waterborne Foreign Trade Top 30 U.S. Ports (Container Cargo) Calendar Year 2003 (Thousand TEUs)			
U.S. Ports	**Total**	**Export**	**Import**
Los Angeles	4,664	1,022	3,642
Long Beach	3,091	723	2,368
New York	2,803	838	1,965
Charleston SC	1,250	529	721
Savannah	**1,124**	**529**	**595**
Norfolk	1,093	460	633
Oakland	1,064	548	517
Houston	933	483	450
Tacoma	931	337	594
Seattle	815	329	486

Worldwide, the liner shipping industry has purchased or assumed responsibility for approximately $155 billion in direct operating assets, such as vessels, containers, chassis, and marine terminals that are now in service. Of that sum, approximately $40 billion worth of assets are devoted to the carriage of American imports and exports. And, given the

expected growth of liner cargo, the industry will likely invest another $35 billion over the next decade just to accommodate U.S. trade growth.

1.1.1 Georgia's Role

The following information gives a high level overview of the significant role the Georgia Ports plays in the overall scheme of maritime shipping. This information is included as reference because the operations of the Georgia Ports served as the base for many of the investigations of this thesis. The following was quoted directly from the GPA:

"Since 1945, Georgia's ports have served as magnets for international trade and investment, enriching the state's economy to benefit all Georgians. The Georgia Ports Authority (GPA) is dedicated to providing customers with the most efficient, productive port facilities in the nation, and to creating jobs and business opportunities to benefit more than 8.6 million Georgians. The GPA is committed to maintaining its competitive edge through development of leading-edge technology, marketing and operation move cargo faster than the competition. And, the Authority is working hard to identify what must be done today to sustain growth, performance and security for tomorrow.

Georgia's deepwater ports in Savannah and Brunswick, together with inland barge operations in Bainbridge and Columbus, are Georgia's gateways to the world. They are the critical conduits through which raw materials and finished products flow to and from destinations around the globe.

As a quasi-state agency, a thirteen-member Board of Directors governs the activities of the GPA. The Board is appointed by the Governor, from the state at large, to serve four-year, staggered terms. A Chief Executive Officer, an experienced international transportation

professional, implements policy directives, administrative duties and managerial controls.

As one of the state's largest public employers, the GPA directly employs more than 780 people. The GPA, however, is responsible for generating far more employment throughout the state. GPA operations, together with private sector, port-related operations, account for more than 275,968 jobs statewide, billions of dollars in revenue, and income exceeding $10.8 billion annually.

The Port of Savannah, home to the largest single-terminal container facility of its kind on the U.S. East and Gulf coasts, is comprised of two modern, deepwater terminals: Garden City Terminal and Ocean Terminal. Together, these facilities exemplify the GPA's exacting standards of efficiency and productivity. Garden City Terminal is one of the top five container handling facilities in the United States, encompassing more than 1,200 acres and moving millions of tons of containerized cargo annually.

Ocean Terminal, Savannah's dedicated break-bulk and Roll-on / Roll-off (Ro-Ro) facility, covers 208 acres and provides customers with more than 1.5 million square feet of covered, versatile storage.

The Port of Brunswick is comprised of three GPA-owned deepwater terminals, two of which are directly operated by the GPA. The port's well-earned reputation for productivity and efficiency is heightened by its position as one of the fastest growing auto and heavy machinery ports in North America. Today, more than 12 major auto manufacturers, supported by three auto processors, utilize the Colonel's Island Terminal. The terminal is also home to the South Atlantic's fastest growing bulk export / import operation. Agricultural products from Georgia and the rich U.S. grain belt, as well as import products, flow smoothly

across the Colonel's Island docks.

Brunswick's Mayor's Point Terminal facilitates the export of Georgia's valuable forest products, while Marine Port Terminals, operated by Logistec U.S.A., specializes in the handling of break-bulk and bulk commodities.

Though Georgia's inland river terminals have been adversely impacted by drought, Port Bainbridge and Port Columbus have been re-tooled to serve as low-cost transportation and distribution facilities. These ports are now providing a strategic advantage for commodities moving to and from the region.

Through leadership, a desire to excel and proven success, the Georgia Ports Authority remains steadfast in its resolve to create economic opportunities for the citizens of Georgia and to grow customer business."

1.2 Seaport Security

Since the events of September 11, 2001, port security has emerged as a significant part of the overall debate on U.S. homeland security. Shipping companies have begun making additional investments in new technologies and process enhancements in an attempt to bolster the security of their own assets and by repercussion are helping to secure America's international cargo transportation. Many security experts believe ports are vulnerable to terrorist attack because of their size, easy accessibility by water and land, and the sheer tremendous volume of cargo they handle. According to conservative estimates, the initial start-up expenditures and on-going costs will run to several tens of millions of dollars per year for US container trades alone.

Even before the terrorist attacks of September 11, 2001, government officials and security experts were concerned about the security of U.S. ports. In the fall of 2000, the Interagency Commission on Crime and Security in U.S. Seaports noted the vulnerability of U.S. seaports to terrorism. The report noted that while the FBI then considered the threat of terrorist attacks on U.S. seaports to be low; their vulnerability to such attacks was high.

Government leaders and security experts worry that the maritime transportation system could be used by terrorists to smuggle personnel, weapons of mass destruction, or other dangerous materials into the United States. They are also concerned that ships in U.S. ports, particularly large commercial cargo ships or cruise ships, could be attacked by terrorists. A large-scale terrorist attack at a U.S. port, experts warn, could not only cause local death and damage, but also paralyze global maritime commerce.

1.2.1 Cargo Containers

Container ships are a growing segment of maritime commerce – and thus are the focus of much of the attention on seaport security. More than 6 million cargo containers enter US seaports every year. For comparison, about 13 million containers arrive by truck or rail from Canada and Mexico. The United States Customs and Border Patrol (USCBP) analyze manifest information about each container and the cargo claimed to be inside, to decide which containers are to be targeted for closer inspection. The decision to inspect is based on a complex algorithm including such factors as origin, destination, shipper and claimed container contents. Many details of the targeting process are kept secret for obvious reasons. Only a small portion (5% on average) has their content physically inspected by USCBP. Physical inspection could include non-intrusive scanning the entire container with a sophisticated x-ray or gamma ray machine (called a VACIS scanner), unloading the contents of a container, or both.

1.2.2 Supply Chain Integrity

The complexity of the process for completing containerized shipments makes it more difficult to ensure the integrity of this type of cargo. Unlike other cargo ships whose loading process occurs at the port and whose cargo is often owned by a single company, container ships carry cargo from hundreds of companies and the containers are loaded away from the port at individual company warehouses. A typical single container shipment may involve a multitude of parties and generate thirty to forty documents. A single container could also carry cargo for several customers, thus exponentially multiplying the number of parties and documents involved.

The parties involved in a shipment usually include the exporter, importer, freight forwarder, customs broker, customs inspector, inland transportation provider(s) (which often includes

more than one trucker and trucking company or railroad), the port terminal operators, possibly a feeder ship or distribution center personnel and the ocean carrier.

Each transfer of the container from one party to the next is a point of vulnerability in the supply chain. The security of each transfer facility and the trustworthiness of each company are therefore critical in the overall security of the shipment. It is also important to keep in mind that not all US bound containers arrive at US ports. Half of the containers discharged at the Port of Montreal, for instance, move by truck or rail for cities in the Northeastern or Midwestern United States. Likewise, many containers that enter US waters are bound for other Nations.

CHAPTER 2

OPTICAL CHARACTER RECOGNITION

2.1 What is Optical Character Recognition (OCR)?

OCR is the abbreviated name for Optical Character Recognition. This technology allows a machine to recognize characters through an optical device or analysis of an image. Humans recognize objects in this manner with our eyes as our "optical device." While the brain "sees" the input, the ability to comprehend these signals varies in each person according to many factors. By reviewing these differences, we can understand the challenges faced by the researcher developing an OCR system.

For example, there is great similarity between many numerical and alphabetical symbols. While examining a string of characters combining letters and numbers, there is very little visible difference between a capital letter "O" and the numeral "0." Humans can re-read the previous sentence or even the entire paragraph to help determine the accurate representation. However, it is much more difficult for a machine to have this sort of contextual intelligence.

We rely on contrast to help us recognize characters. We may find it very difficult to read text which appears against a very dark background, or is printed over other words or graphics. Programming a system to interpret only relevant data and disregard the rest is also a very difficult task.

Machine simulation of human functions has been a very challenging research field since the advent of digital computers. In some areas, requiring certain amounts of intelligence, such as mathematical operations or playing chess, tremendous improvements have been obtained. However, humans still outperform even the most powerful computers in routine functions such as vision. Machine simulation of human reading is one of these areas, which has been the subject of intensive research for the last three decades, yet it is still far from the final frontier.

There are many other problems which challenge the developers of OCR systems. This section of the thesis will explore the various inner-workings of the OCR process and its applicability to a reading ISO container codes on intermodal cargo containers.

2.2 Methodologies of the OCR Process

There are generally five major stages in most OCR applications:

1) Pre-Processing
2) Segmentation
3) Representation
4) Training and Recognition
5) Post-Processing

In some methods, some of these stages are merged and/or omitted. In some other applications a feedback mechanism is used to update the output of each stage.

2.2.1 Pre-Processing

Raw data, depending on the data collection type, is subjected to a number of preliminary processing steps to make it usable in the descriptive stages of character analysis. The goal

of pre-processing is to help groom data so it is easier for the OCR systems to operate on accurately. This is a very important area of focus for the application in a seaport environment, as it is this stage that motion compensation would be applied.

The main objectives of pre-processing are

1) noise reduction
2) normalization of the data
3) compression of information to be retained

In order to achieve the above objectives, the following techniques are used in the pre-processing stage.

Noise Reduction

Noise, introduced by the quality of the optical device or more specific to the seaport application is background image noise from the irregular surface of an container described previously. Regardless of the various ways for noise to occur, it causes disconnected line segments, bumps and gaps in lines, filled loops, etc. The distortion, including local variations, rounding of corners, dilation, and erosion, is also a problem. Prior to the actual OCR, it is important to eliminate these imperfections. Hundreds of available noise reduction techniques can be categorized in three major groups [153], [161].

Filtering

This aims to remove noise and diminish spurious points, usually introduced by poor sampling rate of the data acquisition device. Various spatial and frequency domain filters can be designed for this purpose. The basic idea is to convolute a predefined mask with the image to assign a value to a pixel as a function of the gray values of its neighboring pixels.

Filters can be designed for smoothing [104], sharpening [105], thresholding [119], removing slightly textured or colored background [101], and contrast adjustment purposes [142].

Morphological Operations

Morphological operations filter the document image replacing the convolution operation by logical operations. Various morphological operations can be designed to connect the broken lines [7], decompose connected lines [26], smooth contours, reduce wild points, [153], thin characters [147] and extract the boundaries [185]. Therefore, morphological operations can be successfully used to remove the noise on the images due to low quality of the characters themselves.

Noise Modeling

Noise could be removed by some calibration techniques if a model for it were available. However, modeling the noise is not possible in most applications. This is specifically true for a seaport, where it is impossible to tell where and how the noise will occur. There is very little work on modeling noise introduced by optical distortion, such as speckle, skew, and blur [10], [92]. Nevertheless, it is possible to assess the quality of the images and remove the noise to a certain degree, as suggested in [21].

Normalization

Normalization methods aim to remove the variations of the characters and obtain as standardized a data-set as possible. The following are the basic methods for normalization [40], [55].

1) skew normalization
2) slant normalization
3) size normalization
4) contour smoothing

Iijima also wrote two very important papers in 1962 and 1963 regarding the theory of normalizing patterns [32] and the theory of feature extraction [33]. These papers have had a great influence on the field of OCR and pattern recognition. Iijima proved that the mathematical form of the transformation must be a convolution of the signal $f(x')$ with a Gaussian kernel. This kernel produced a blurring transformation. Initially the research was to try to recover the blurring effect from the imaging system and obtain normalization of the blurring. This quickly led to use the blurring transformation on the input characters instead of the output. Today, the blurring preprocessing step is a widely used process in different types of pattern recognition.

Skew and slant normalization (below) are not typically an issue in the application to preparing container code for OCR. These normalization processes are typically needed in handwritten text and are included for reference and completeness.

Skew Normalization

Due to inaccuracies in the image capture process, the characters might be slightly tilted or curved within the image. This can hurt the effectiveness of later algorithms and, therefore, should be detected and corrected. Additionally, some characters are distinguished according to the relative position with respect to the baseline (e.g., “9” and “g”). Methods of baseline extraction include using the projection profile of the image [78], a form of nearest neighbors clustering [61], cross correlation method between lines [31], and using the Hough transform [189]. In [134], an attractive repulsive NN is used for extracting the baseline of complicated handwriting in heavy noise. After skew detection, the character or word is translated to the origin, rotated, or stretched until the baseline is horizontal and retranslated back into the display screen space.

Slant Normalization

One of the measurable factors of different handwriting styles is the slant angle between the longest stroke in a word and the vertical direction. Slant normalization is used to normalize all characters to a standard form. The most common method for slant estimation is the calculation of the average angle of near-vertical elements. In [111], vertical line elements from contours are extracted by tracing chain code components using a pair of one-dimensional (1-D) filters. Coordinates of the start and end points of each line element provide the slant angle. Another study [56] uses an approach in which projection profiles are computed for a number of angles away from the vertical direction. The angle corresponding to the projection with the greatest positive derivative is used to detect the least amount of overlap between vertical strokes and, therefore, the dominant slant angle.

Size Normalization

This is used to adjust the character size to a certain standard. Methods of OCR may apply both horizontal and vertical size normalizations. In [186], the character is divided into number of zones and each of these zones is separately scaled. Size normalization can also be performed as a part of the training stage, and the size parameters are estimated separately for each particular training data [5].

Size normalization is of possible use for container codes to help standardize the data-set as much as possible. As described previously, the font as well as the size of the container characters will vary randomly from container to container.

Contour Smoothing

It generally reduces the number of sample points needed to represent the script, thus improving efficiency in remaining preprocessing steps [7], [104]. This smoothing is similar to the blurring transformation mentioned previously.

Compression

It is well known that classical image compression techniques transform the image from the space domain into domains which are not suitable for recognition. Compression for OCR requires space domain techniques for preserving the shape information. Two popular compression techniques are thresholding and thinning.

Thresholding

In order to reduce storage requirements and to increase processing speed, it is often desirable to represent gray-scale or color images as binary images by picking a threshold value. Two categories of thresholding exist: global and local. Global thresholding picks one threshold value for the entire document image which is often based on an estimation of the background level from the intensity histogram of the image [160]. Local (adaptive) thresholding use different values for each pixel according to the local area information [151].

Thinning

While it provides a tremendous reduction in data size, thinning extracts the shape information of the characters. Two basic approaches for thinning are 1) pixel wise and 2) non-pixel wise thinning [97]. Pixel wise thinning methods locally and iteratively process the image until one pixel wide skeleton remains. They are very sensitive to noise and may deform the shape of the character. Non-pixel wise methods use some global information

about the character during the thinning. They produce a certain median or centerline of the pattern directly without examining all the individual pixels [11].

Segmentation

The pre-processing stage yields a “clean” document in that shape information and low noise on a normalized image is obtained. The next stage is segmenting the document into its subcomponents. Segmentation is an important stage because the extent one can reach in separation of characters directly affects the recognition rate of the overall procedure. There are two types of segmentation: external and internal.

External Segmentation

It is the most critical part of the image analysis, which is a necessary step prior to the actual OCR and simply includes segmenting the image into text and non-text regions.

Internal Segmentation

This is the process of breaking the character into individual segments. These strategies are divided into three categories [22].

1) Explicit segmentation
2) Implicit segmentation
3) Mixed segmentation

Explicit Segmentation:

In this strategy, the segments are identified based on “character-like” properties. The process of cutting up the image into meaningful components is given a special name: dissection.

Implicit Segmentation:

This segmentation strategy is based on recognition. It searches the image for components that match predefined classes. Segmentation is performed by the use of recognition confidence, including syntactic or semantic correctness of the overall result.

Mixed Strategies:

Combines explicit and implicit segmentation in a hybrid way. A dissection algorithm is applied to the image, but the intent is to "over segment," i.e., to cut the image in sufficiently many places that the correct segmentation boundaries are included among the cuts made.

Representation

Image representation plays one of the most important roles in a recognition system. In the simplest case, gray-level or binary images are fed to a recognizer. However, in most of the recognition systems, in order to avoid extra complexity and to increase the accuracy of the algorithms, a more compact and characteristic representation is required. For this purpose, a set of features is extracted for each class that helps distinguish it from other classes while remaining invariant to characteristic differences within the class [131]. In the following, hundreds of document image representation methods are categorized into three major groups.

1) Global Transformation and Series Expansion
2) Statistical Representation
3) Geometrical and Topological Representation

Global Transformation and Series Expansion

A continuous signal usually contains more information than needs to be represented for the purpose of classification. This may be true for discrete approximations of continuous signals as well. One way to represent a signal is by a linear combination of a series of simpler well-defined functions. The coefficients of the linear combination provide a compact encoding known as transformation or/and series expansion. Deformations like translation and rotations are invariant under global transformation and series expansion. Common transform and series expansion methods used in the OCR field and include the following.

Fourier Transforms:

The general procedure is to choose magnitude spectrum of the measurement vector as the features in an n-dimensional Euclidean space.

Gabor Transform:

This is a variation of the windowed Fourier transform. In this case, the window used is not a discrete size but is defined by a Gaussian function [62].

Wavelets:

Wavelet transformation is a series expansion technique that allows us to represent the signal at different levels of resolution.

Moments:

Moments, such as central moments, Legendre moments, and Zernike moments, form a compact representation of the original document image that make the process of recognizing an object scale, translation, and rotation invariant [34], [83]. Moments are

considered as series expansion representation since the original image can be completely reconstructed from the moment coefficients.

Karhunen–Loeve Expansion:

This is an eigenvector analysis which attempts to reduce the dimension of the feature set by creating new features that are linear combinations of the original ones. It is the only optimal transform in terms of information compression. Karhunen–Loeve expansion is used in several pattern recognition problems such as face recognition.

Statistical Representation:

Representation of an image by statistical distribution of points takes care of style variations to some extent. Although this type of representation does not allow the reconstruction of the original image, it is used for reducing the dimension of the feature set providing high speed and low complexity.

Geometrical and Topological Representation:

Various global and local properties of characters can be represented by geometrical and topological features with high tolerance to distortions and style variations. This type of representation may also encode some knowledge about the structure of the object or may provide some knowledge as to what sort of components make up that object. Hundreds of topological and geometrical representations can be grouped into the following categories.

Extracting and Counting Topological Structures:

In this representation, a predefined structure is searched in a character or word. The number or relative position of these structures within the character forms a descriptive representation. Common primitive structures are the strokes, which make up a character.

These primitives can be as simple as lines and arcs which are the main strokes of Latin characters and can be as complex as curves and splines making up Arabic or Chinese characters

Measuring and Approximating the Geometrical Properties:

In many studies [36] and [94], the characters are represented by the measurement of the geometrical quantities such as the ratio between width and height of the bounding box of a character, the relative distance between the last point and the last y-min, the relative horizontal and vertical distances between first and last points, distance between two points, comparative lengths between two strokes, width of a stroke, upper and lower masses of words, and word length. A very important characteristic measure is the curvature or change in the curvature [133].

Coding:

One of the most popular coding schemas is Freeman's chain code. This coding is essentially obtained by mapping the strokes of a character into a 2-D parameter space, which is made up of codes. There are many versions of chain coding. As an example, in [57], the character frame is divided to left–right sliding window and each region is coded by the chain code.

Graphs and Trees:

Words or characters are first partitioned into a set of topological primitives, such as strokes, loops, cross points, etc. Then, these primitives are represented using attributed or relational graphs [106].

The major goal of representation is to extract and select a set of features, which maximizes the recognition rate with the least amount of elements. In [87], feature extraction and selection is defined as extracting the most representative information from the raw data, which minimizes the within class pattern variability while enhancing the between class pattern variability.

2.2.2 Training and Recognition Techniques

OCR systems extensively use the methodologies of pattern recognition, which assigns an unknown sample into a predefined class. Numerous techniques for OCR can be investigated in four general approaches of pattern recognition, as suggested in [76]:

1) Template Matching
2) Statistical Techniques
3) Structural Techniques
4) Neural Networks (NN's)
5) Combined Techniques

Occasionally, an OCR technique in one approach can also be considered to be a member of other approaches. In all of the above approaches, OCR techniques use either holistic or analytic strategies for the training and recognition stages: holistic strategy employs top–down approaches for recognizing the full word, eliminating the segmentation problem. The price for this computational savings is to constrain the problem of OCR to limited word set. However, the analytic strategies employ bottom–up approaches starting from stroke or character level and going toward producing a whole word. Explicit or implicit segmentation algorithms are required for this strategy, not only adding extra complexity to the problem, but also introducing segmentation error into the system. If the segmentation stage was carried

out effectively previously, the problem is reduced to the recognition of simple isolated characters or strokes.

Template Matching

OCR techniques vary widely according to the feature set selected from the long list of features described in the previous section for image representation. Features can be as simple as the gray-level image frames with individual characters or words or as complicated as graph representation of character primitives. The simplest way of OCR is based on matching the stored prototypes against the character or word to be recognized. Generally speaking, matching operation determines the degree of similarity between two vectors (group of pixels, shapes, curvature, etc.) in the feature space. Matching techniques can be studied in three groups.

Direct Matching

A gray-level or binary input character is directly compared to a standard set of stored prototypes. According to a similarity measure (e.g., Euclidean, Mahalanobis, Jaccard, or Yule similarity measures, etc.), a prototype matching is done for recognition.

The matching techniques can be as simple as one-to-one comparison or as complex as decision tree analysis in which only selected pixels are tested. A template matcher can combine multiple information sources, including match strength and k-nearest neighbor measurements from different metrics [46], [175]. Although direct matching method is intuitive and has a solid mathematical background, the recognition rate of this method is very sensitive to noise.

Deformable Templates and Elastic Matching

An alternative method is the use of deformable templates, where an image deformation is used to match an unknown image against a database of known images. In [73], two characters are matched by deforming the contour of one to fit the edge strengths of the other. A dissimilarity measure is derived from the amount of deformation needed, the goodness of fit of the edges, and the interior overlap between the deformed shapes.

The basic idea of elastic matching is to optimally match the unknown symbol against all possible elastic stretching and compression of each prototype. Once the feature space is formed, the unknown vector is matched using dynamic programming and a warping function [68], [169].

Relaxation Matching

It is a symbolic level image matching technique that uses feature-based description for the character image. First, the matching regions are identified. Then, based on some well-defined ratings of the assignments, the image elements are compared to the model. Recognition is done by computing the matching probabilities between two features by a relaxation method.

The matching techniques mentioned above are sometimes used individually or sometimes they are combined in various ways as part of a composite style OCR scheme.

Statistical Techniques

Statistical decision theory is concerned with statistical decision functions and a set of optimality criteria, which maximizes the probability of the observed pattern given the model of a certain class [38]. Statistical techniques are mostly based on three major assumptions:

- Distribution of the feature set is Gaussian or in the worst case uniform.
- There are sufficient statistics available for each class.
- Given ensemble of images, one is able to extract a set of features

The measurements taken from n features of each word unit can be thought to represent an n-dimensional vector space and the vector, whose coordinates correspond to the measurements taken, represents the original word unit. The major statistical approaches applied in the OCR field are listed below.

Non-parametric Recognition

This method is used to separate different pattern classes along hyper-planes defined in a given hyper-space. The best known method of non-parametric classification is the Neural Network and is extensively used in OCR [159]. It does not require *a priori* information about the data. An incoming pattern is classified using the cluster, whose center is the minimum distance from the pattern over all the clusters.

Parametric Recognition

Since *a priori* information is available about the characters in the training data, it is possible to obtain a parametric model for each character [14]. Once the parameters of the model, which are based on some probabilities, are obtained, the characters are classified according to some decision rules such as maximum likelihood or Bayes' method.

Clustering Analysis

The clusters of character features, which represent distinct classes, are analyzed by way of clustering methods. Clustering can be performed either by agglomerative or divisive algorithms.

Agglomerative algorithms operate step-by-step merging of small clusters into larger ones by a distance criterion, while divisive methods split the character classes under specified rules with the purpose of identifying the underlying character [188].

Hidden Markov Modeling (HMM)

HMM's are the most widely and successfully used technique for OCR situations [26], [27], [89], [120], [121]. It is defined as a stochastic process generated by two interrelated mechanisms: a Markov Chain having a finite number of states and a set of random functions, each of which is associated with a state [145]. At discrete instants of time, the process is assumed to be in some state, and an observation is generated by the random function corresponding to the current state. The underlying Markov chain then changes states according to its transitional probabilities. Here, the job is to build a model that explains and characterizes the occurrence of the observed symbols [77]. The output corresponding to a single symbol can be characterized as discrete or continuous. Discrete outputs may be characters from a finite alphabet or quantized vectors from a codebook, while continuous outputs are represented by samples from a continuous waveform. In generating a word or a character, the system passes from one state to another, each state emitting an output according to some probabilities until the entire word or character is out. The major design issue in the HMM problem is the selection of the feature set and HMM topology. These two tasks are strongly related to each other, and there is no systematic approach developed for this purpose.

Fuzzy Set Reasoning

Instead of using a probabilistic approach, this technique employs fuzzy set elements in describing the similarities between the features of the characters. Fuzzy set elements give more realistic results when there is no *a priori* knowledge about the data, and therefore, the probabilities cannot be calculated. The characters can be viewed as a collection of strokes, which are compared to reference patterns by fuzzy similarity measures. Since the strokes under consideration are fuzzy in nature, the concept of fuzziness is utilized in the similarity measure. In order to recognize a character, an unknown input character is matched with all the reference characters and is assigned to the class of the reference character with the highest score of similarity among all the reference characters.

Structural Techniques

The recursive description of a complex pattern in terms of simpler patterns based on the shape of the object was the initial idea behind the creation of structural pattern recognition. These patterns are used to describe and classify the characters in the OCR systems. The characters are represented as the union of the structural primitives. It is assumed that the character primitives extracted from writing are quantifiable, and one can find the relations among them. The following are structural methods that are currently applied to OCR schemes.

Grammatical Methods

In the mid-1960s, researchers started to consider the rules of linguistics for analyzing the speech and writing. Later, various orthographic, lexicographic, and linguistic rules were applied to the recognition schemes. The grammatical methods create some production rules in order to form the characters from a set of primitives through formal grammars. These methods may combine any type of topological and statistical features under some

syntactic and/or semantic rules [137]. Grammatical methods are mostly used in the post-processing stage for correcting the recognition errors [17], [152].

Graphical Methods

Character units are represented by trees, graphs, di-graphs, or attributed graphs. The character primitives (e.g., strokes) are selected by a structural approach, regardless of how the final decision is made in the recognition [82], [180].

For each class, a graph or tree is formed in the training stage to represent strokes, letters, or words. Recognition stage assigns the unknown graph to one of the classes by using a graph similarity measure.

Neural Networks (NN's)

A NN is defined as a computing architecture that consists of a massively parallel interconnection of adaptive "neural" processors. Because of its parallel nature, it can perform computations at a higher rate compared to the classical techniques. Because of its adaptive nature, it can adapt to changes in the data and learn the characteristics of input signal. An NN contains many nodes. The output from one node is fed to another one in the network and the final decision depends on the complex interaction of all nodes. NN architectures can be classified into two major groups: feed-forward and feed-back (recurrent) networks. The most common NN's used in OCR systems are the multilayer perceptron of the feed-forward networks and the Kohonen's self organizing map (SOM) of the feedback networks.

Combined Techniques

The above indicates that there are many training and recognition methods available for the OCR systems. All of the methods have their own superiorities and weaknesses. Various

strategies are developed by combining the OCR techniques previously mentioned. Hundreds of these studies can be classified either according to the algorithmic point of view, the representational point of view, or according to the architecture they use [3].

2.2.3 Post-Processing

Until now, no semantic information has been considered for usage during the stages of OCR discussed thus far. It is well known that humans read by context up to 60% of the time. While pre-processing tries to “clean” the image in a certain sense, it may remove important information, since the context information is not available at this stage. The lack of context information during the segmentation stage may cause even more severe and irreversible errors since it yields meaningless segmentation boundaries. It is clear that if the semantic information were available to a certain extent, it would contribute a lot to the accuracy of the OCR stages. However, the entire OCR problem is for determining the context of the document image. Therefore, utilization of the context information in the OCR problem creates a chicken and egg problem.

The review of the recent OCR research indicates minor improvements when only shape recognition of the character is considered. Therefore, the incorporation of context and shape information in all the stages of OCR systems is necessary for meaningful improvements in recognition rates. This is done in the post-processing stage with a feedback to the early stages of OCR.

2.3 Commercial Applications of OCR

As the cargo industry continues to grow so do the market segments that technology companies try to capitalize on. Shown below are some of the commercially available OCR systems. These products vary slightly in application and setup but all are mainly the same large array of cameras that is required to produce the kind of images needed to accomplish OCR in such a challenging environment.

The technology is currently quite cumbersome, requiring, as is shown, quite an array of tools to be implemented and more importantly, maintained over time. Finally, because of the high-technology nature of the solution, the size of installation as well as its relative newness, the price for this type of container recognition system is quite expensive.

Notice the large array of cameras, lights and other equipment needed to, among other things, compensate for the motion of the truck driving through the portal. Even with this level of technology, the truck still has to pass through at a relatively low speed.

The main vendors and their accompanying photos offering products are SAIC, High Tech Solutions, Camco and Northrop-Grumman:

Figure 2.1 SAIC's Commercial OCR System

Figure 2.2 Commercial OCR for Railroads

Figure 2.3 Camco’s OCR Solution

Figure 2.4 HTSOL's Truck Gate OCR System

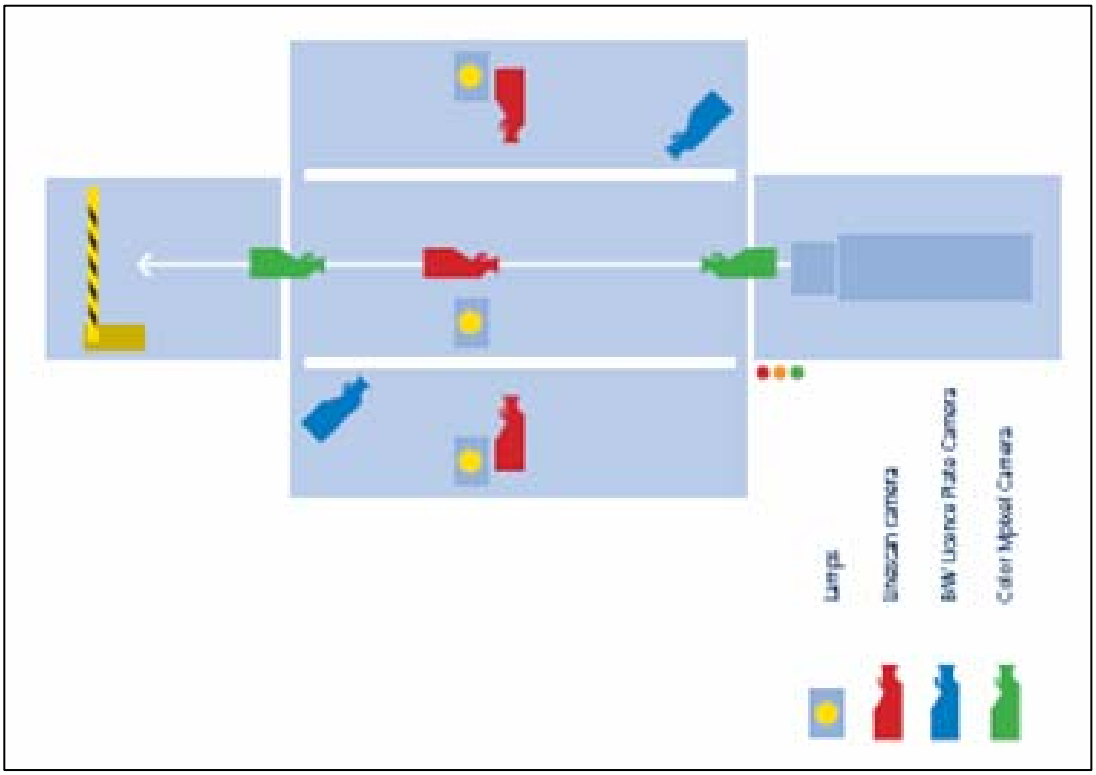

Figure 2.5 Typical Camera Configuration

CHAPTER 3

SEAPORT OPTIMIZATION

3.1 Operations Overview

This operation of an intermodal transportation terminal is one of the most complex environments in the transportation industry. The investment in even a mid-sized terminal can easily be more than one-hundred million dollars. Participating in the overall operation of most terminals are a wide variety of players: Shipping companies, terminal operators, railroads and trucking companies, brokers, shippers, forwarders and Government agencies all work together to move cargo from one place to another. Balancing the often conflicting objectives, interests and operational criteria is again a very complex task.

The goal of this section of the thesis is to give an overview of intermodal terminal operations, highlighting the different ways cargo arrives and leaves the port; as it is as these egress points that character recognition would be taking place.

The phrase “port operations” in short, refers to all the services required to move cargo inside of a port facility. The dynamics of the processes that take place inside the terminal initially depend on the nature of the cargo being handled, import or export. Once this is established other factors such as type of terminal and mode of transportation, cargo handling technology as well as financial capabilities and labor relationships of a given port. These elements are important to note but the latter are usually fairly stable once established.
In an attempt to simplify the obvious complex environment, the port can be broken into four major blocks of activity:

1) Water side (berth) interface
2) Ship operations
3) Yard operations
4) Land side interface

Scheduling could be classified as the sixth area of activity, while it affects all of the other five activities it mainly serves as coordination between the water side interface and the gate operations, which will be described shortly.

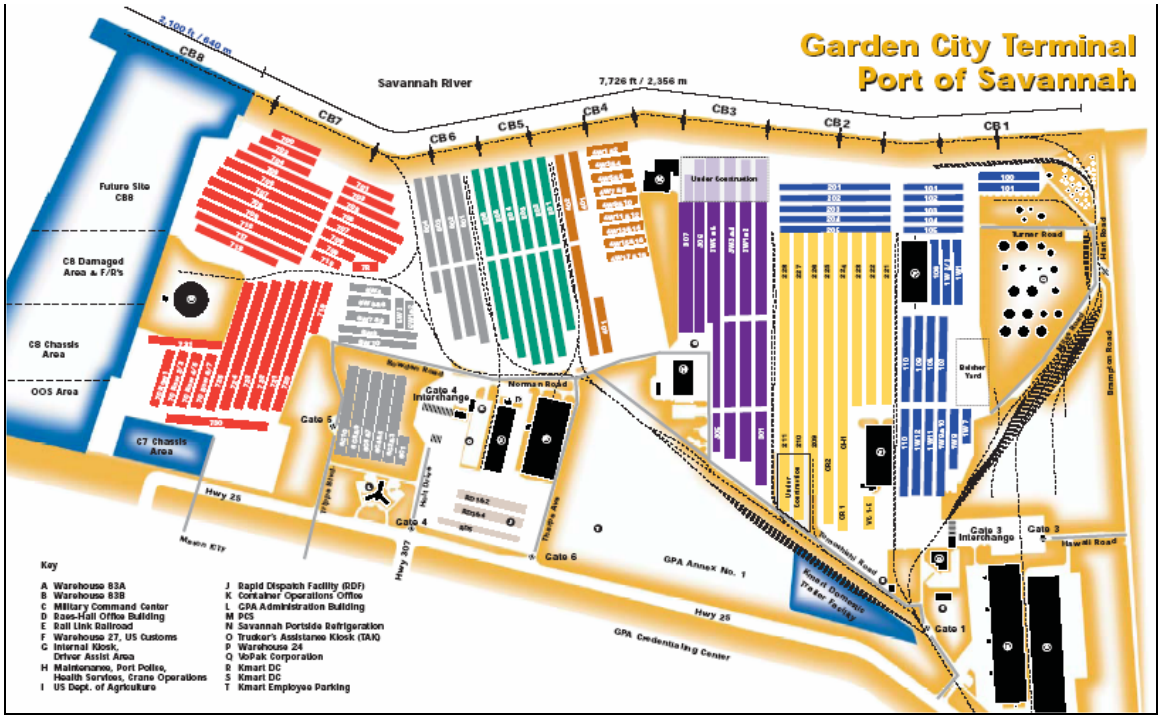

Figure 3.1 Map Included with "Truckers Guide to GPA"

Figure 3.2 Aerial Photo of the GPA Garden City Terminal

3.1.1 Water-side (Berth) Interface

The berth is the quite simply the parking place on the dock for the container ship. The terminal operator usually knows in advance when a ship is a reasonable distance away in order to do some scheduling and coordination with other arriving ships. Before this ship arrives it is required by a new federal law entitled the “24-Hour Rule”, for the manifest of the arriving ship including all container numbers and their destinations to be transmitted to the destination Port at least twenty-four hours before arrival.

Figure 3.3 Three of the Seventeen Quay Cranes at GPA

Once this information received it is loaded into the ports “Terminal Operating System” or TOS. In turn the TOS automatically provides a notification to the receiving party that the container will soon be ready to pick-up, and assign a “booking number” for each container. The TOS also begins to plan where to place all the containers once they arrive. This information will link to the “land side interface” that will be discussed in a moment and will complete the information loop.

3.1.2 Ship Operations

The ship operation involves the loading and unloading of containers and cargo onboard the vessel. This movement is done with special (very large) cranes called quay cranes. A ship normally will have two to three quay or gantry cranes “working it” to handle the task of either loading or unloading the vessel as quickly as possible. The efficiency of the quay cranes is the most important factor in determining the vessels turnaround time, and thus the cranes must work in carefully coordinated synchronization to ensure this happens.

Figure 3.4 Container Being Loaded Onto a Jockey Truck

Once the quay crane takes the container from the vessel, it must be identified for tracking in order to continue on to the rest of the operations at the port. This is the first opportunity to apply optical character recognition to capture the container identification codes. Currently the container identification code is read by a human, and keeps track with a handheld computer.

Figure 3.5 RTG Moving a Container

3.1.3 Yard Operations

The yard operation is probably the busiest of all activities in the terminal. The quay crane delivers the container from the vessel on to what is called a jockey-truck. The jockey truck is a small truck with a flatbed attached big enough to carry a container. The jockey truck driver is like the taxi driver for the container, delivering it to a predetermined area in the yard. Once there, a container handler such as a top-pick (a variation of a really large fork-lift) will lift the container from the jockey truck and place it into the container stack, again into a predetermined location. In some terminals the containers are delivered directly to the yard with the same crane as pulls it from the vessel.

Figure 3.6 Aerial Photo of the GPA Container Yard

3.1.4 Land Side Interface

The main portion of this area of activities involves gate operations. For example at the Georgia Ports Authority's Garden City terminal there are thirty-three total truck gates operated by clerks in a central processing location called the "kitchen." These clerks process paperwork and grant access to incoming and outgoing trucks. It is important to understand the current process a truck goes through upon arrival at a Port. The process listed below is extracted exactly from the Georgia Ports Trucker's Guide [64]

When a trucker arrives at the Garden City terminal these are the instructions they are given:
Delivery and Pickup Instructions:

1) Please have the following information available and ready at the time you arrive at the pedestal.
    a. Tractor license plate number
    b. Trucking company you represent
    c. Load or empty

d. Steamship line

e. Booking number, Container number, or EDO number

2) When complete, the clerk will print you a drop-off and/or a pick-up ticket. You then can proceed to the interchange lanes for the inspection process.

3) If there is an error of any kind, the clerk will print you a trouble ticket. You will then proceed to the Trucker's Assistance Kiosk (TAK). There you will find pay phones, GPA house phones and a printer.

   a. For paperwork issues, call your dispatcher. The dispatcher can fax the proper information to your attention at the gate operation office at 912-963-6926

   b. For booking related errors, use the house phone. It will automatically dial a GPA gate customer service representative (CSR). When the error is corrected, the CSR will transfer you to the ILA clerk to finish your transaction. The ILA clerk can then print you the applicable ticket. You then can proceed to the interchange lines.

   c. If you exit the gate without resolving the trouble, you must turn in the trouble ticket to the port police officer at the gatehouse.

4) At the interchange lane, give the clerk your ticket. The clerk will then perform the inspection of the unit using a Radio Data Terminal (RDT). Give the clerk any export documents, hazmat documents, etc. as required/

5) The clerk will then print two EIR tickets. One is for you to keep and the other you give to the Port police officer at the gatehouse when exiting the terminal.

6) Proceed into the container yard to the location specified on the EIR ticket.

7) The Container Handling Equipment (CHE) will be notified via the RDT of your assigned location. When complete, proceed to the wheeled area to park the chassis if necessary. You will then proceed to the next location to pick-up another unit. For faster

identification, display your tractor license plate number on the roof of your cab or on a large sign.

8) After you pick up your outbound unit, proceed to the interchange lanes for inspection. Provide the clerk with your pick-up ticket.
9) When complete, the ILA clerk will print two EIR tickets. One is for you to keep and the other you give to the Port police officer at the gatehouse as you exit the terminal.

3.2 Challenges

3.2.1 Overview

Each of the many containers flowing into terminals and ports every day is laden with a unique identification code. This code will be described in much greater detail later, but in short, it is an eleven digit code consisting of four letters followed by seven numerical digits.

Observing the operations of the Georgia Ports as well as the massive amount of containers that move around and through it, revealed there are many challenges facing a potential implementation of an OCR system in such a demanding and unique environment.

This section will provide a high-level view and outline some of the challenges specifically facing the automation and optimization of the processing of containers into and out of the port. These challenges can be roughly broken into four main areas:

1) Physical container limitations
2) Identification code characters
3) Environmental factors
4) The current process *(motion compensation application)*

3.2.2 Physical Container Limitations

Individual characters of the identification code will quite possibly (and often) will be masked by rust, dirt, peeling paint, or unexpected objects such as stickers, vandalism, and logos. Also, additional nearby characters representing container information such as weight may be misrecognized as part of the actual container identification number and thus will require time for the system to remove them, if possible. It is also important to note that containers do not have a consistent surface. Some containers are smooth surfaced, imposing possible glare, while others have vertical structural bars creating an uneven surface and thus creating shadows and slant for printed characters.

3.2.3 Identification Codes

It was determined that even though there is an ISO standard (explained in great detail later in this thesis) attempting regulation over the appearance and placement of identification codes it does actually very little to reduce the complexity introduced to a potential OCR system. This ISO standard does much more for standardizing the actual number format used vice the application thereof.

In summary, although there are generally only three styles of containers, their identification codes appear in a wide range of fonts, spacing, locations and sizes from 1”x4” to 3”x6”. The characters on the container identification number can also be aligned in one, two or three horizontal or vertical lines. The color of the containers is also almost completely random, ranging in color from dark green to light orange. This imposes yet another dimension the OCR system must deal with. The identification numbers may also appear slanted due to the orientation of the video system.

3.2.4 Environmental Factors

The illumination in a seaport environment varies according to the time of day, because of the often non-stop operations of the facility as well as changing weather. Sunshine, rain, fog, and artificial light will affect the contrast of the container images. Therefore, the recognition system must be robust to changes in illumination and colors used.

3.2.5 The Current Process (motion compensation application)

All of the aforementioned issues are very real and exist right now at any port throughout the world. As mentioned in the previous chapter on OCR, there has been much research accomplished in the area of OCR and while important to note resolving these issues is a topic of future work.

Regardless of the time of day, the condition of a container, orientation of characters or weather which all cannot be controlled and will vary drastically, the container must still go through the same process of entering the port regardless of the conditions. As previously explained in great detail, the current inbound process has the truck and container onboard stop at a pedestal to begin the interchange process. The key word to notice in the previous sentence is: stop. This stopping of the forward progress of the containers is an obvious bottleneck in the overall process of getting cargo into and/or out of the port.

At the other end of the spectrum is the desired scenario where the container does not slow down at all upon arrival, is processed on-the-fly, and thus given instantaneous entry into the port facility.

The obvious challenge here is the balance to keep the cargo moving while trying to obtain a clear image of the container identification code for accurate processing by an OCR system.

This solution is not so terribly unrealistic, and some commercial vendors have made initial attempts at implementing technologies. A sample of these solutions will be shown in an upcoming section. However, to summarize, current commercial applications compensate for this motion of the container by using a vast array of hardware such as cameras and line scanners to build a composite view of the image to achieve the level of recognition accuracy required.

3.3 Container Configuration

3.3.1 Container Identification Codes: ISO 6346:1995

All cargo containers must be uniquely identified and adhere to the regulations and specifications set forth by the International Standards Organization (ISO) in a regulation entitled “ISO 6346:1995 Freight Containers - Coding, Identification and Marking”. This section outlines the information in that standard as it explains what types of characters are to be dealt with via the OCR system. These identification numbers are also called "BIC Codes" or "ISO Alpha-codes", because they were proposed by the Bureau International des Containers (BIC)

The standard states the ISO container codes must consist of:

- Owner/operator code of four letters, the last one being U for all freight containers
- Serial number of six numerals
- One check digit, providing validation of the recording and transmission accuracy

Example: BICU 123456 5

These codes facilitate:

- International circulation and temporary admission for customs purpose
- Control of containers, manually or automatically by computerized and/or remote control systems at any stage of the transportation chain and especially in intermodal transport.

The ISO codes were developed by a consortium of industry leaders and therefore has wide acceptance as the standard for marking all containers. Currently ISO codes are recognized and accepted by:

- World Customs Organization (WCO)
- International Air Transport Association (IATA),
- International Chamber of Shipping (ICS),
- International Federation of International Removers (FIDI),
- International Union of Railways (UIC),
- International Road Transport Union (IRU)
- In 110 various countries around the world
- Approximately 1,200 owners or operators representing more than 90% of the world container fleet.

The following information is referenced directly from the ISO 6346:1995 standard. It is included to provide a clearer picture of how the identification codes should be positioned on the container. There will be more detail, including many photos in the sections to follow.

Size and color of marks

The letters and numerals of the owner code, equipment category identifier, serial number and check digit shall be not less than 100 mm (4 in) high. The letters and numerals for MAX GROSS and TARE shall be not less than 50 mm (2 in) high. All characters shall be of proportionate width and thickness, they shall be durable and in a color contrasting with that of the container.

Layout and location of marks

The requirements of this clause are particularly applicable to containers of the “closed box” type. For containers of other types, all possible practicable steps should be taken to follow the marking layout and location given for the “closed box” type of container.

Mandatory identification marks

The layout of the owner code, equipment category identifier, serial number and check digit on containers shall preferably be in one single horizontal line. Where constructional features of the container dictate otherwise, the layout may be vertical. The layout of size and type codes should, as far as practicable, be in a single horizontal line underneath the horizontal line giving the owner code, equipment category identifier and serial number and check digit. When the owner code, equipment category identifier, serial number and check digit are displayed vertically, the size and type codes should be placed adjacent to the other mandatory marks. If, on some special-purpose containers, a fully horizontal or fully vertical layout is not possible, the layout of the other mandatory identification marks shall be maintained in the horizontal or vertical manner.

On those special-purpose containers where a fully horizontal or fully vertical layout is not possible and the layout of the other mandatory identification marks is horizontal, the size and type codes should be placed beneath the other mandatory marks. The size and type codes should be used as a whole. The owner code and equipment category identifier shall be joined and shall be separated from the serial number by at least one character space. The serial number shall be separated from the check digit by one character space and the check digit shall be displayed in a box.

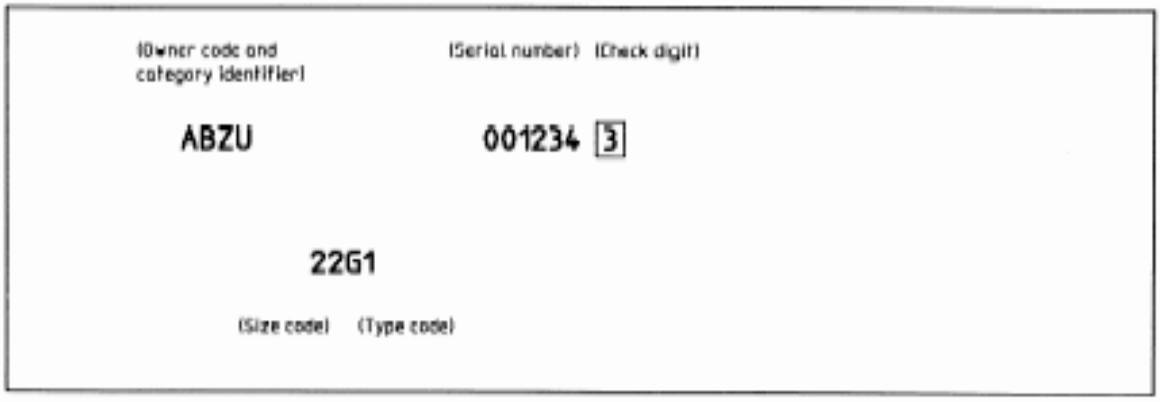

Figure 3.7 Preferred ISO Code Horizontal Layout

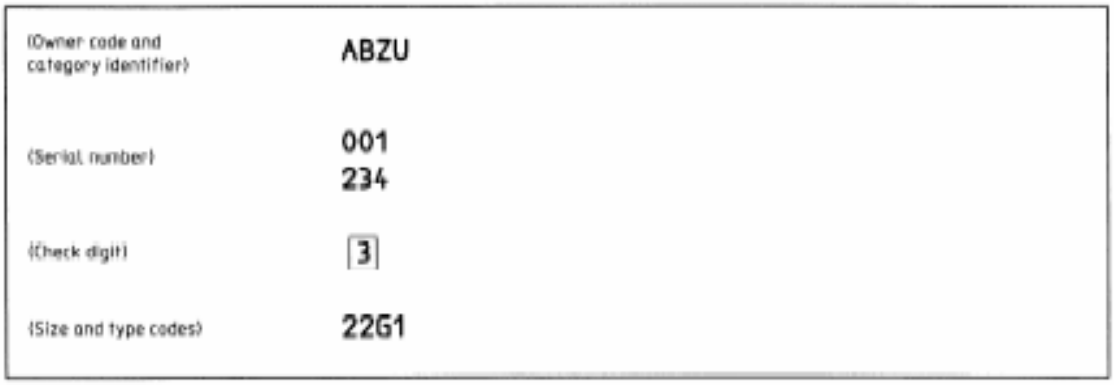

Figure 3.8 Alternate ISO Code Horizontal Grouping Layout

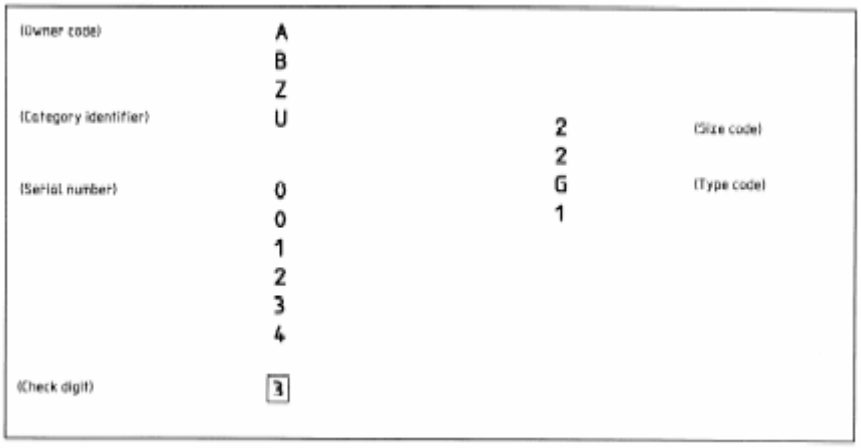

Figure 3.9 Preferred ISO Code Vertical Layout

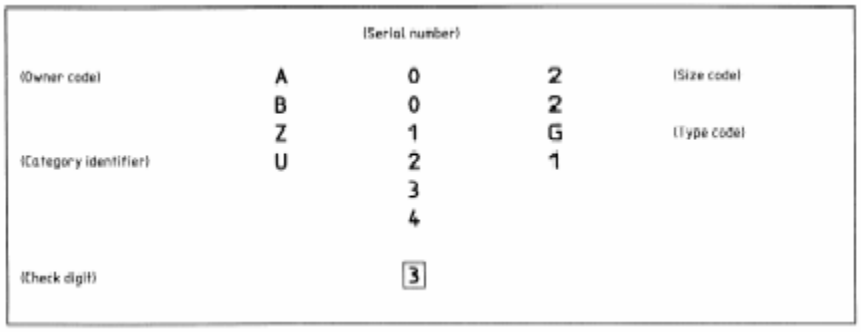

Figure 3.10 Alternate (Multiple Column) Vertical Layout

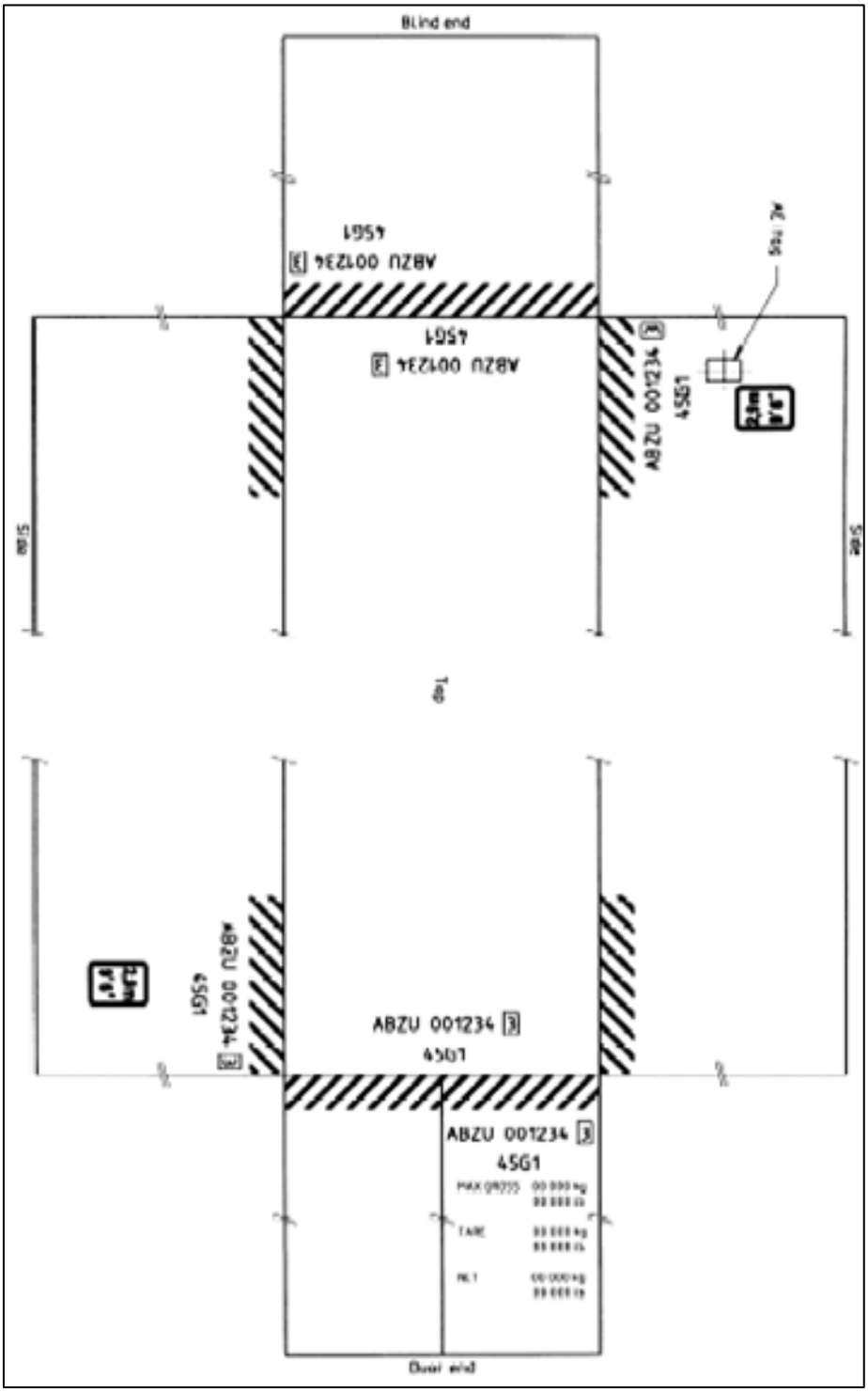

Figure 3.11 Location of Mandatory and Optional ISO Markings

Table 3.1 ISO Container Identification Code Example Photographs

CHAPTER 4

MOTION ESTIMATION AND COMPENSATION

4.1 Theory of Motion Estimation and Compensation

What is actually estimated when comparing the same point in an image in two consecutive frames is displacement vice motion. However, because there is a strong correlation between motion and the displacements found in an image sequence, the terms are used interchangeably. Motion is a prominent source of temporal variations in image sequences. In order to model and thus compute motion, an understanding is needed as to how images are formed. In most instances motion in image sequences acquired by a video camera is induced by movement sources: objects in the scene and camera motion. Thus, camera parameters, such as its 3-D motion (rotation and translation) or focal length, are important in motion modeling. If these parameters are known precisely, only object motion needs to be recovered. It should be noted from the previous description of a seaport that the camera location is known and is stationary throughout the capturing sequence. However, this scenario is rather rare and both object and camera motion usually needs to be computed. The 3-D motion of objects and camera induces 2-D motion on the image plane via a suitable projection system. It is this 2-D motion, also called apparent motion or optical flow that needs to be recovered from intensity and color information of a video sequence.

An image in a sequence is a single two-dimensional projection from a three-dimensional world. The task of estimating the real world motion is known as motion estimation and faces a problem of under-constraint because of the nature of the 2D to 3D transformation requiring the inversion of a many to one mapping process. This means that when the motion of an

image is estimated we need to make some assumptions concerning the images statistics. Once the motion is estimated, we can compensate for it through the appropriately named process of motion compensation. One example of the under-constrained property of motion estimation process is called the aperture problem, and is shown below.

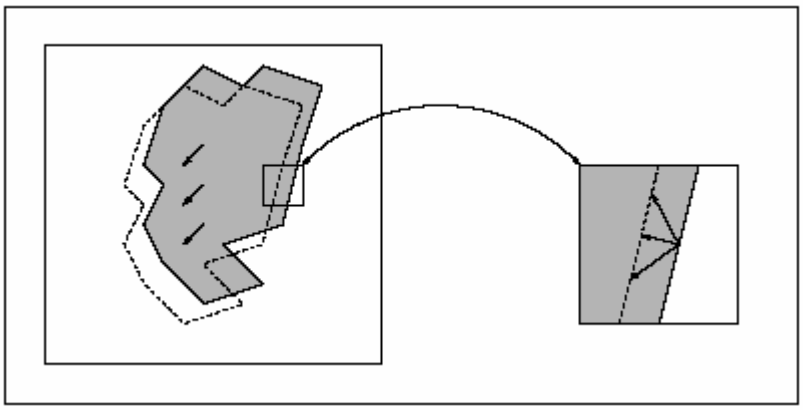

Figure 4.1 Illustration of aperture problem

This image shows that if the aperture window does not reveal enough information about the boundaries of the image, then the motion about the image cannot be accurately estimated. The majority of motion compensation algorithms assume an image based on the relationship:

$$I_n(x) = I_m(F(x)) \tag{4.1}$$

Where $I_n(x)$ is the grey level value at pixel position x in image n of a sequence. If $m = n-1$ the algorithm performs backwards matching where the current frame is matched to the previous in a sequence and conversely for the $m = n+1$ scenario. The model stated above implies that the content of the current image is correlated to the adjacent image content in a sequence by the function $F(\,)$, this function is the motion model used by a motion compensation algorithm. The other assumption this model makes is that in general a sequence of images consists of the same objects that vary relatively slowly over time.

4.2 Motion Estimation and Compensation Techniques

This section investigates a select group of main-stream approaches to motion compensation. The first section will discuss translation motion models including: gradient based, correspondence matching and hierarchical methods followed by higher order compensation models including affine, bilinear and perspective models**.** Finally, alternative methods such as motion compensated temporal filtering *(MCTF)* are briefly discussed.

4.2.1 Translational Motion Compensation Methods

This simple model accounts only for the translational motion in an image plane, given by:

$$\begin{aligned} x_{n-1} &= T_x(x_n, y_n) = x_n + c \\ y_{n-1} &= T_y(x_n, y_n) = y_n + f \end{aligned} \tag{4.2}$$

It has two parameters c and f which denote the translations along x and y respectively. The block translational model has been employed in several international standards, such as H.261, H.263, MPEG-1, MPEG-2 and MPEG-4. Wide availability of fast block-matching solutions in hardware and software makes this model a popular choice. However, in most practical applications, it fails to capture the actual motion, thus low bit-rate compressed video often includes unwanted block artifacts.

Gradient Estimation Methods

Gradient-based estimation methods are fundamentally based on the optical flow constraint equation. An optical flow field is the distribution of noticed velocities of moving intensity patterns in an image. These velocities are estimated over small time durations and are plotted to create the optical flow field representing the estimated 3D motion of objects inside a 2D image sequence. To compute this, the optical flow constraint equation is utilized:

$$I(x+\Delta x, y+\Delta y, t+\Delta t) = I(x, y, t) \tag{4.3}$$

Where the image intensity at location (x, y) and time t which is represented as a scalar function of three continuous variables $I(x, y, t)$ is equal to that intensity at time $t+\Delta t$ and location $(x+\Delta x, y+\Delta y)$ where $(\Delta x, \Delta y)$ represents the optical flow. If a Taylor series expansion process is followed, the left hand side of the optical flow constraint equation becomes:

$$\begin{aligned} I(x+\Delta x, y+\Delta y, t+\Delta t) &= I(x, y, t) + \frac{\partial I}{\partial x}\Delta x + \frac{\partial I}{\partial y}\Delta y + \frac{\partial I}{\partial t}\Delta t + \varepsilon \\ &= I(x, y, t) + I_x \Delta x + I_y \Delta y + I_t \Delta t + \varepsilon \end{aligned} \tag{4.4}$$

Where $I_x, I_y \text{ and } I_t$ are partial derivatives and ε is the highest order terms of the Taylor series which are of insignificant values. Substituting 2.2 into 2.3 and setting $u = \frac{\triangle x}{\triangle t}$ and $u = \frac{\triangle y}{\triangle t}$ we obtain:

$$I_x \frac{dx}{dt} + I_y \frac{dy}{dt} + I_t = I(x, y, t) + I_x u + I_y v + I_t = 0 \tag{4.5}$$

Where the velocity component in the x and y directions are given by u and v and represent the optical flow at point (x, y). The partial derivatives $I_x, I_y \text{ and } I_t$ can be estimated from the image. It is important to note that equation (4.5) is unconstrained in a direction perpendicular to the spatial gradient of $I(x, y)$. This mathematically describes the aperture problem mentioned previously. If additional constraints are applied equation (4.5) can be uniquely solved as a motion vector field.

Local Smoothness Constraint

One common constraint is the "local smoothness constraint" which assumes that neighboring points in an image have similar velocities and thus the optical flow being relatively smooth in a localized area. Horn and Schunck [38] used this constraint by minimizing the error in the optical flow through the following expression:

$$\begin{aligned} E^2(x,y) &= (I_x u + I_y v + I_t)^2 + \lambda\left(\left(\frac{\partial u}{\partial x}\right)^2 + \left(\frac{\partial u}{\partial y}\right)^2 + \left(\frac{\partial v}{\partial x}\right)^2 + \left(\frac{\partial v}{\partial y}\right)^2\right) \\ &= (I_x u + I_y v + I_t)^2 + \lambda(u_x^2 + u_y^2 + v_x^2 + v_y^2) \end{aligned} \tag{4.6}$$

Where the first three terms on the right hand side of the equation are the same as in equation (4.5), λ is the Lagrange multiplier and the squared partial derivatives represent the smoothness criterion. To solve for u and v requires the minimization of the following function over the entire image I.

$$\iint_{x,y\in I} E^2(x,y)\,dx\,dy \tag{4.7}$$

The solution to the above equation requires calculus of various methods and yields an iterative solution using pairs of consecutive images. This procedure generates a velocity vector for each pixel in the image and thus creates a "dense vector field." As its name implies, the local smoothness constraint approach is only effective in the main field of an image and fails in areas of discontinuity such as at the edge of a moving object. This approach also requires the values of $\triangle x$, $\triangle y$ and $\triangle t$ to be very small. If the motion between the images is too vast or if the sampling rate is too small this approach will not prove effective.

Locally Constant Constraint

Another constraint approach entitled "locally constant constraint" assumes that optical flow in a localized area is constant as opposed to just smooth. Cafforio and Rocca [18] used this

approach to constraining a gradient based translational motion model. Cafforio and Rocca segmented the images into a sequence of moving foregrounds and stationary backgrounds and could then determine the motion of the foreground. This was done by using a linear regression method to solve the following equation:

$$R = \iint_{x,y \in F} (I_x u + I_y v + I_t)^2 \, dxdy \tag{4.8}$$

Where the limits of evaluation for the integral are over the foreground area of the image. However, just as in the "locally smooth" approach the assumption must be made that the overall motion is small for the approach to be useful.

Pel-Recursive Iterative Approach

Netravali and Robbins in the late 1970's devised a pel-recursive, iterative approach removing this small-motion assumption needed for constraint [64, 65, 66]. Representing the initially shown motion model function $F(x)$ of (4.1) as $x' + d'$ with $x' = (x, y)^T$ and pixel displacement $d' = (d_x, d_y)^T$ the new equation (4.9) below is formed:

$$I_n(x') = I_m(x' + d') \tag{4.9}$$

The Taylor series expansion of this equation then becomes:

$$I_n(x') = I_m(x') + d'^T \nabla I_m(x') + \varepsilon_m(x') \tag{4.10}$$

Where $\nabla I_m(x') = (\frac{\partial}{\partial x} I_m(x') \frac{\partial}{\partial y} I_m(x'))^T$ is the partial derivative at position x' in image m.

The definition of the difference between adjacent frames as

$FD(x') = I_n(x') - I_m(x')$ the above equation becomes:

$$FD(x') = d'^T \nabla I_m(x') + \varepsilon_m(x') \approx d'^T \nabla I_m(x') \tag{4.11}$$

This is the discrete equivalent to equation (4.5) with the partial differential in time is approximated by a frame difference term of $-FD(x')$. It is assumed that pixels within a localized area of an image have the same displacement thus allowing a vector equation to be setup.

$$F'D' = \Phi d' \tag{4.12}$$

Where a solution for d' can be found by using a pseudo-inverse approach:

$$d' = [\Phi^T \Phi]^{-1} \Phi^T F'D' \tag{4.13}$$

This iterative approach linearizes the intensity function around an initial pixel displacement estimate of d'.

The pel-recursive portion of the approach updates the displacement of each pixel on a per pixel basis and can therefore theoretically model varying motion within a moving object as long as the recursion recourses fairly quickly. This is an important discovery because if the pel-recursion is done in such a way that a current pixel's displacement value d' is based on pixels previously decoded (above or to the left of current location) a compression algorithm could be created so that no motion vectors need be transmitted.

Other Constraint Methods

There are a few other methods that have had significant impact on the constraint of the gradient-based translational model. Biemond *et al.* [10] proposed an improved iterative process that takes into account the higher order terms of the Taylor expansion which were previously not valued. Martinez [56] implemented an entirely different approach demonstrating that an eigen-value ratio is dependent on the relative magnitudes of average special gradients in the two directions indicated by the corresponding eigen-vectors.

Martinez also demonstrated that the motion estimate is most accurate in the direction of the maximum image gradient

Correspondence Estimation Methods: Block Matching

Correspondence methods rely on identifying elements in different images that can be used to represent the same object at different times in the sequence. The most widely deployed approach to correspondence estimation is the block-matching approach. This approach is based on the concept that pixels in a localized region have the same purely translational estimated motion between frames in a sequence. Jain and Jain [46] in 1981 were the pioneers to this approach and have thus been followed by many others such as: Boyce [14], Ghanbari [34] and Zaccarain *et al.* [102] who have all proposed various improvements on the work of Jain and Jain. The basic premise of the block matching approach is shown here:

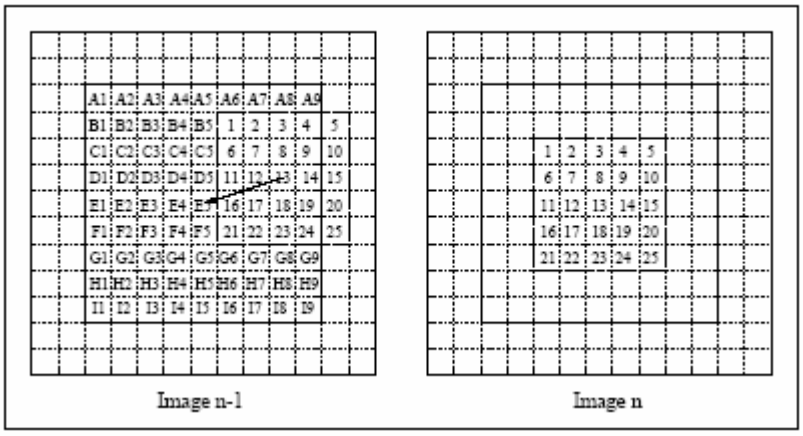

Figure 4.2 Block matching approach

The current image is split into small blocks, rectangular in shape and a strategy of searching or matching is started. Each small block is translated over a defined search area and compared to the un-translated adjacent image area in the sequence. This matching process is done using either the mean square error (MSE), mean absolute error (MAE) or a cross-

correlation function (CCF). Both the MSE and MAE methods look for the distortion function to be minimized where the CCF looks for the correlation to reach a maximum value to indicate a best match.

Mean Square Error (MSE) Approach

At every pixel location in the defined search space the comparative error value is determined over the range of P pixels:

$$MSE = \sum_{x,x'=1}^{P} (I_m(x') - I_n(x))^2 \tag{4.14}$$

Here $I_n(x)$ represents the pixel intensity of the un-translated pixel location x in image n and $I_m(x')$ is the intensity of the translated pixel location x' in adjacent image m. After computing the MSE at all pixel locations in the search space, the pixel location having the overall lowest error value is designated as the best match for that space. Now that the new best matching location is determined, a motion vector is derived representing a vector between the un-translated positions to the best match position.

If a defined search area is of size $\pm k$ pixels, and it is known there are P pixels in the block and there is one comparison made at each pixel position, thus delivering a "full motion search" the number of computation operations needed to perform this full search is $P(2k+1)^2$ operations per block. There has been much research accomplished to reduce this computational load for the MSE approach. The first idea was once again provided by Jain and Jain [46] and suggests a reduction in the number of locations at which the MSE computation is done by using a two-dimensional directed search approach. The approach exploits the fact that the MSE function is concave over the defined search space. The directed search method is an iterative process evaluating the error at five different positions

per iteration forming a cross. If the minimum value in an iteration is determined to be at the center of the cross, the dimension of the cross is reduced by two. This continues until the dimension of the cross equals one, at which point a three by three pixel search is conducted. Subsequent research was extended from Jain and Jain's theme by Ghanbari [34] in 1990 when he suggested a variation that reduces the computational load even further still. Recalling that a typical search covers $\pm k$ pixels the reduction in complexity from the directed search method generally reduces the number of search locations from order $O(k^2)$ to $O(\log(k))$.

Another reduction in computation load stemmed from the work of Zaccarin [102] who suggested the use of a sub-sampling approach. It is important to note that while the reduction of computational load is in general a good trend to follow, it comes at a cost of degraded accuracy because of the reduction of the search size.

Successive Elimination Algorithm (SEA) and the MEA Approach

Along these lines, one approach worth mentioning is that of Li and Salari [54] in 1995 which describes a fast search method entitled the Successive Elimination Algorithm (SEA) and claims to guarantee the same accuracy as the full search block matching method at a reduced computational overhead cost. In this algorithm, the pixels within a block are arranged into a single P length column vector. The notation of the process follows as such: the ith block from an image n is represented as g_n^i; and from that ith block at a given search location k in the adjacent frame m is represented as $g_{m,k}^i$. Therefore the error computation is based on the mean absolute error (MAE) and is defined as:

$$MAE_k^i = \left\| g_n^i - g_{m,k}^i \right\|_1 \tag{4.15}$$

The location of the search position k that contains the minimum MAE value is used to derive an optimum motion vector. According to the triangular inequality, for a vector of length P the following relationship applies:

$$|\sum_{j=1}^{P}\alpha_j| \le \sum_{j=1}^{P}|\alpha_j| \tag{4.16}$$

With a column vector described:

$$S_P = [1,1,...,1]^T \tag{4.17}$$

equation (4.16) above can be written as:

$$|s_P^T\alpha| \le \|\alpha\|_1 \tag{4.18}$$

Then setting $\alpha = g_n^i - g_{m,k}^i$ and assuming the intensity at a location vector x is grater than or equal to zero yields:

$$\left|\|g_n^i\|_1 - \|g_{m,k}^i\|_1\right| \le \|g_n^i - g_{m,k}^i\|_1 \tag{4.19}$$

This last formula (4.19) gives the lower bound to the value the MAE can be at a search location k. Having evaluated an initial MAE for block i at location $k=0$, and setting $MAE_{current} = MAE_0^i$, every subsequent search location k can be compared to see if it satisfies the following condition:

$$\left|\|g_n^i\|_1 - \|g_{m,k}^i\|_1\right| \le MAE_{current} \tag{4.20}$$

If so, then a new MAE value equal to $MAE_{current} = \min(MAE_k^i, MAE_{current})$ is determined and stored. Once a full search is complete, the resulting $MAE_{current}$ represents the location for the optimal motion vector. Even though the full search is still being conducted the computational load is much less because:

- The sum norms values $\left\| g_n^i \right\|_1$ need to be determined once per block
- The MAE only needs to be actually computed when (4.20) is satisfied
- The sum norms $\left\| g_{m,k}^i \right\|_1$ can be efficiently updated from adjacent search positions

Now that each block in an image n has a motion vector associated with it, a reconstructed image can be created using them and an adjacent image m.

There has been quite a bit of research devoted to improving upon the very simple block-matching algorithm. Two popular suggestions were the implementation of sub-pixel accuracy and block segmentation.

Sub-pixel Accuracy Methods

In video sequences, all pixels are assumed to lie at integer positions; however, objects do not necessarily move in integer-pixel displacements. Many motion compensation algorithms employ motion vectors with resolution increased to sub-pixel accuracy. The process involves using a half or quarter pixel size step to perform the search for the minimum MAE value. This increased accuracy typically yields a better prediction of motion at the cost of a higher motion-vector rate overhead. In many cases, however, a net gain in rate-distortion performance is achieved. In sub-pixel accurate methods, motion vectors often point to positions in between integer pixels, thereby requiring some form of interpolation to determine sub-pixel values. There are a few different methods available such as spline, quadratic, sinc and the simplest and most common being the bilinear interpolation method.

Extensive literature has shown that a simple bilinear interpolation achieves good performance for half-pixel accuracy in traditional block matching motion compensation

methods. However, to further increase accuracy beyond half-pixel in these systems, bilinear interpolation does not typically improve performance since the additional motion-vector overhead usually outweighs the potential reduction in distortion [26]. As a result, more sophisticated interpolation filters replace bilinear interpolation in most modern systems; for example, quarter-pixel accuracy is achieved in the block matching methods of MPEG-4 using an 8-tap filter [26].

Block-Segmentation Methods

The second popular improvement is that of block segmentation. The basis for block segmentation is that if a search block happens to overlap multiple objects then the normal process of determining the motion vector cannot possibly work as it would require two motion vectors for that one location. The block segmentation process simply says in that overlapping situation split the block and derives the motion vectors separately [20, 31, 72]. Some techniques suggest splitting the block diagonally, horizontally or vertically; but the most popular strategy for splitting is the quad-tree method where the block is divided into four smaller sub-blocks. There are other methods that deviate from the symmetric, rectangular shape but are much more cumbersome to implement in actual hardware.

Hierarchical Approach

The multi-resolution or Hierarchical approach was first introduced in the 1980's by Bierling [11] as a solution to the problem of tracking large motion at low computational complexity. The theory is that every pair of adjacent images in a given sequence generates a corresponding set of low-pass filtered, sub-sampled images. Each of these copied images is scaled by a factor of one-quarter and when compared to the previous image begins to create a pyramid type structure, shown below. It is for this reason that the multi-resolution approach is also commonly referred to as the hierarchical approach.

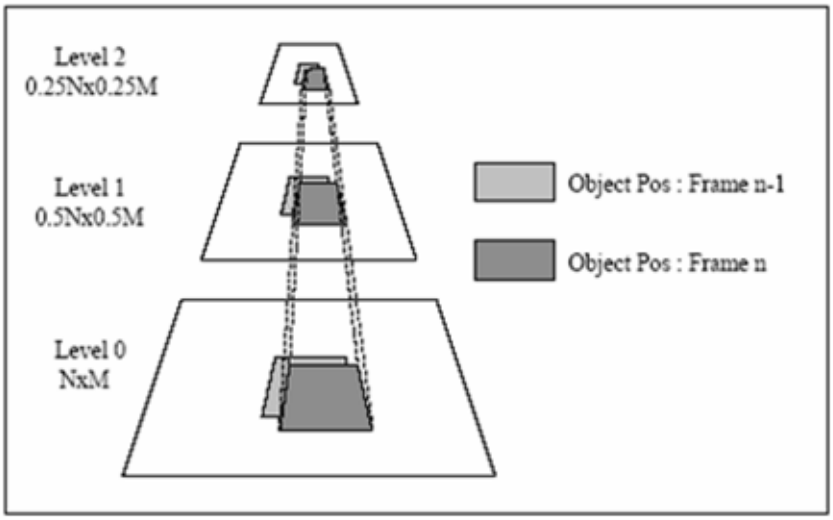

Figure 4.3 Hierarchical block matching scheme

This approach begins at the coarsest image scale located at the top of the pyramid shown above, where a set of motion vectors are derived using a typical block matching process. These vectors are then scaled by a factor of two and sent down the next finer level below. The idea is to try to generate an initial estimate of the motion at the coarsest level thus reducing the computational overhead.

If the block in use is of size P at full scale and the hierarchical pyramid has L levels, and the search needs to cover $\pm k$ pixels, the number of operations per can be found by:

$$\sum_{i=1}^{L-1} \frac{P}{2^{2i}} (2k+1)^2 \tag{4.21}$$

This procedure can reduce the number of operations, compared to a full scale search by as much as a factor of ten. Bierling noted that when a very textured moved across a less textured background, a situation he called the “vector halo” effect occurred. This included blocks in the less textured background seemed to follow the motion of the more textured

object. Bierling proposed the inclusion of a test at the lowest level to flag all blocks thought to be stationary. Thus, only blocks that are un-flagged could be assigned motion vectors.

4.2.2 Higher-Order Motion Compensation Methods

After the foundation of translational models and block matching had begun, researchers began to look into other innovative ways to improve upon that basic system. Most of the research was based on meshes of connected yet non-overlapping triangles or quadrilaterals, some examples of which are shown below. This connected property of the mesh allows the compensation algorithm to avoid the blocking artifacts present in the translational models at low bit-rates. There are many types of mesh arrangements that have been used over the years and include mainly, as mentioned before, triangles and quadrilaterals in either a regular or irregular composition. Some examples can be seen below:

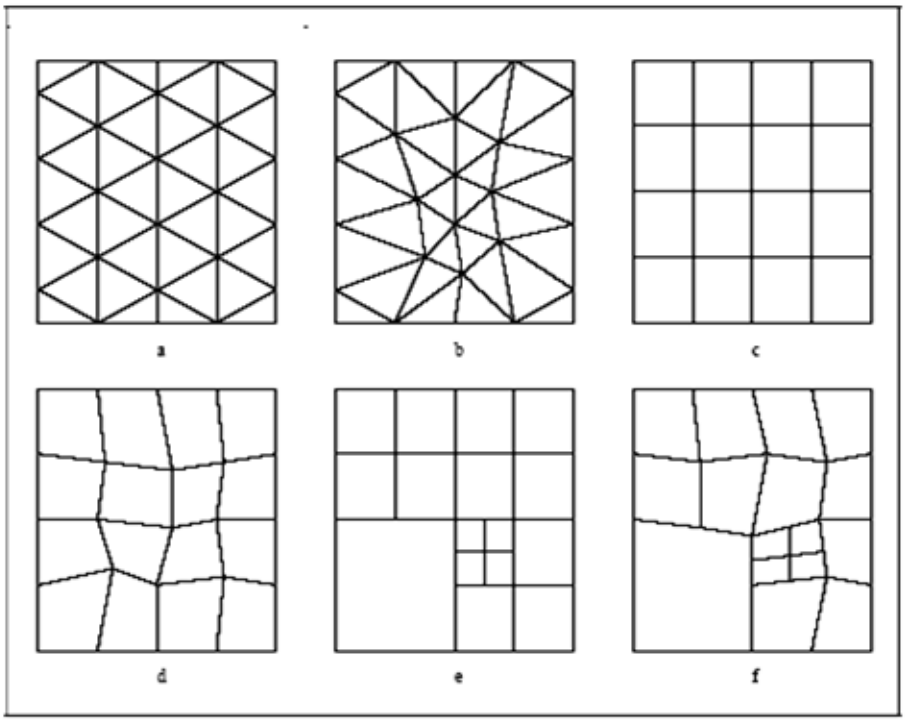

Figure 4.4 Mesh designs used in higher order motion model schemes

In general, irregular structures allow for more accurate motion compensation but also are more computationally intense because the location of the irregularly placed vertices needs to be sent as additional information in order to be useful. This additional property gives irregular meshes the additional title of content-based or active meshes. The other differences and advantages of one arrangement over another mainly depend on the type of motion model operating the mesh pattern.

In addition to the regular and irregular composition of triangular or quadrilateral meshes there also exists a “knowledge based” mesh where the type of image sequence is known before hand. For example, a constant focus research is the enhancement of the encoding of “head and shoulder” scenes like in a television news broadcast. In the 1980’s, for this specific type of image sequence Rydfalk [85] developed the 3D CANDIDE model.

Affine Transformation

The affine motion model uses a mesh of triangles allowing six-degrees of freedom, two located at each vertex. If we have the j th triangle in which a pixel is mapped, we obtain the affine parameters $a_{j1},...,a_{j6}$ from $(x,y)\ to\ (x',y')$ as follows:

$$\begin{aligned} x' &= a_{j1}x + a_{j2}y + a_{j3} \\ y' &= a_{j4}x + a_{j5}y + a_{j6} \end{aligned} \tag{4.22}$$

One important property of the overall affine transform is that parallel lines retain that property as do straight lines. However, neither angles nor lengths are maintained by the transformation. Given a motion vector at each vertex of the triangle all six of the corresponding affine transform terms can be uniquely determined. This transform model is the most common and versatile model because it is the simplest in form as well as low in computational complexity compared to other transformation alternatives.

Bilinear Transformation

If a quadrilateral structure is utilized there will be eight degrees of freedom and either the bilinear or the perspective transformations can be used. If we have the j th quadrilateral in which a pixel is mapped, we obtain the bilinear parameters using $a_{j1},...,a_{j8}$ in the following format:

$$\begin{aligned} x' &= a_{j1}xy + a_{j2}x + a_{j3}y + a_{j4} \\ y' &= a_{j5}xy + a_{j6}x + a_{j7}y + a_{j8} \end{aligned} \tag{4.23}$$

If the quadrilaterals are initially rectangular or otherwise regular in shape, it is much less computationally intensive to compute the above eight parameters. Unlike the affine transform, parallel lines are not guaranteed to stay that way, however similar to the affine transform straight lines do. Also like the affine transform, neither angles nor lengths are maintained.

Perspective Transformation

The perspective transformation of the pixel of the j th quadrilateral with parameters $a_{j1},...,a_{j8}$ is accomplished in the following manner:

$$x' = \frac{a_{j1}x + a_{j2}y + a_{j3}}{a_{j7}x + a_{j8}y + 1}$$
$$y' = \frac{a_{j4}x + a_{j5}y + a_{j6}}{a_{j7}x + a_{j8}y + 1} \quad (4.24)$$

This process is by far the most computationally intensive of the three investigated thus far. Therefore, the Perspective Transformation model is usually reserved for use with computer graphics due to its ability to model motion of rigid planar objects undergoing perspective projection.

Alternative Higher Order Transformations

The main theme to all the previously discussed mesh structures is the fact they are all constrained by constant connectivity. This is an intentional result and a good thing, as it reduces blocking artifacts seen in various translational models. However, In order to increase the number of parameters used and thus increase the order of the model, that connectivity constraint must be waived or at a minimum reduced.

One popular example of this higher order mesh is suggested by Papadopoulos [76] which increases the parameters to twelve represented by $a_{j1},...,a_{j12}$, and does not maintain the connectivity to adjacent blocks. The process is carried out as such:

$$x' = a_{j1}x^2 + a_{j2}x + a_{j3}xy + a_{j4}y^2 + a_{j5}y + a_{j6}$$
$$y' = a_{j7}x^2 + a_{j8}x + a_{j9}xy + a_{j10}y^2 + a_{j11}y + a_{j12} \quad (4.25)$$

Once again, the affine transform has the least computation overhead regarding matching as well as reconstruction. Also, because of the lower dimensionality, it is more constrained in nature and thus better conditioned in terms of its matching ability.

The next section reviews motion estimation methods that employ various mesh structures discussed above. These methods include similar approaches as discussed before such as gradient-based and correspondence matching methods.

Gradient Based Methods

In 1994 Nakaya and Harashima [62] suggested two strategies for the warping of a triangular mesh. These suggestions were a gradient based and correspondence method. Both of the two methods used a two stage approach to derive the location of the motion field vectors. The first stage was the same for both methods in which an energy minimization process was implemented. This energy minimization process was a combination of the block matching function, previously discussed in great detail, and a mesh shape preserving method. In their suggestion they based the block matching portion on a very specific seven-by-seven block and a ± 15 pixel searchable area. The center-point of each block is located at the vertex of a triangle, and the shape preserving function assumes the edges of the triangles are like springs and defines a shape preserving energy (SPE) term as such:

$$SPE_{ij} = \kappa(r'_{ij} - r_{ij})^2 \tag{4.26}$$

Where r_{ij} is the initial length of the edge from the central vertex i to the adjacent vertex j and r'_{ij} is the final length after block matching. The spring constant is represented by κ.

Given a grid-point i, the overall energy function to minimize at that point is found by:

$$E_i = \sum_{x,x'=1}^{7\text{x}7} \left| I_n(x) - I_{n-1}(x') \right| + \kappa \sum_{j=1}^{6} (r'_{ij} - r_{ij})^2 \tag{4.27}$$

Here the error is measured by comparing mean absolute difference vice mean square error as we saw before in translational models. $I_n(x)$ represents the intensity value at the un-translated position x in image n and $I_{n-1}(x')$ is the resulting intensity value in the translated pixel location x' in the previous image $n-1$.

Nakaya's process depends on the assumption that a good estimate of the actual motion was determined in the first stage. Based on this assumption, Nakaya defines a new vector estimate which is a combination of the current estimate and by applying the refinement vector:

$$\begin{pmatrix} \hat{u} \\ \hat{v} \end{pmatrix} = \begin{pmatrix} \hat{u}^c \\ \hat{v}^c \end{pmatrix} + \begin{pmatrix} \hat{u}^r \\ \hat{v}^r \end{pmatrix} \tag{4.28}$$

As in the translational models, the majority of the gradient based affine methods are based upon the same optical flow constraints.

Three years later Altunbasak (GO Yellow Jackets!) and Tekalp suggested four methods for mesh based motion compensation [5]; two were called "patch-based" and two "node-based". The patch-based mesh processes only considered one triangle at a time when determining updates for the vertices. This was a very different concept to the current models that consider all the triangles associated with the common vertex. The node-based gradient methods suggested by Altunbasak not only update the central vertex but all those surrounding it as well.

Huang *et al.* [41] suggest a hierarchical approach to warping motion estimation. This suggestion follows a quad-tree segmentation approach of the image which creates previously described as a regular-quadrilateral hierarchical mesh. Three other important

areas Huang highlighted were his classification of grid points; inclusion of course motion estimation stage; and a fast hierarchical grid interpolation method. The quad-tree segmentation method recursively decomposes rectangular image regions into smaller quadrants. This is repeated until the individual region meets a constraint based on the uniformity of that region.

Other research includes Papadopoulos and Clarkson [76] in which second order geometric transformations were the motion model utilized. In this suggestion, a connected mesh was replaced by regular blocks of 32x32 pixels in size, which were matched to the adjacent frame via a second order transform like (4.25). This scheme was found to be excessively computationally intensive.

Wang and lee *et al.* [97,99] suggested a method that utilized what is called a finite element mesh (FEM). The key to this approach is the mathematical tool of mapping between an individual element and an identified "master" element which has an easily computable shape function. This coding method creates a scene adaptive approach that creates nodes that are "locked" to features in the image and thus provides very accurate tracking of objects.

Correspondence Methods

In 1991, Sullivan and Baker [93] suggested what was called "Control Grid Interpolation" or CGI and is based on a regular grid of quadrilateral blocks. In their approach a backwards iterative matching procedure is used to optimize the positions of the grid to derive the location of the motion vectors. The vector update at each node is performed by a translational block matching method with the node at the center of the translating block. A bilinear interpolation process is used to reconstruct the frame which updates all the pixels inside the four quadrilaterals touching the current node. In the work accomplished by

Sullivan, it was stated that with this method convergence was normally obtained in five to ten iterations. Two years later Nieweglowski and Haavisto suggested [67,68] an approach that stemmed from the control grid interpolation methods of Sullivan and Baker. Their suggestion had three main contributions: an enhanced, two stage search strategy; the use of an adaptive mesh generation approach for forward matching; an attempt to model discontinuities in the motion field by adaptively changing the interpolation method.

The attempt to model discontinuities is achieved by analyzing both the vertical and horizontal components of the four motion vectors tied to the four corners of the block. If the variance of either of these components is greater than a predetermined threshold, a "nearest neighbor" approach is used to reconstruct the block rather than the usual bilinear interpolation approach. This effectively means that a translational motion model is being applied along all edge boundaries of the image. The modeling of discontinuities ultimately results in a lower prediction error while at the same time reintroducing blocking artifacts because of the translational nature of parts of the process.

In 1994 Nakaya and Harashima [62] released a well respected paper that attempted to mathematically quantify how well different warping methods work. Affine motion compensation methods (AFMC) and bilinear motion compensation methods (BLMC) were compared:

1. AFMC and BLMC methods perform better than block matching methods even when the actual motion vector field is discontinuous
2. The optimal regular polygon shape for AFMC methods is a grid constructed from equilateral triangles.
3. The optimal regular polygon shape for BLMC methods is a grid constructed from equilateral squares.

4. AFMC and BLMC methods have nearly the same performance when the patch shape is optimized but as the number of grid points is decreased the degradation in performance of AFMC methods is less than that of BLMC methods

From their experiments they found that the correspondence based method outperforms the gradient approach but that this increased performance is outweighed by the resulting reduction in complexity achieved when using the gradient based methods.

4.2.3 Motion Compensated Temporal Filtering (MCTF)

The concept of coding video by grouping several frames together into a 3-D volume and then employing transforms in the spatial and temporal directions has been explored for the past several decades. However, temporal transforms for video pose a unique problem that causes 3-D video coding to be different from the coding of other 3-D data types, such as volumetric medical imagery and multi-spectral remotely-sensed imagery. Specifically, motion of an object in time can produce high-frequency coefficients in the temporal transform, even if the object does not vary in shape or gray-level intensity over the temporal interval.

Consequently, researchers have sought temporal transforms that track object trajectories because they temporally de-correlate object pixels regardless of the motion they undergo. These temporal transforms are called motion-compensated temporal filtering (MCTF). One popular group of algorithms in the area of MCTF involves traditional block-based motion compensation (previously discussed in detail) with temporal filtering [8,11,16,17]. These methods are referred to as block-displacement methods and are briefly explained below.

Block-Displacement Methods

First proposed by Ohm [8, 16], then further developed by Woods et al. [9, 10], the block-displacement methods have a long history in the area of 3D sub-band video coding, naturally providing the spatial-resolution as well as frame-rate scalability which are increasingly expected of modern multimedia applications. Although a popular approach to MCTF, block-displacement techniques have traditionally encountered a number of drawbacks. First, the rigid block-motion model fails to capture all aspects of the motion field, leaving significant numbers of pixels unconnected between frames; these unconnected pixels are coded separately to the detriment of coding efficiency. Second, it is difficult to achieve sub-pixel accuracy in the MCTF.

CHAPTER 5

RESULTS

5.1 PROCEDURE

5.1.1 Overview

As discussed in great detail in the previous sections, there are many possibilities when applying motion estimation and compensation. This final section however, will focus on the application of one of the most common compensation methods, translational block matching, in comparison with the results of the affine matrix based transformation procedure.

To demonstrate the facts and assumptions made throughout this thesis we conclude by comparing image sequences collected via a digital video camera at the main cargo gate of the Georgia Ports Authority. The image sequences were not preprocessed in any way and were selected to attempt to show a sample of various scenarios that are encountered in the Port environment.

The procedure followed to obtain the results is outlined and diagramed below, followed with a greater detailed discussion of the process and the outcomes:

1. **Frames were obtained**
    a. From digital video obtained of trucks using the main access road of the Georgia Ports.
    b. No image pre-processing applied to enhance images
    c. Six different sets of image sequences were chosen for variety

2. **Sequences were processed using Translational** Block Matching
    a. Motion estimation accomplished with an exhaustive search estimation algorithm.
3. **Sequences were processed using an affine** warping transformation
    a. Motion estimation accomplished by first computing the appropriate affine matrix for the sequence.
4. **The image area containing the container code text was isolated** on all estimated and original frames.
5. **The residual image was computed** between the original and estimated frames from both affine and translational procedures.
6. **For quantitative comparison the Mean Absolute Distortion was computed** between the original and its corresponding estimated frame, again from both the affine and translational procedures.

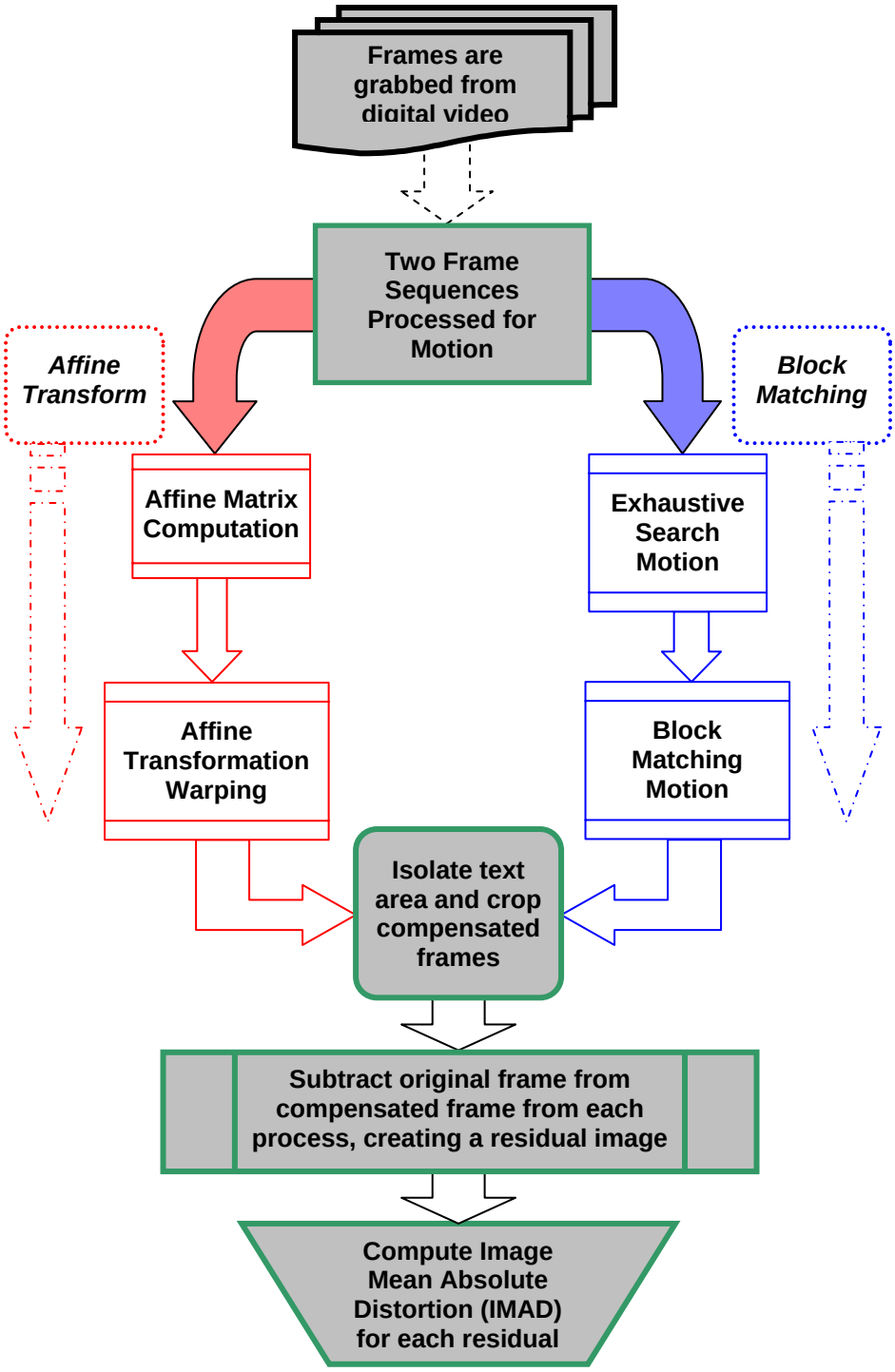

Figure 5.1 Block diagram of process

The following sections go into greater detail on each of the algorithms used in creating the results. All code used can be referenced in Appendix A of this thesis.

5.1.2 Translational Block Matching

Exhaustive Search Motion Estimation Algorithm

This algorithm generates an estimated motion vector for each block within the image. It requires inputs of two images, the first being the "reference frame" or the first frame in the sequence; the second image input is the "current frame" and will be the one the code will compare the reference to compute the motion vector block by block. The other inputs are related to the block search size, you must supply the width and height of the block and the maximum search size. For this thesis, the width and height are set at 16 pixels and the maximum search size is set to 32 pixels. This is sufficient for the images under consideration, if a larger block size were to be used it would speed up the computation time some for certain images.

To perform block matching, the current frame is first divided into several blocks with size N1x N2, in this thesis it is set at a width and height of 16 pixels. For each block B in the current frame the Mean Absolute Distortion (MAD) is computed and is defined as follows:

$$MAD(d_1,d_2) = 1/(N_1*N_2) * \sum|S_{ref}(n_1+d_1,n_2+d_2) - S_{cur}(n_1,n_2)|, \text{ where } (n_1, n_2) \in B \qquad (5.1)$$

$S_{ref}(x, y)$ represents the pixel (x, y) in the reference frame and $S_{cur}(x, y)$ represents the pixel (x, y) in the current frame. Motion displacement is represented by d_1 and d_2 which are integers between –R and R-1 with R being the maximal search size of 32 (in this thesis). The goal is to find a (d_1,d_2) pair for each block such that the $MAD(d_1,d_2)$ is minimized.

The current and reference image files are read through the imread( ) Matlab function to create an array of the intensity values of each pixels in the images. FOR loops were written to go through each block of specified size on the reference frame. For each block in the reference frame, two more FOR loops are ran through to search within the block according to maximal search size, in this thesis it was set at 32 pixels. The desired result is to predict the best match to the block in the current frame based on minimum MAD. A motion vector is created which tells the movement of the block in the reference frame to predict the next frame. The motion vector has x and y axis for the movement of each block in the reference frame.

There are two outputs of this algorithm; the first is a set of intensity values of the inputted image files, titled original_frame and next_frame. The second output is a motion vector consisting of (X=d1, Y=d2) pairs for each block in the image. These values are used in the next step when the actual compensation is applied to the images.

Translational Block-Matching Compensation Algorithm

This portion reads in the values computed in the estimation portion of the process and uses them to create an estimated image of what we called the “current frame” or the second-frame in the sequence. The block size and dimensions are the same as those used in the estimation portion, width and height of 16 pixels and maximum search size of 32 pixels.

Motion compensation is the process of constructing an estimated current frame by replacing each block in the current frame with a best matching block in the reference frame. The best matching block is indicated by the motion vector. For example, suppose we have a 16x16 block in the current frame with the left-top pixel at (17, 33) and the motion vector of this block is (-5,10). This block is replaced by the block whose left-top pixel is (12,43) in the

reference frame. The motion compensated residual is computed, which is the difference between the current frame and the computed estimate frame. The Mean Absolute Distortion (MAD) between the current image and the estimated image, which is the absolute of sum of residuals of all the pixels on the images divided by the product of height and width of the image.

$$\text{Image\_MAD} = \frac{\sum(\sum(\|\text{residual}\|))}{(\text{height} * \text{width}) \text{ of image}} \tag{5.2}$$

A color bar is plotted next to the residual plot, which is set to represent no error in the residual by representing that shade by a zero and the color green. Similarly, the most red shade and highest value represents the most error in the residual.

5.1.3 Affine Transformation

Affine Matrix Computation

This section of the process requires the input of two images and will return the computed 3x3 matrix that describes the displacement between the reference and current frame, as described previously. Because the affine transform allows for so many combinations of transformation, there are many different ways the algorithm can be used. For the purposes of this thesis, the general affine transform will be used with the bottom row of the resultant matrix always containing exactly [0 0 1].

Affine Motion Compensation

The actual application of the affine transform is relatively simple. The previously computed transform matrix is passed into the algorithm along with the image to undergo warping.

5.2 RESULTING DATA AND IMAGES

After processing the image sequences with the processes described above and whose code is detailed in Appendix A, it was apparent that that Affine transform was able to create estimations of motion and compensate for it in a more efficient manner. This can be seen in the table below which provides a summary of the MAD that was computed for the final images and the associated estimates the two compensation techniques created.

Table 5.1 Summary table of MAD values from residual images

Block Matched Residual MAD	Affine Transformed Residual MAD
7.163	4.603
5.598	4.811
5.5281	4.712
6.274	4.904
8.079	7.843
9.019	6.503
Average MAD: 6.944	**Average MAD: 5.563**

5.2.1 Container Sequence 216_B

Figure 5.2 Original container image frames 215 and 216

Figure 5.3 Isolated section of original frame 216_B

Figure 5.4 Estimated frame 216_B using block matching

Figure 5.5 Estimated frame 216_B using affine transformation

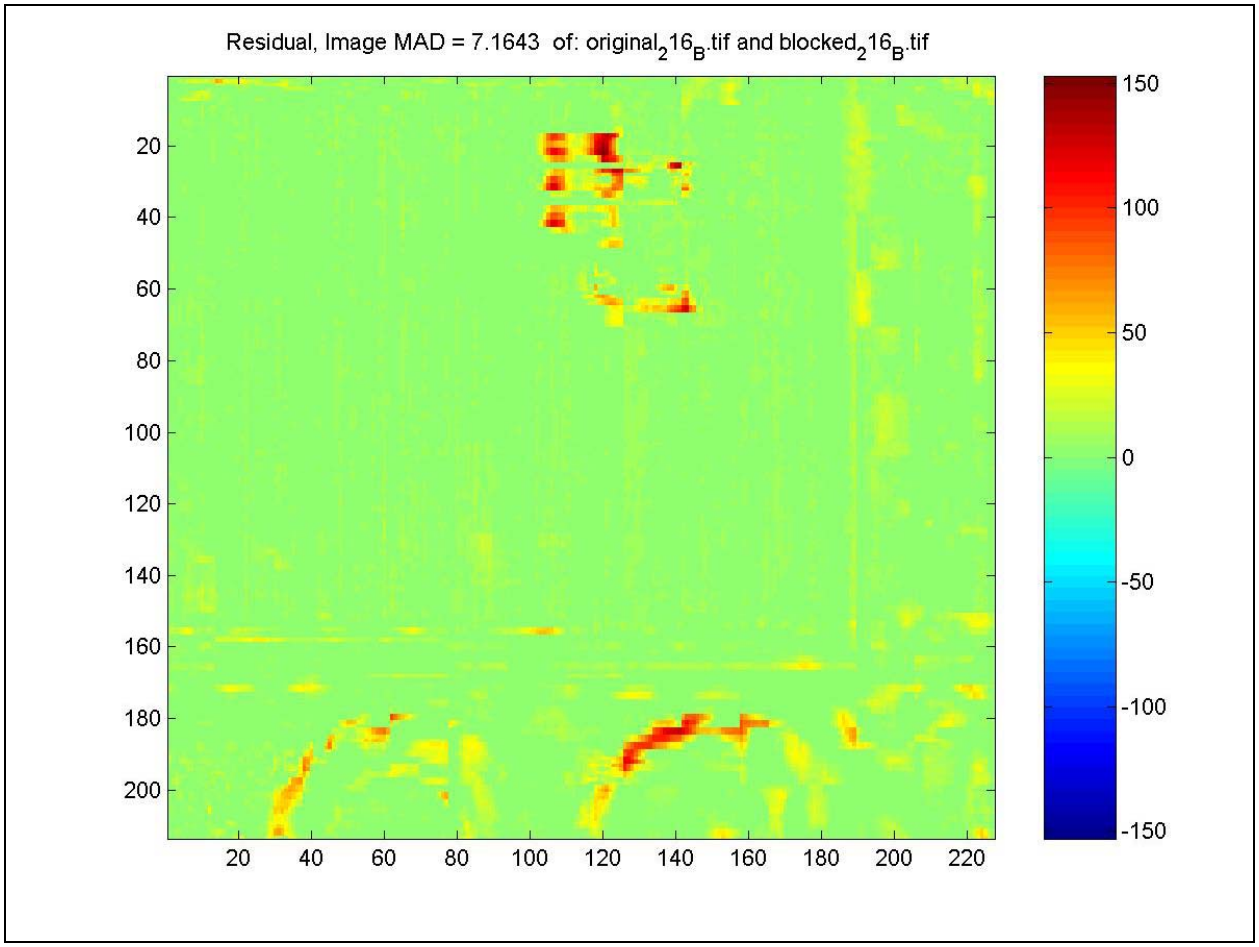

Figure 5.6 Residual image of block match estimate of frame 216_B

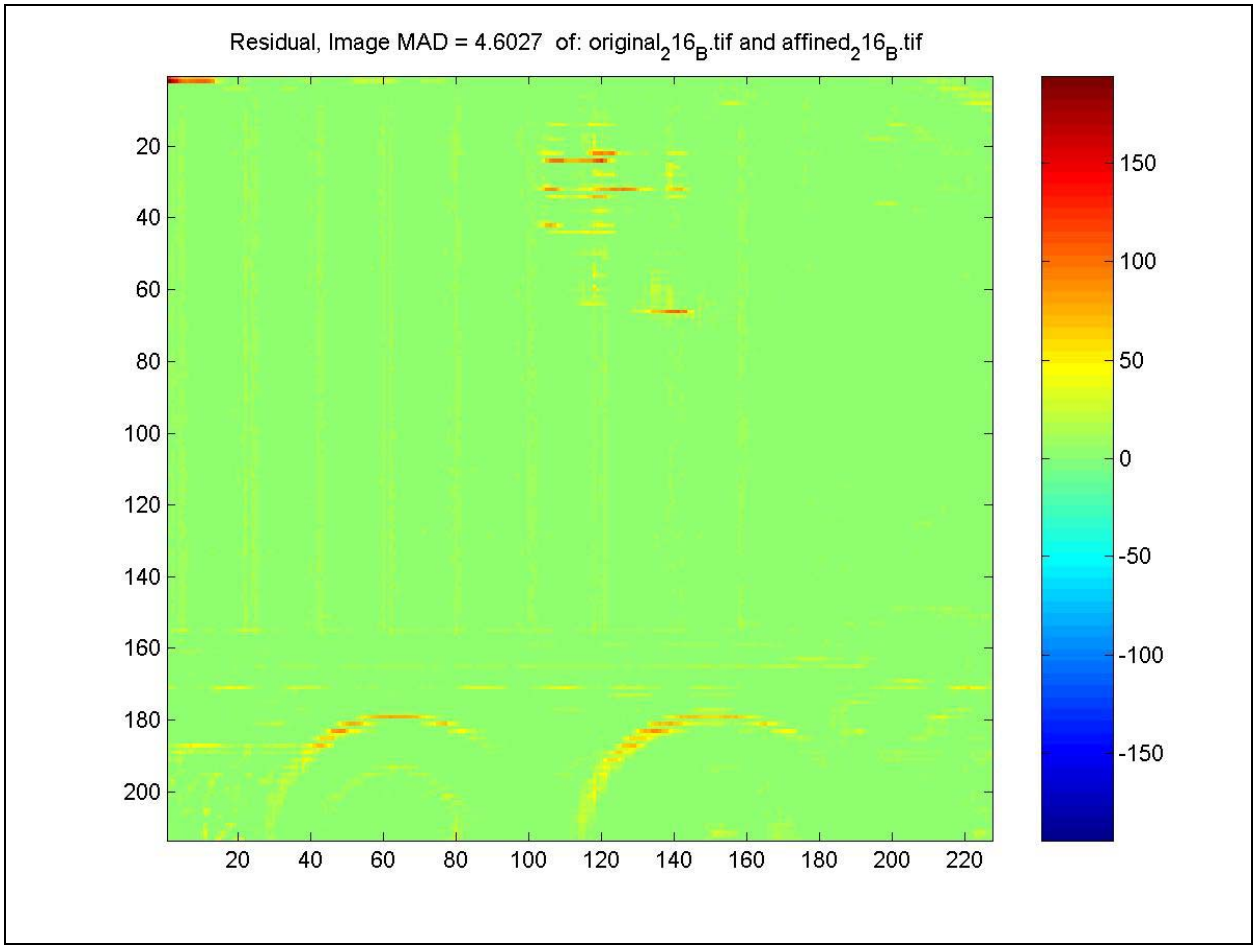

Figure 5.7 Residual image of affine estimate of frame 216_B

5.2.2 Container Sequence 034

Figure 5.8 Original container image frames 033 and 034

Figure 5.9 Isolated section of original frame 034

Figure 5.10 Estimated frame 034 using block matching

Figure 5.11 Estimated frame 034 using affine transformation

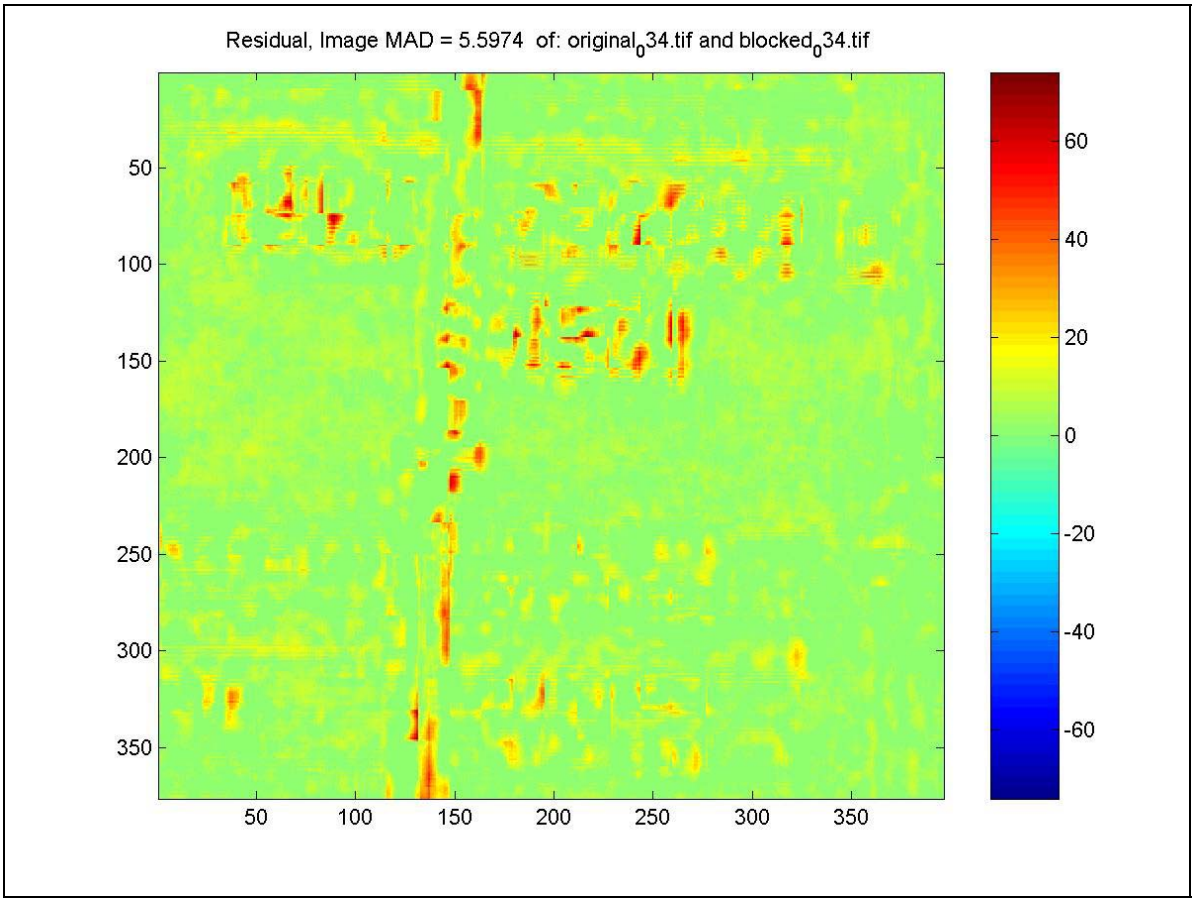

Figure 5.12 Residual image of block match estimate of frame 034

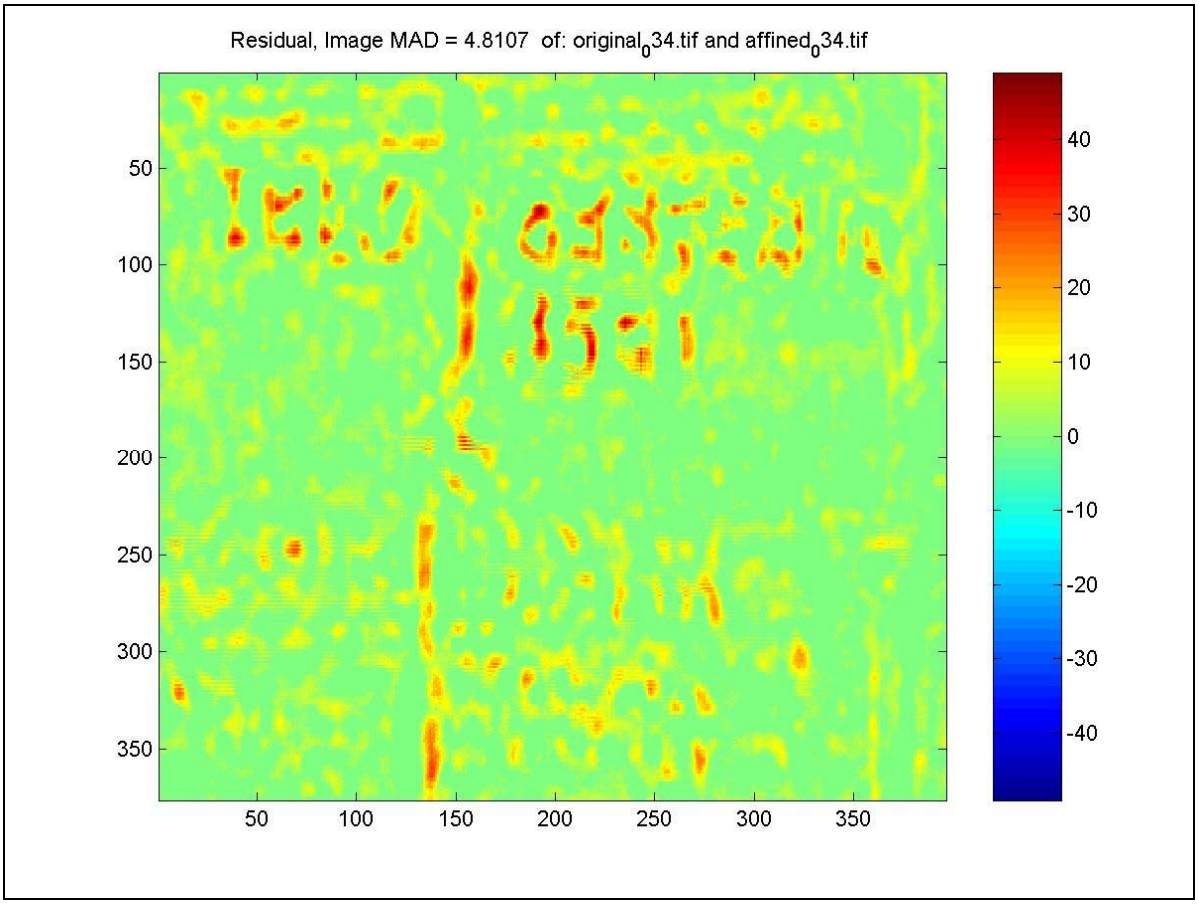

Figure 5.13 Residual image of affine estimate of frame 034

5.2.3 Container Sequence 035

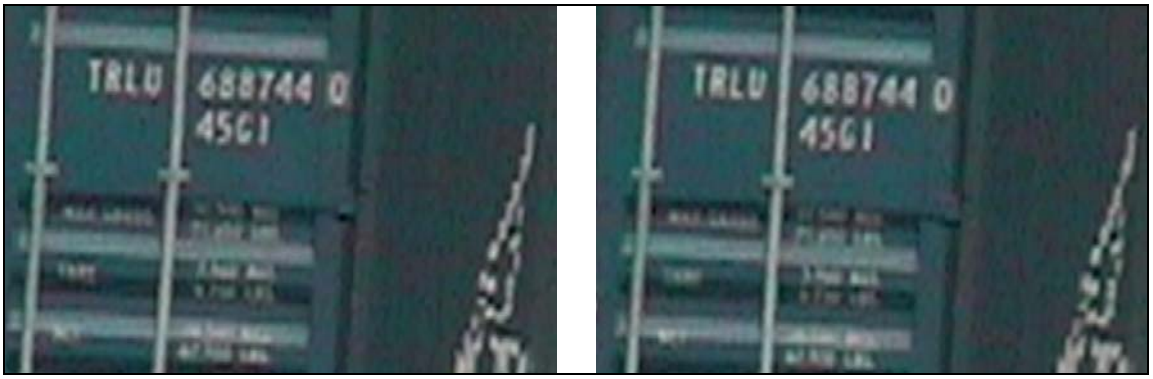

Figure 5.14 Original container image frames 034 and 035

Figure 5.15 Isolated section of original frame 035

Figure 5.16 Estimated frame 035 using block matching

Figure 5.17 Estimated frame 035 using affine transformation

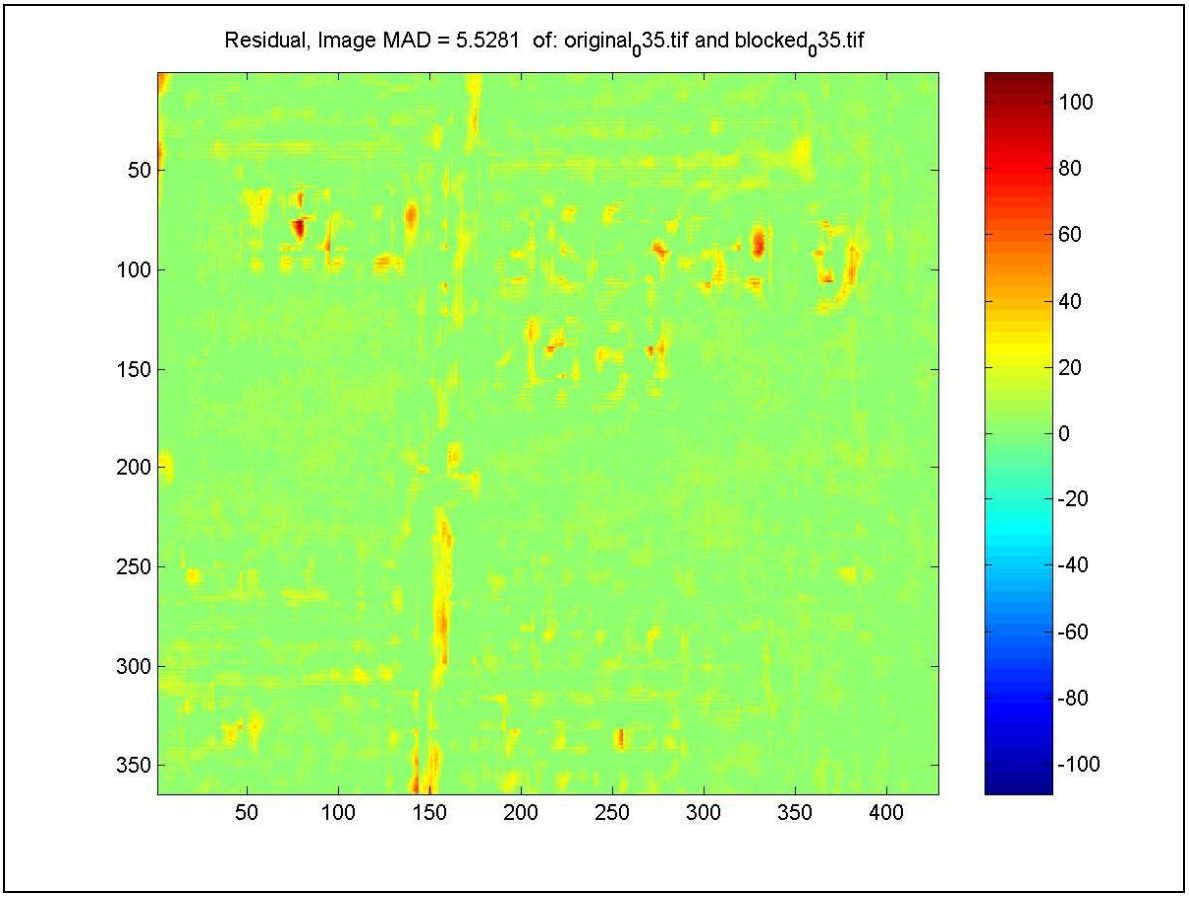

Figure 5.18 Residual image of block match estimate of frame 035

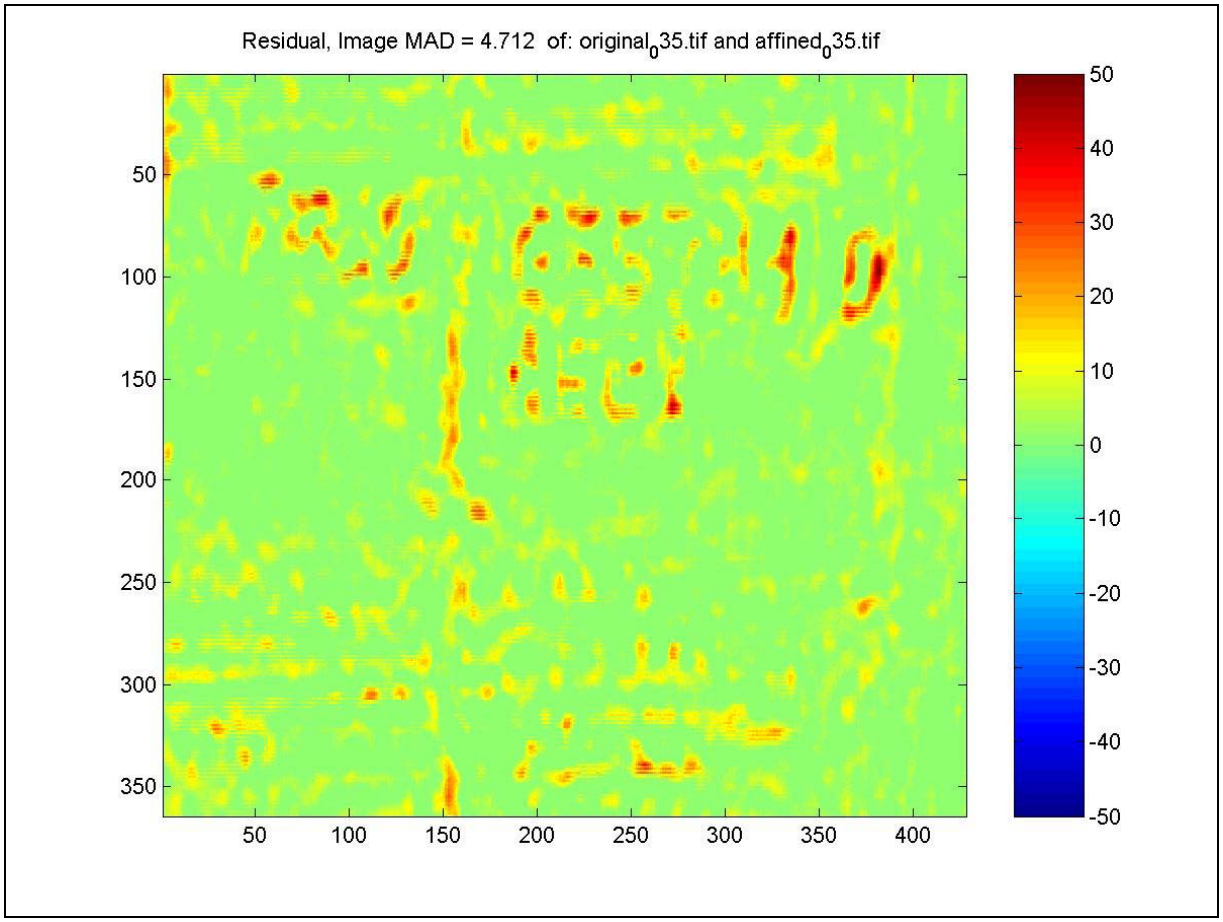

Figure 5.19 Residual image of affine estimate of frame 035

5.2.4 Container Sequence 206

Figure 5.20 Original container image frames 205 and 206

Figure 5.21 Isolated section of original frame 206

Figure 5.22 Estimated frame 206 using block matching

Figure 5.23 Estimated frame 206 using affine transformation

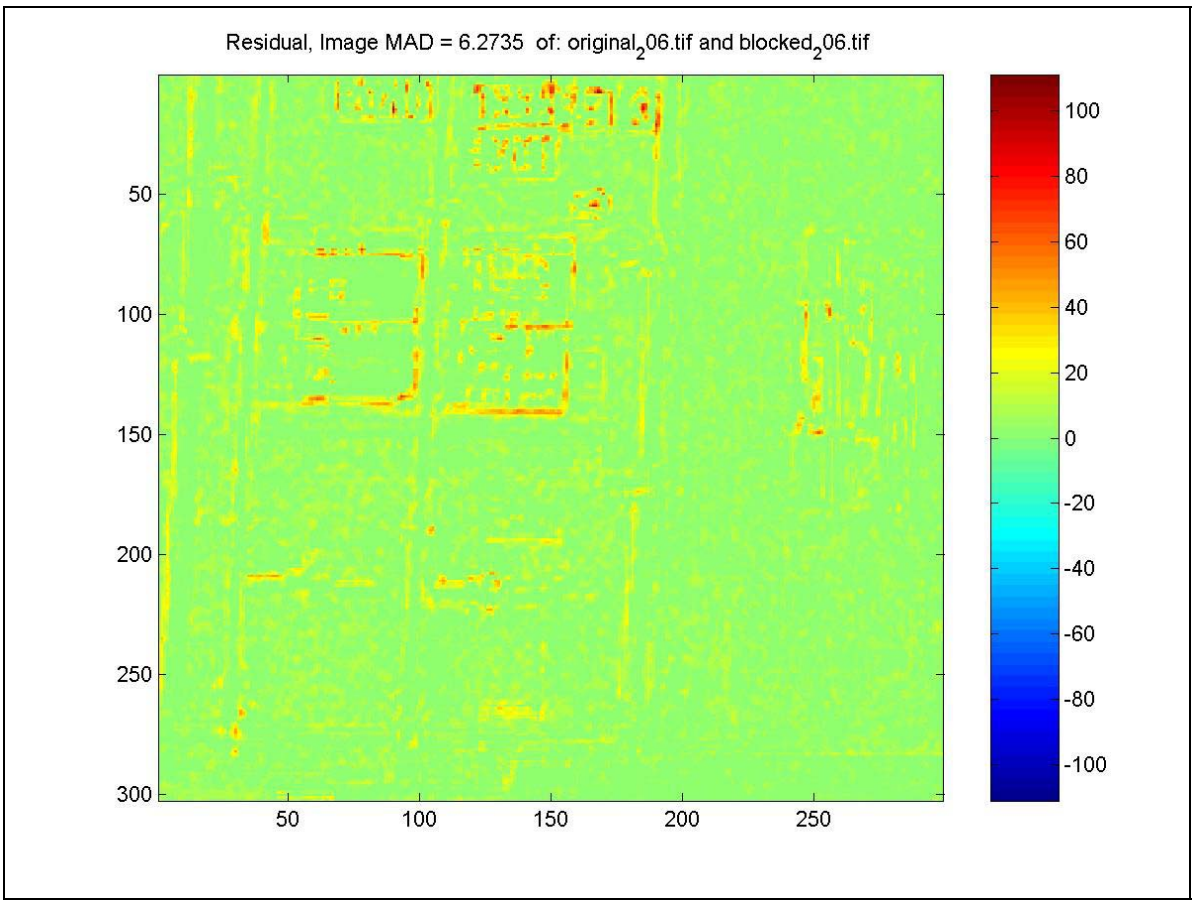

Figure 5.24 Residual image of block match estimate of frame 206

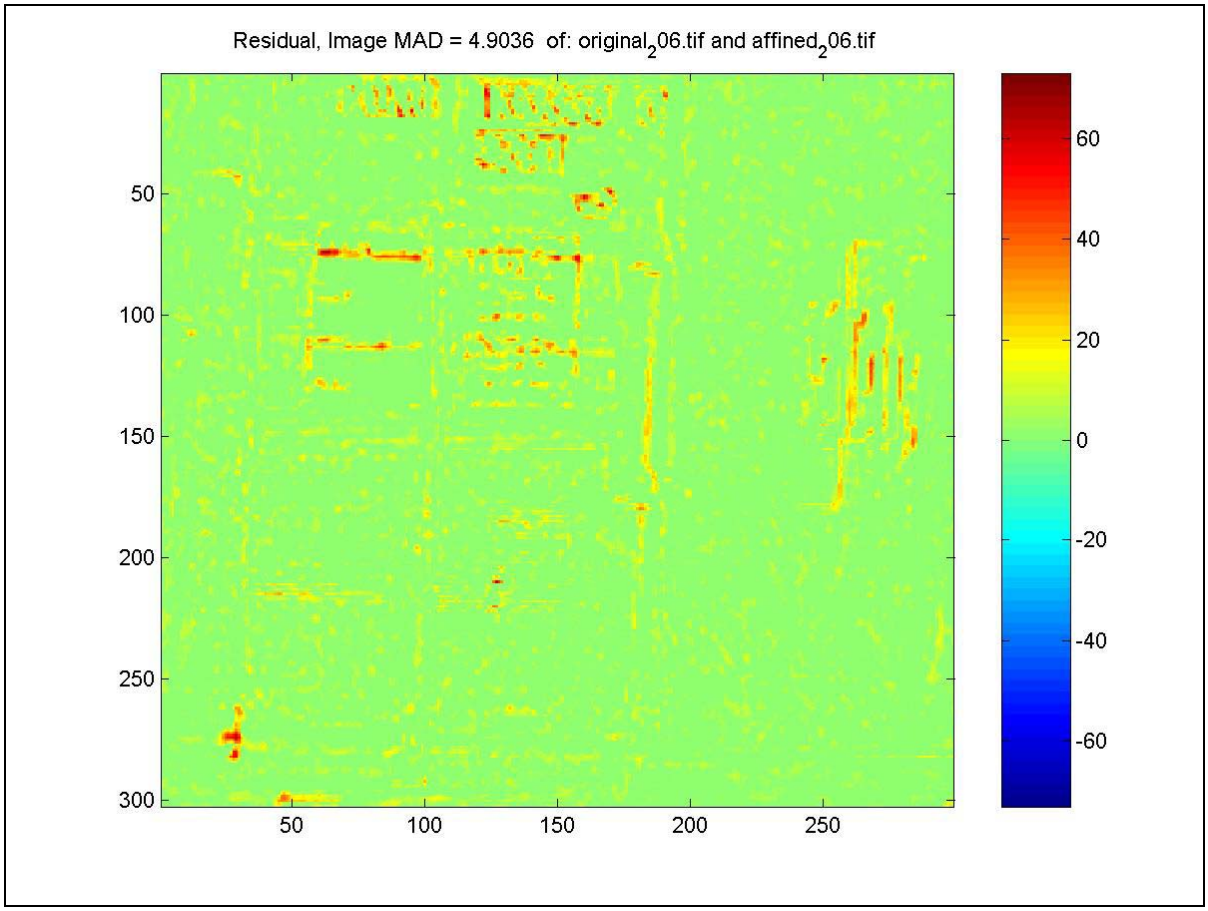

Figure 5.25 Residual image of affine estimate of frame 206

5.2.5 Container Sequence 216

Figure 5.26 Original container image frames 215 and 216

Figure 5.27 Isolated section of original frame 216

Figure 5.28 Estimated frame 216 using block matching

Figure 5.29 Estimated frame 216 using affine transformation

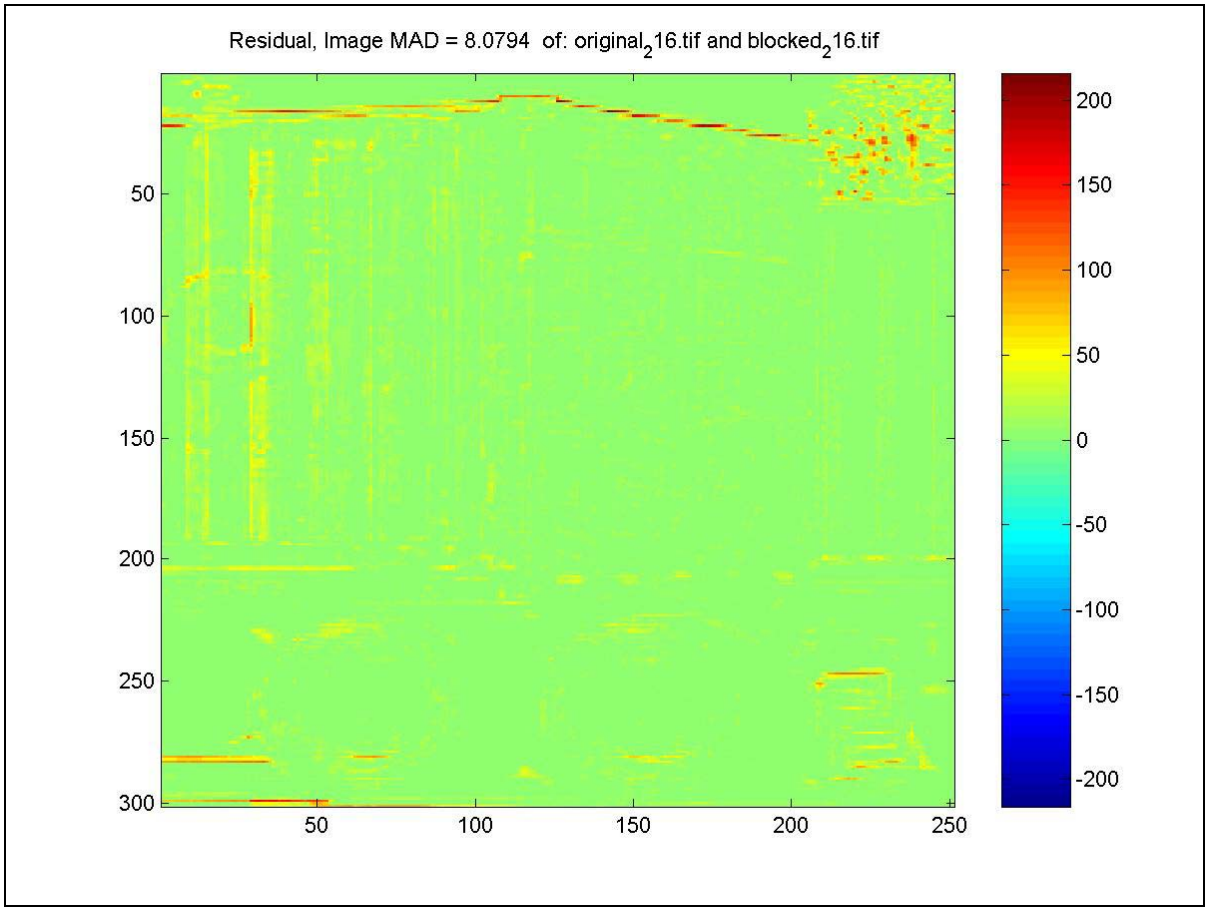

Figure 5.30 Residual image of block match estimate of frame 216

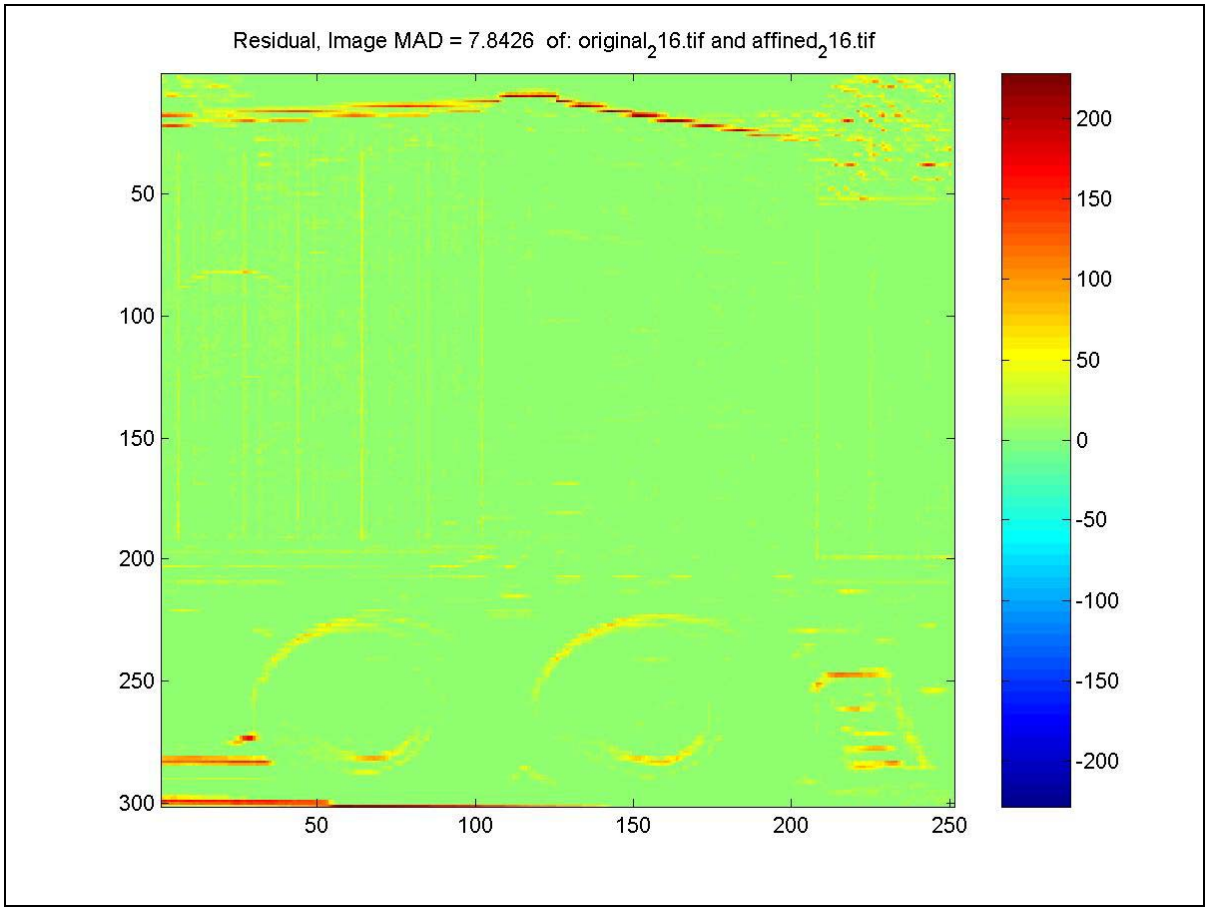

Figure 5.31 Residual image of affine estimate of frame 216

5.2.6 Sequence Container 08

Figure 5.32 Original container image frames 07 and 08

Figure 5.33 Isolated section of original frame 08

Figure 5.34 Estimated frame 08 using block matching

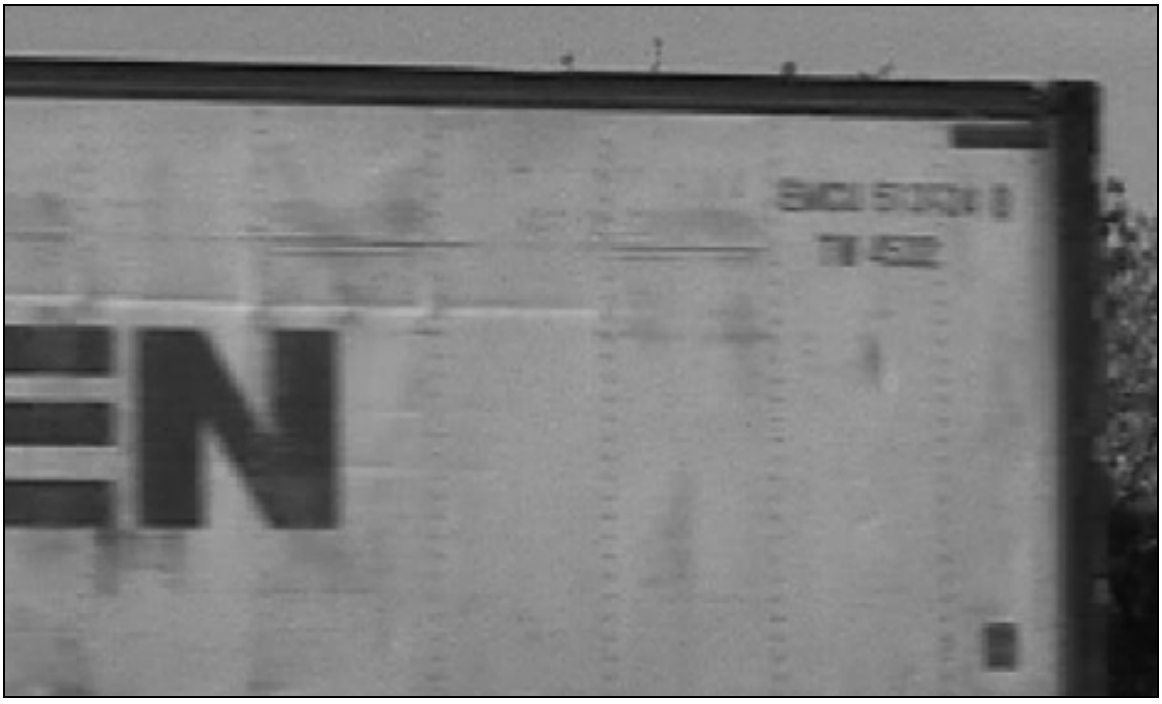

Figure 5.35 Estimated frame 08 using affine transformation

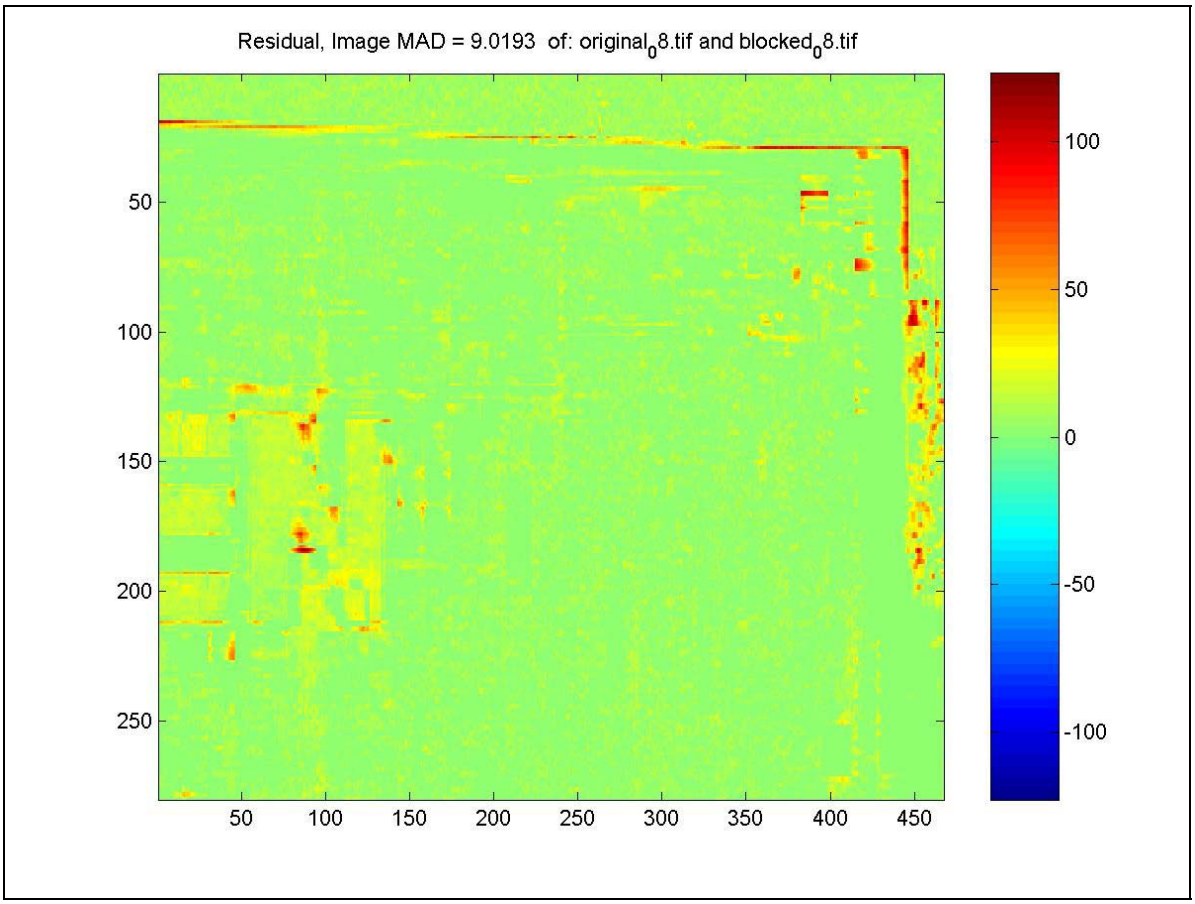

Figure 5.36 Residual image of block match estimate of frame 08

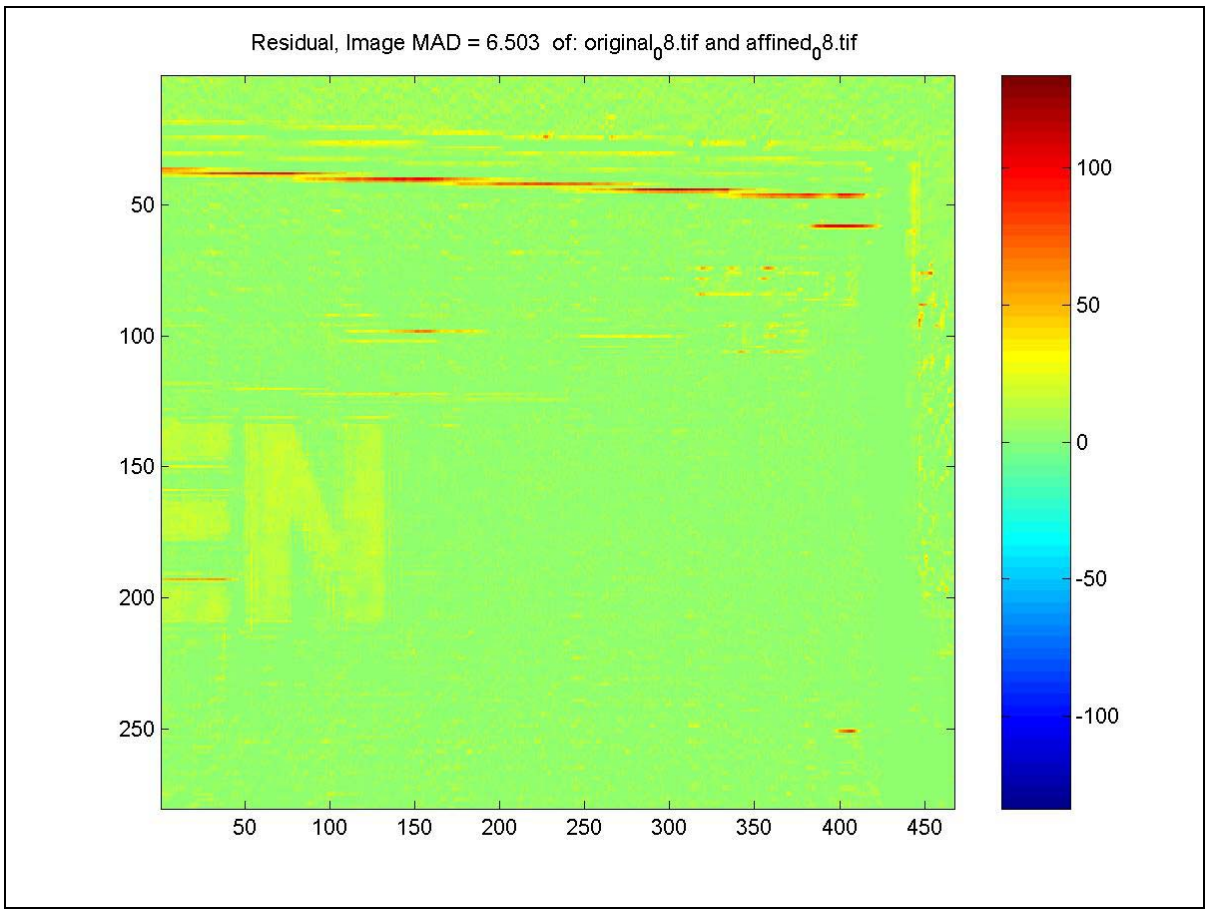

Figure 5.37 Residual image of affine estimate of frame 08

CHAPTER 6

CONCLUSIONS

This thesis investigated the application of an affine based motion compensation algorithm as a complementary enhancement to existing research of OCR and specifically to OCR of cargo containers in a seaport environment. This motion compensation could offer a cost effective alternative to the sophisticated hardware systems being offered to US ports. It was clear both visually and computationally that the affine transform was able to provide a more accurate estimate via motion compensation when tested at a seaport environment.

At its core, the translational model only compensates for square or perpendicular (translational) style motion and again for the application in a seaport environment, it does not quite suit our needs. For these reasons, in our specific application of compensating for motion while identifying cargo container codes in an intermodal environment, the translational motion model is not appropriate.

For many years the translational model and its many derivative methods and alternatives have been the most commonly used motion model. As shown, quite a bit of research has been on the block-matching and gradient based methods and there have been many improvements made to these algorithms over the years. The block matching method is currently implemented into all of the known standard video compression systems because of its ability to handle a lot of motion in the image sequence. The main problem with block matching is that at low bit-rates, which is needed in a seaport environment, introduces blocking artifacts into the reconstructed image. This occurs because the block matching algorithm believes all pixels within a given block are associated with the same object in an

image. The previous sections described two main topics of focus in the area of motion compensation; translational motion models as well as higher order models. The affine transform was highlighted due to its various benefits compared to other methods. By using a higher order motion model such as an affine transformation based model, a higher level of accuracy is usually achieved and thus leads to a higher coding efficiency. This fact was observed during the work accomplished during this thesis. The Affine process, including the computation of the Affine matrix plus the application of that matrix to the image on average took only seconds to accomplish. In comparison, the block matching process, including the exhaustive search estimation plus the compensation portion took on average an order of magnitude longer, taking minutes to complete.

The following are some additional summary observations made during the course of this thesis:

- Motion compensation is needed to allow the container to keep moving while capturing the characters for OCR processing.
- Motion will not always be translational in nature.
- Block-matching does not allow the level of compression needed to handle the volume of containers transitioning through the port per day, given the network infrastructure available due to sheer size (acreage) of the facilities.
- There are many constraints to performing OCR on a container.
- More constraints are added due to the seaport environment.
- Commercial alternatives not widely accepted due to cost and complexity of available OCR systems.

APPENDIX A

MATLAB CODES

```
function image_MAD = mc(N1,N2,R,mv, original_frame, next_frame, filename_ref,
                                filename_current);

% Generate and plot the estimated frame using the motion vector of exhaustive search
%
%  image_MAD = motion_compensation(N1,N2,R,mv, original_frame, next_frame)
%  Input:
%     N1: height of each block in the image.
%     N2: width of each block in the range.
%     R:  maximal search size.
%     mv: motion vector is a (x = d1, y = d2) pair for each block such that the MAD(d1, d2)
%          is minimized.
%     original_frame: intensity values of each pixel of the reference image file.
%     next_frame: intensity values of each pixel of the current image file.
%     filename_ref: file names of 1st image
%     filename_current:  file names of 2nd image
%
%  Output:
%     image_MAD: mean absolute distortion (MAD) between the current image and
%                      the estimated image
%
% % ************************************************************
% % begin timing the overall process
% % ************************************************************
timestamp

% % ************************************************************
% % Plotting the original and next frame
% % ************************************************************
figure
subplot(2,3,1), imshow(uint8(original_frame)),title(['Original Frame: ' , filename_ref, ' '])
subplot(2,3,2), imshow(uint8(next_frame)),title(['Next Frame: ' , filename_current, ' '])

% % ************************************************************
% % Defines and Loads relevant variables for the estimating the
% % next frame
% % ************************************************************
[height, width] = size(original_frame);

block_height = round(height/N1);
block_width = round(width/N2);

estimated_frame = 255*(ones(height,width));

% % ************************************************************
% % Takes the Motion vector mv, and transforms it into the new
% % position vectors
% % ************************************************************
pos_old_col = 1:N2:width;
pos_mat(:,:,1) = repmat(pos_old_col,block_height,1);
pos_old_row = 1:N1:height;
pos_mat(:,:,2) = repmat(pos_old_row,block_width,1).';
pos_new = pos_mat + mv;

% % ************************************************************
% % Creating the estimated frame using the new position vector
```

```
% % ************************************************************
i_col = 1;
i_row = 1;
for ii_blks= 1:block_height*block_width % for every block in the frame
   new_x = pos_new(i_row,i_col,1);
   new_y = pos_new(i_row,i_col,2);
   estimated_frame(pos_old_row(i_row):pos_old_row(i_row)+N1-
1,pos_old_col(i_col):pos_old_col(i_col)+N2-1) ...
       = original_frame(new_y:(new_y+N1-1),new_x:(new_x+N2-1));
       i_col = i_col + 1;
   if i_col == (block_width+1)
       i_col = 1;
       i_row = i_row + 1;
   end
   if i_row == (block_height+1)
       i_row = 1;
       i_col = i_col + 1;
   end
end

% % ************************************************************
% % Computing the difference between the current frame and the
% % estimated frame which is called residual
% % ************************************************************
residual = ((double(next_frame) - estimated_frame));

% % ************************************************************
% % Calculating image_MAD: mean absolute distortion (MAD) between
% % the current image and the estimated image
% % ************************************************************
image_MAD = sum(sum(abs(residual)))/(size(residual,1)*size(residual,2));

% % ************************************************************
% % Creating the range of greyness intensity for the residual that
% % contains all the intensities of greyness
% % ************************************************************
maxR = max(abs([min(min(residual)) max(max(residual))]));

axis auto % Setting the axis back to default

% % ************************************************************
% % Plotiing the estimated frame and residual
% % ************************************************************
subplot(2,3,4), imshow(uint8(estimated_frame)),title(['Estimated Frame of ',
                        filename_current, ' '])
subplot(2,3,5), imagesc(uint8(residual),[-maxR maxR]),title(['Residual, Image MAD = ',
                        num2str(image_MAD),'  of: ', filename_ref, ' and ', filename_current,
                        ' ']),colorbar
figure
imshow(uint8(estimated_frame)),title(['Estimated Frame of ', filename_current, ' '])
figure
imagesc(uint8(residual),[-maxR maxR]),title(['Residual, Image MAD = ',
num2str(image_MAD),'  of: ', filename_ref, ' and ', filename_current, ' ']),colorbar

% % ************************************************************
% % END timing the overall process
```

**% % **
timestamp**

```
function [mv,original_frame,next_frame] = me(N1,N2,R,filename_ref,filename_current);

% Generate motion vector using Exhaustive search algorithm
%
%  [mv,original_frame,next_frame] = me(N1,N2,R,filename_ref,filename_current)
%
%  Input:
%     N1: height of each block in the image.
%     N2: width of each block in the range.
%     R:  maximal search size.
%     filename_ref: name of the image file of the reference frame.
%     filename_current: name of the image file of the current frame.
%
%  Output:
%     mv = motion vector is a (x = d1, y = d2) pair for each block such that the MAD(d1, d2)
%          is minimized.
%     original_frame: intensity values of each pixel of the reference image file.
%     next_frame: intensity values of each pixel of the current image file.

% % ************************************************************
% % begin timing the overall process
% % ************************************************************
timestamp

% % ************************************************************
% % Reading the current image flie into next_frame
% % ************************************************************
next_frame = double(rgb2gray((imread(filename_current))));
imshow(uint8(next_frame));

% % ************************************************************
% % Reading the previous image flie into original_frame
% % ************************************************************
original_frame = double(rgb2gray(imread(filename_ref)));

% % ************************************************************
% % Finding the size of the original frame
% % ************************************************************
[height, width] = size(original_frame);

% % ************************************************************
% % Initializing few variables and arrays that will later be used
% % in the for loop
% % ************************************************************

counter = 1;
motion_vector_x = 0;
motion_vector_y = 0;
blk_row = round(height/N1);
blk_col = round(width/N2);

mv = zeros(blk_row,blk_col,2);

motion_vector = zeros(1,(blk_row*blk_col)+1,2);

for i=1:N1:(height),
```

```
   for j=1:N2:(width)   % for every block in the reference frame
      MAD_min=double([]);  % initialzing minimum MAD to highest value possible
      motion_vector(1,counter,1) = motion_vector_x;  % motion vector of each block
      motion_vector(1,counter,2) = motion_vector_y;  % motion vector of each block
      counter = counter + 1;
       for k=-R:1:R-1,
         for l=-R:1:R-1   % for every search candidate thus motion vector will have values
                             from -R to R-1
            % % ***************************************************************
            % % Checking if the search is moving out of the image
            % % ***************************************************************
            if (i+k) > 0 && (i+k+N2-1) > 0 && (j+l) > 0 && (i+k) < (i+k+N2-1) && (j+l+N1-1) > 0 ...
                  && (j+1) < (j+l+N1-1) && (i+k) < height && (j+l) < width && (i+k+N2-1)...
                  < height + 1 && (j+l+N1-1) < width + 1
               % % ***************************************************************
               % % Calculating MAD with in the search size
               % % ***************************************************************
               MAD_temp=double((1/(N1*N2))*sum(sum(abs(next_frame(i:i+N2-1,j:j+N1-1)...
                  - original_frame((i+k):(i+k+N2-1),(j+l):(j+l+N1-1))))));
               % % ***************************************************************
               % % Finding the minimum MAD
               % % ***************************************************************
               if isempty(MAD_min)
                  MAD_min=MAD_temp;
                  motion_vector_x = l;
                  motion_vector_y = k;
               elseif MAD_temp < MAD_min
                  MAD_min=MAD_temp;
                  motion_vector_x = l;
                  motion_vector_y = k;
               end;
            end;
         end;
       end;
   end;
end;

% % ***************************************************************
% % Putting the motion vector for the last block
% % ***************************************************************
motion_vector(1,counter,1) = motion_vector_x;
motion_vector(1,counter,2) = motion_vector_y;

% % ***************************************************************
% % Putting the motion vector in a form that can be read by MC
% % ***************************************************************
counter = 1;
for i = 2:blk_col:(blk_row*blk_col)
   mv(counter,:,1) = motion_vector(1,i:i+blk_col-1,1);
   counter = counter + 1;
end

counter = 1;
for i = 2:round(width/N2):(blk_row*blk_col)
   mv(counter,:,2) = motion_vector(1,i:i+blk_col-1,2);
   counter = counter + 1;
```

```
end

% % ************************************************************
% % END timing the overall process
% % ************************************************************
timestamp
```

```
function image_MAD = residual(filename_ref, filename_affine);

%% computes residual and MAD from two images

estimated_frame = double((imread(filename_affine)));

ref_frame = double((imread(filename_ref)));

residual = ((double(ref_frame) - estimated_frame));

image_MAD = sum(sum(abs(residual)))/(size(residual,1)*size(residual,2));

maxR = max(abs([min(min(residual)) max(max(residual))]));

axis auto % Setting the axis back to default

figure
imagesc(uint8(residual),[-maxR maxR]),title(['Residual, Image MAD = ',
num2str(image_MAD),'  of: ', filename_ref, ' and ', filename_affine, ' ']),colorbar
```

```
function result = warpAffine2(im,A)

% function result = warpAffine2(im,A)
%
% im: input image
% A: 3x3 affine transform matrix with [0 0 1] for the last row.
%
% if a transformed point is outside of the volume, NaN is used
%
% result: output image, same size as im
%
% % ************************************************************

im = double(rgb2gray((imread(im))));

A=double(A);

if (size(A,1)>2)
  A=A(1:2,:);
end

% % ************************************************************
% % Compute coordinates corresponding to input and transformed coordinates for result
% % ************************************************************

% % ************************************************************
% % begin timing the overall process
% % ************************************************************
timestamp

% % ************************************************************
[x,y] = meshgrid(1:size(im,2),1:size(im,1));
cords = [x(:)'; y(:)'];
homogeneousCoords = [coords; ones(1,prod(size(im)))];
warpedCoords = A*homogeneousCoords;
xprime = warpedCoords(1,:)';
yprime = warpedCoords(2,:)';

result = interp2(x,y,im,xprime,yprime);
result = reshape(result,size(im));

figure
imshow(uint8(im)),title('Original')

figure
imshow(uint8(result)),title('Warped')

residual = ((double(im) - result));
maxR = max(abs([min(min(residual)) max(max(residual))]));

image_MAD = sum(sum(abs(residual)))/(size(residual,1)*size(residual,2));

figure
imagesc(uint8(residual),[-maxR maxR]),title(['Residual, Image MAD = ',
            num2str(image_MAD),' ']),colorbar
```

**% % **
% % END timing the overall process
**% % **
timestamp

```
function [M,w2d] = estMotion2(im1,im2,rotFlag,robustFlag)

% function [M,w] = estMotion2(im1,im2,rotFlag,robustFlag)
%
% im1 and im2 are images
%
% M is 3x3 transform matrix: X' = M X
% where X=(x,y,1) is starting position in homogeneous coords
% and X'=(x',y',1) is ending position
%
% If rotFlag is activated ( = 1), then M is a rotation+translation,
% If rotFlag is NOT-activated ( = 0), then M , is a general affine transform
%
% If robustFlag is activated ( = 1) then uses a robust M-estimator
% If robustFlag is NOT-activated ( = 0) then uses conventional Least Squares
%
% Solves fs^t theta + ft = 0
% where theta = B p is image velocity at each pixel
%       B is 2x3 (2x6 if affine) matrix that depends on image positions
%       p is vector of trans+rot (or affine) motion parameters
%       fs is vector of spatial derivatives at each pixel
%       ft is temporal derivative at each pixel
% Mulitplying fs^t B gives a 1x6 vector for each pixel.  Piling
% these on top of one another gives A, an Nx3 (Nx6 if affine) matrix, where N is
% the number of pixels.  Solve M p = ft where ft is now an Nx1
% vector of the the temporal derivatives at every pixel.
%

% % ************************************************************
% % begin timing the overall process
% % ************************************************************
timestamp

% read images in
im1 = double(rgb2gray((imread(im1))));
im2 = double(rgb2gray((imread(im2))));

% default values
if ~exist('robustFlag')
  robustFlag = 0;
end
if ~exist('rotFlag')
  rotFlag = 1;
end

[fx,fy,ft]=computeDerivatives2(im1,im2);
[xgrid,ygrid]=meshgrid(1:size(im1,2),1:size(im1,1));

% subsample
dims=size(fx);
fx=fx([1:2:dims(1)],[1:2:dims(2)]);
fy=fy([1:2:dims(1)],[1:2:dims(2)]);
ft=ft([1:2:dims(1)],[1:2:dims(2)]);
% *** Assumes that the filters have 5 taps!
xgrid=xgrid([3:2:dims(1)+2],[3:2:dims(2)+2]);
ygrid=ygrid([3:2:dims(1)+2],[3:2:dims(2)+2]);
```

```
dimsS=size(fx);
pts=find(~isnan(fx));
pts=find((~isnan(fx))&(~isnan(fy))&(~isnan(ft)));
fx = fx(pts);
fy = fy(pts);
ft = ft(pts);
xgrid = xgrid(pts);
ygrid = ygrid(pts);

if rotFlag
        A= [ fx(:), fy(:), xgrid(:).*fy(:)-ygrid(:).*fx(:)];
else
        A= [xgrid(:).*fx(:), ygrid(:).*fx(:), fx(:),...
           xgrid(:).*fy(:), ygrid(:).*fy(:), fy(:)];
end
b = -ft(:);

if robustFlag
        [p w] = robustMest(A,b,CB,SC);
        w2d = zeros(dimsS);
        w2d(pts)=w;
else
        p = A\b;
        w2d = [];
end

if rotFlag
        M= [cos(p(3))  -sin(p(3)) p(1);
           sin(p(3))  cos(p(3))  p(2);
           0     0    1];
else
        M= [1+p(1) p(2)   p(3);
           p(4)   1+p(5) p(6);
           0      0      1];
end

% % ************************************************************
% % END timing the overall process
% % ************************************************************
timestamp
```

REFERENCES

[1] E. Alpaydin, “Techniques for combining multiple learners,” in *Proc.Eng. Intell.*, 1998, vol. 2, pp. 6–12.

[2] N. Arica and F. T. Yarman-Vural, “One dimensional representation of two dimensional information for HMM based handwritten recognition,” *Pattern Recognit. Lett.*, vol. 21, no. 6–7, pp. 583–592, 2000.

[3] A. Atici and F. Yarman-Vural, “A heuristic method for Arabic character recognition,” *Signal Process.*, vol. 62, pp. 87–99, 1997.

[4] H. I. Avi-Itzhak, T. A. Diep, and H. Gartland, “High accuracy optical character recognition using neural network with centroid dithering,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 17, pp. 218–228, Feb. 1995.

[5] H. S. Baird, “Document image defect models,” in *Proc. Int. Workshop Syntactical Structural Pattern Recognit.*, 1990, pp. 38–47.

[6] O. Baruch, “Line thinning by line following,” *Pattern Recognit. Lett.*, vol. 8, no. 4, pp. 271–276, 1989.

[7] S. O. Belkasim, M. Shridhar, and M. Ahmadi, “Pattern recognition with moment invariants: A comparative survey,” *Pattern Recognit.*, vol. 24, no. 12, pp. 1117–1138, 1991.

[8] D. Bouchaffra, V. Govindaraju, and S. N. Srihari, “Postprocessing of recognized strings using nonstationary Markovian models,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 21, pp. 990–999, Oct. 1999.

[9] M. Cannon, J. Hockberg, and P. Kelly, “Quality assessment and restoration of typewritten document images,” *Int. J. Document Anal. Recognit.*, vol. 2, no. 2/3, pp. 80–89, 1999.

[10] R. G. Casey and E. Lecolinet, “A survey of methods and strategies in character segmentation,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 18, pp. 690–706, July 1996.

[11] M. Y. Chen, A. Kundu, and J. Zhou, “Off-line handwritten word recognition using a hidden Markov model type stochastic network,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 16, pp. 481–496, May 1994.

[12] M. Y. Chen, A. Kundu, and S. N. Srihari, “Variable duration hidden Markov model and morphological segmentation for handwritten word recognition,” *IEEE Trans. Image Processing*, vol. 4, pp. 1675–1688, Dec. 1995.

[13] M. Chen and X. Ding, “A robust skew detection algorithm for gray-scale document image,” in *Proc. 5th Int. Conf. Document Anal. Recognit.*, Bangalore, India, 1999, pp. 617–620.

[14] F. H. Cheng, W. H. Hsu, and C. A. Chen, “Fuzzy approach to solve the recognition problem of handwritten Chinese characters,” *Pattern Recognit.*, vol. 22, no. 2, pp. 133–141, 1989.

[15] H. Cheng,W. H. Hsu, and M. C. Kuo, “Recognition of handprinted Chinese characters via stroke relaxation,” *Pattern Recognit.*, vol. 26, no. 4, pp. 579–593, 1993.

[16] Y. C. Chim, A. A. Kassim, and Y. Ibrahim, “Character recognition using statistical moments,” *Image Vis. Comput.*, vol. 17, pp. 299–307, 1999.

[17] M. Cote *et al.*, “Reading of cursive scripts using a reading model and perceptual concepts, the PERCEPTO system,” *Int. J. Document Anal. Recognit.*, vol. 1, no. 1, pp. 3–17, 1998.

[18] P. A. Devijer and J. Kittler, *Pattern Recognition: A Statistical Approach*. Englewood Cliffs, NJ: Prentice-Hall, 1982.

[19] C. Downtown and C. G. Leedham, “Preprocessing and presorting of envelope images for automatic sorting using OCR,” *Pattern Recognit.*, vol. 23, no. 3–4, pp. 347–362, 1990.

[20] P. D. Gader *et al.*, “Recognition of handwritten digits using template and model matching,” *Pattern Recognit.*, vol. 24, no. 5, pp. 421–431, 1991.

[21] W. Guerfaii and R. Plamondon, “Normalizing and restoring on-line handwriting,” *Pattern Recognit.*, vol. 26, no. 3, pp. 418–431, 1993.

[22] D. Guillevic and C. Y. Suen, “Cursive script recognition: A sentence level recognition scheme,” in *Proc. Int. Workshop Frontiers in Handwriting Recognit.*, 1994, pp. 216–223.

[23] , “HMM–KNN word recognition engine for bank cheque processing,” in *Proc. 14th Int. Conf. Pattern Recognit.*, Brisbane, Australia,
1998, pp. 1526–1529.

[24] Y. Hamamoto *et al.*, “Recognition of handprinted Chinese characters using Gabor features,” in *Proc. 3rd Int. Conf. Document Anal. Recognit.*, Montreal, QC, Canada, 1995, pp. 819–823.

[25] A. Hashizume, P. S. Yeh, and A. Rosenfeld, “A method of detecting the orientation of aligned components,” *Pattern Recognit. Lett.*, vol. 4, pp. 125–132, 1986.

[26] J. Hu and T. Pavlidis, “A hierarchical approach to efficient curvilinear object searching,” *Comput. Vis. Image Understanding*, vol. 63, no. 2, pp. 208–220, 1996.

[27] A. K. Jain and D. Zongker, “Representation and recognition of handwritten digits using deformable templates,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 19, pp. 1386–1391, Dec. 1997.

[28] A. K. Jain, R. P. W. Duin, and J. Mao, “Statistical pattern recognition: A review,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 22, pp. 4–38, Jan. 2000.

[29] B. H. Huang and L. R. Rabiner, “The segmental K-means algorithm for estimating parameters of hidden Markov models,” *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 38, pp. 1639–1641, Sept. 1990.

[30] J. Kanai and A. D. Bagdanov, “Projection profile based skew estimation algorithm for JPIG compressed images,” *Int. J. Document Anal. Recognit.*, vol. 1, no. 1, pp. 43–51, 1998.

[31] H. Y. Kim and J. H. Kim, “Handwritten Korean character recognition based on hierarchical random graph modeling,” in *Proc. Int. Workshop Frontiers Handwriting Recognit.*, 1998, pp. 577–586.

[32] W. Y. Kim and P.Yuan, “A practical pattern recognition system for translation, scale and rotation invariance,” in *Proc. Int. Conf. Comput. Vis. Pattern Recognit.*, Seattle, WA, June 1994.

[33] J. Kitler, “Feature selection and extraction,” in *Handbook of Pattern Recognition and Image Processing*, T. Young and K. Fu, Eds. New York: Academic, 1986, pp. 59–83.

[34] A. Kornai, K. M. Mohiuddin, and S. D. Connell, “An HMM-based legal amount field OCR system for checks,” in *Proc. IEEE Int. Conf. Syst., Man, Cybern., Intell. Syst. 21st Century*, vol. 3, 1995, pp. 2800–2805.

[35] T. Kunango, R. Haralick, and I. Phillips, “Nonlinear local and global document degradation models,” *Int. J. Imaging Syst. Technol.*, vol. 5, no. 4, pp. 274–282, 1994.

[36] A. Kundu and Y. He, “On optimal order in modeling sequence of letters in words of common language as a Markov chain,” *Pattern Recognit.*, vol. 24, no. 7, pp. 603–608, 1991.

[37] L. Lam, S. W. Lee, and C. Y. Suen, “Thinning methodologies—A comprehensive survey,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 14,
pp. 869–885, Sept. 1992.

[38] W. L. Lee and K.-C. Fan, “Document image preprocessing based on optimal Boolean filters,” *Signal Process.*, vol. 80, no. 1, pp. 45–55, 2000.

[39] R. Legault and C. Y. Suen, “Optimal local weighted averaging methods in contour smoothing,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 18, pp. 690–706, July 1997.

[40] J. G. Leu, “Edge sharpening through ramp width reduction,” *Image Vis. Comput.*, vol. 18, no. 6–7, pp. 501–514, 2000.

[41] X. Li, W. Oh, J. Hong, and W. Gao, “Recognizing components of handwritten characters by attributed relational graphs with stable features,” in *Proc. 4th Int. Conf. Document Anal. Recognit.*, 1997,
pp. 616–620.

[42] S. Madhvanath, G. Kim, and V. Govindaraju, “Chaincode contour processing for handwrittenword recognition,” *IEEE Pattern. Anal. Machine Intell.*, vol. 21, pp. 928–932, Sept. 1999.

[43] M. A. Mohamed and P. Gader, “Generalized hidden Markov models—Part II: Application to handwritten word recognition,” *IEEE Trans. Fuzzy Syst.*, vol. 8, pp. 82–95, Feb. 2000.

[44] , “Handwritten word recognition using segmentation-free hidden Markov modeling and segmentation based dynamic programming techniques,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 18, pp.
548–554, May 1996.

[45] I. S. Oh, J. S. Lee, and C. Y. Suen, “Analysis of class separation and combination of class-dependent features for handwriting recognition,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 21, pp. 1089–1094, Oct. 1999.

[46] M. Okamoto and K. Yamamoto, “On-line handwriting character recognition method with directional features and direction change features,” in *Proc. 4th Int. Conf. Document Anal. Recognit.*, Ulm, Germany, 1997, pp. 926–930.

[47] T. Pavlidis, “Recognition of printed text under realistic conditions,” *Pattern Recognit. Lett.*, p. 326, 1993.

[48] A. Polesel, G. Ramponi, and V. Matthews, “Adaptive unsharp masking for contrast enhancement,” in *Proc. Int. Conf. Image Process.*, vol. 1, 1997, pp. 267–271.

[49] L. R. Rabiner, “A tutorial on hidden Markov models and selected applications in speech recognition,” *Proc. IEEE*, vol. 77, pp. 257–286, 1989.

[50] J. M. Reinhardt and W. E. Higgins, “Comparison between the morphological skeleton and morphological shape decomposition,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 18, pp. 951–957, Sept. 1996.

[51] J. Saula and M. Pietikainen, “Adaptive document image binarization,” *Pattern Recognit.*, vol. 33, no. 2, pp. 225–236, 2000.

[52] A. W. Senior and A. J. Robinson, “An off-line cursive handwriting recognition,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 20, pp. 309–322, Mar. 1998.

[53] J. Serra, “Morphological filtering: An overview,” *Signal Process.*, vol. 38, no. 1, pp. 3–11, 1994.

[54] S. Smith *et al.*, “Handwritten character classification using nearest neighbor in large databases,” *IEEE. Trans. Pattern Anal. Machine Intell.*, vol. 16, pp. 915–919, Sept. 1994.

[55] Y. Solihin and C. G. Leedham, “Integral ratio: A new class of global thresholding techniques for handwriting images,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 21, pp. 761–768, Aug. 1999.

[56] M. Sonka, V. Hlavac, and R. Boyle, *Image Processing, Analysis and Machine Vision*, 2nd ed. Pacific Grove, CA: Brooks/Cole, 1999.

[57] C. C. Tappert, “Cursive script recognition by elastic matching,” *IBM J. Res. Develop.*, vol. 26, no. 6, pp. 765–771, 1982.

[58] D. Tubbs, “A note on binary template matching,” *Pattern Recognit.*, vol. 22, no. 4, pp. 359–365, 1989.

[59] T. Watanabe, L. Qin, and N. Sugie, “Structure recognition methods for various types of documents,” *Mach. Vis. Appl.*, vol. 6, pp. 163–176, 1993.

[60] J. Yang and X. B. Li, “Boundary detection using mathematical morphology,” *Pattern Recognit. Lett.*, vol. 16, no. 12, pp. 1287–1296, 1995.

[61] B. A. Yanikoglu and P. Sandon, “Recognizing off-line cursive handwriting,” in *Proc. Int. Conf. Comput. Vision Pattern Recognit.*, Seattle, WA, 1994, pp. 397–403.

[62] F. T. Yarman-Vural and E. Ataman, “Noise histogram and cluster validity for Gaussian mixtured data,” *Pattern Recognit.*, vol. 20, no. 4, pp. 385–401, 1987.

[63] B. Yu and A. K. Jain, “A robust and fast skew detection algorithm for generic documents,” *Pattern Recognit.*, vol. 29, 1996.

[64] “GPA Truckers Guide to the Garden City Terminal”, http://www.gaports.com

[65] K. C. Hayes, “Reading handwritten words using hierarchical relaxation,” *Comput. Graph. Image Process.*, vol. 14, pp. 344–364, 1980.

[66] R. B. Hennis, “The IBM 1975 optical page reader, system design,” *IBM J. Res. Develop.*, pp. 346–353, 1968.

[67] J. Hu, S. G. Lim, and M. K. Brown, “Writer independent on-line handwriting recognition using an HMM approach,” *Pattern Recognit.*, vol. 33, no. 1, pp. 133–147, 2000.

[68] J. Hu and T. Pavlidis, “A hierarchical approach to efficient curvilinear object searching,” *Comput. Vis. Image Understanding*, vol. 63, no. 2, pp. 208–220, 1996.

[69] Y. S. Huang and C. Y. Suen, “A method of combining multiple experts for the recognition of unconstrained handwritten numerals,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 17, pp. 90–94, Jan. 1995.

[70] B. Hussain and M. Kabuka, “A novel feature recognition neural network and its application to character recognition,” *IEEE Trans Pattern Anal. Machine Intell.*, vol. 16, pp. 99–106, Jan. 1994.

[71] R. A. Jacobs *et al.*, “Adaptive mixtures of local experts,” *Neural Comput.*, vol. 3, pp. 79–87, 1991.

[72] A. K. Jain and B. Yu, “Document representation and its application to page decomposition,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 20, pp. 294–308, Mar. 1998.

[73] A. K. Jain and D. Zongker, “Representation and recognition of handwritten digits using deformable templates,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 19, pp. 1386–1391, Dec. 1997.

[74] A. K. Jain, J. Mao, and K. M. Mohiuddin, “Artificial neural networks: A tutorial,” *IEEE Comput.*, pp. 31–44, Mar. 1996.

[75] A. K. Jain and K. Karu, “Page segmentation using texture analysis,” *Pattern Recognit.*, vol. 29, pp. 743–770, 1996.

[76] A. K. Jain, R. P. W. Duin, and J. Mao, “Statistical pattern recognition : A review,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 22, pp. 4–38, Jan. 2000.

[77] B. H. Huang and L. R. Rabiner, “The segmental K-means algorithm for estimating parameters of hidden Markov models,” *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 38, pp. 1639–1641, Sept. 1990.

[78] J. Kanai and A. D. Bagdanov, “Projection profile based skew estimation algorithm for JPIG compressed images,” *Int. J. Document Anal. Recognit.*, vol. 1, no. 1, pp. 43–51, 1998.

[79] H. J. Kang and S.W. Lee, “Combining classifiers based on minimization of a Bayes error rates,” in *Proc. 5th Int. Conf. Document Anal. Recognit.*, Bangalore, India, 1999, pp. 398–401.

[80] G. Kaufmann and H. Bunke, “Detection and correction of recognition errors in check reading,” *Int. J. Document Anal. Recognit.*, vol. 2, no. 4, pp. 211–221, 2000.

[81] F. G. Keenan, L. J. Evett, and R. J. Whitrow, “A large vocabulary stochastic syntax analyzer for handwriting recognition,” in *Proc. 1st Int. Conf. Document Anal. Recognit.*, 1991, pp. 794–802.

[82] H. Y. Kim and J. H. Kim, “Handwritten Korean character recognition based on hierarchical random graph modeling,” in *Proc. Int. Workshop Frontiers Handwriting Recognit.*, 1998, pp. 577–586.

[83] W. Y. Kim and P.Yuan, “A practical pattern recognition system for translation, scale and rotation invariance,” in *Proc. Int. Conf. Comput. Vis. Pattern Recognit.*, Seattle, WA, June 1994.

[84] G. Kim and V. Govindaraju, “A lexicon driven approach to handwritten word recognition for real-time applications,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 19, pp. 366–379, Apr. 1997.

[85] G. Kim, V. Govindaraju, and S. Srihari, “Architecture for handwritten text recognition systems,” in *Proc. Int.Workshop Frontiers Handwriting Recognit.*, 1998, pp. 113–122.

[86] K. Kise, O. Yanagida, and S. Takamatsu, “Page segmentation based on thinning of background,” in *Proc 13th Int. Conf. Pattern Recognit.*, Vienna, Austria, 1996, pp. 788–792.

[87] J. Kitler, “Feature selection and extraction,” in *Handbook of Pattern Recognition and Image Processing*, T. Young and K. Fu, Eds. New York: Academic, 1986, pp. 59–83.

[88] T. Kohonen, *Self Organizing Maps*. New York: Springer, 1995, vol. 30.

[89] A. Kornai, K. M. Mohiuddin, and S. D. Connell, “An HMM-based legal amount field OCR system for checks,” in *Proc. IEEE Int. Conf. Syst., Man, Cybern., Intell. Syst. 21st Century*, vol. 3, 1995, pp. 2800–2805.

[90] , “Recognition of cursive writing on personal checks,” in *Proc. Int. Workshop Frontiers Handwriting Recognit.*, Essex, U.K., 1996, pp. 373–378.

[91] K. Kukich, “Techniques for automatically correcting words in text,” *ACM Comput. Surv.*, vol. 24, no. 4, pp. 377–439, 1992.

[92] T. Kunango, R. Haralick, and I. Phillips, “Nonlinear local and global document degradation models,” *Int. J. Imaging Syst. Technol.*, vol. 5, no. 4, pp. 274–282, 1994.

[93] A. Kundu, Y. He, and M. Y. Chen, “Alternatives to variable duration HMM in handwriting recognition,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 20, pp. 1275–1280, Nov. 1998.

[94] A. Kundu and Y. He, “On optimal order in modeling sequence of letters in words of common language as a Markov chain,” *Pattern Recognit.*, vol. 24, no. 7, pp. 603–608, 1991.

[95] L. Lam and C. Y. Suen, “Increasing experts for majority vote in OCR: Theoretical considerations and strategies,” in *Proc. Int. Workshop Frontiers Handwriting Recognit.*, 1994, pp. 245–254.

[96] , “An evaluation of parallel thinning algorithms, for character recognition,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 17, pp. 914–919, Sept. 1995.

[97] L. Lam, S. W. Lee, and C. Y. Suen, “Thinning methodologies—A comprehensive survey,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 14, pp. 869–885, Sept. 1992.

[98] E. Lecolinet, “A new model for context driven word recognition,” in *Proc. SDAIR*, 1993, pp. 135–139.

[99] E. Lecolinet and J.-P. Crettez, “A grapheme based segmentation technique for cursive script recognition,” in *Proc. 1st Int. Conf. Document Anal. Recognit.*, Saint-Malo, France, 1991, pp. 740–744.

[100] S.W. Lee, D. J. Lee, and H. S. Park, “A new methodology for gray-scale character segmentation and recognition,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 18, pp. 1045–1050, Oct. 1996.

[101] W. L. Lee and K.-C. Fan, “Document image preprocessing based on optimal Boolean filters,” *Signal Process.*, vol. 80, no. 1, pp. 45–55, 2000.

[102] S. W. Lee and Y. J. Kim, “Direct extraction of topographic features for gray-scale character recognition,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 17, pp. 724–729, July 1995.

[103] , “Multiresolutional recognition of handwritten numerals with wavelet transform and multilayer cluster neural network,” in *Proc. 3rd Int. Conf. Document Anal. Recognit.*, Montreal, QC, Canada, 1995, pp. 1010–1014.

[104] R. Legault and C. Y. Suen, “Optimal local weighted averaging methods in contour smoothing,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 18, pp. 690–706, July 1997.

[105] J. G. Leu, “Edge sharpening through ramp width reduction,” *Image Vis. Comput.*, vol. 18, no. 6–7, pp. 501–514, 2000.

[106] X. Li, W. Oh, J. Hong, and W. Gao, “Recognizing components of handwritten characters by attributed relational graphs with stable features,” in *Proc. 4th Int. Conf. Document Anal. Recognit.*, 1997, pp. 616–620.

[107] C. Y. Liou and H. C. Yang, “Handprinted character recognition based on spatial topology distance measurements,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 18, pp. 941–945, Sept. 1996.

[108] J. Liu, Y. Tang, Q. He, and C. Suen, “Adaptive document segmentation and geometric relation labeling: Algorithms and experimental results,” in *Proc 13th Int. Conf. Pattern Recognit.*, Vienna, Austria, 1996, pp. 763–767.

[109] W. Lu, Y. Ren, and C. Y. Suen, “Hierarchical attributed graph representation and recognition of handwritten chinese characters,” *Pattern Recognit.*, vol. 24, no. 7, pp. 617–632, 1991.

[110] S. Madhvanath, E. Kleinberg, V. Govindaraju, and S. N. Srihari, “The HOVER system for rapid holistic verification of off-line handwritten phrases,” in *Proc. 4th Int. Conf. Document Anal. Recognit.*, Ulm, Germany, 1997, pp. 855–890.

[111] S. Madhvanath, G. Kim, and V. Govindaraju, “Chaincode contour processing for handwrittenword recognition,” *IEEE Pattern. Anal. Machine Intell.*, vol. 21, pp. 928–932, Sept. 1999.

[112] S. Madhvanath and V. Govindaraju, “Reference lines for holistic recognition of handwritten words,” *Pattern Recognit.*, vol. 32, no. 12, pp. 2021–2028, 1999.

[113] S. A. Mahmoud, I. Abuhabia, and R. J. Green, “Skeletonization of Arabic characters using clustering based skeletonization algorithm,” *Pattern Recognit.*, vol. 24, no. 5, pp. 453–464, 1991.

[114] J. Mantas, “An overview of character recognition methodologies,” *Pattern Recognit.*, vol. 19, no. 6, pp. 425–430, 1986.

[115] J. Mao, K. M. Mohiuddin, and T. Fujisaki, “A two stage multi-network OCR system with a soft pre-classifier and a network selector,” in *Proc. 3rd Int. Conf. Document Analysis and Recognit.*, Montreal, QC, Canada, 1995, pp. 821–828.

[116] A. Meyer, “Pen computing: A technology overview and a vision,” *SIGCHI Bull.*, vol. 27, no. 3, pp. 46–90, 1995.

[117] M. L. Minsky and S. Papert, *Perceptrons: An Introduction to Computational Geometry*. Cambridge, MA: MIT Press, 1969.

[118] K. T. Miura, R. Sato, and S. Mori, “A method of extracting curvature features and its application to handwritten character recognition,” in *Proc. 4th Int. Conf. Document Anal. Recognit.*, Ulm, Germany, 1997, pp. 450–454.

[119] S. Mo and V. J. Mathews, “Adaptive, quadratic preprocessing of document images for binarization,” *IEEE Trans. Image Processing*, vol. 7, pp. 992–999, July 1998.

[120] M. A. Mohamed and P. Gader, “Generalized hidden Markov models—Part II: Application to handwritten word recognition,” *IEEE Trans. Fuzzy Syst.*, vol. 8, pp. 82–95, Feb. 2000.

[121] , “Handwritten word recognition using segmentation-free hidden Markov modeling and segmentation based dynamic programming techniques,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 18, pp. 548–554, May 1996.

[122] K. M. Mohiuddin and J. Mao, “A comparative study of different classifiers for handprinted character recognition,” *Pattern Recognit. Practice IV*, pp. 437–448, 1994.

[123] S. Mori, C. Y. Suen, and K. Yamamoto, “Historical review of OCR research and development,” *Proc. IEEE*, vol. 80, pp. 1029–1057, July 1992.

[124] S. Mori, H. Nishida, and H. Yamada, *Optical Character Recognition*. New York: Wiley, 1999.

[125] S. Mori, K. Yamamoto, and M. Yasuda, “Research on machine recognition of handprinted characters,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-6, pp. 386–404, Apr. 1984.

[126] M. Nakagava, T. Oguni, and A. Homma, “A coarse classification of on-line handwritten characters,” in *Proc. 5th Int. Workshop Frontiers Handwriting Recognit.*, Essex, U.K., 1996, pp. 417–420.

[127] M. Nakagawa and K. Akiyama, “A linear-time elastic matching for stroke number free recognition of on-line handwritten characters,” in *Proc. Int. Workshop Frontiers Handwriting Recognit.*, Taiwan, R.O.C., 1994, pp. 48–56.

[128] W. Niblack, *An Introduction to Digital Image Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1986.

[129] L. O’Gorman, “The document spectrum for page layout analysis,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 15, pp. 162–173, 1993.

[130] I. S. Oh and C. Y. Suen, “Distance features for neural network-based recognition of handwritten characters,” *Int. J. Document Anal. Recognit.*, vol. 1, no. 2, pp. 73–88, 1998.

[131] I. S. Oh, J. S. Lee, and C. Y. Suen, “Analysis of class separation and combination of class-dependent features for handwriting recognition,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 21, pp. 1089–1094, Oct. 1999.

[132] I. S. Oh *et al.*, “Class-expert approach to unconstrained handwritten numeral recognition,” in *Proc. 5th Int. Workshop Frontiers Handwriting Recognit.*, Essex, U.K., 1996, pp. 95–102.

[133] M. Okamoto and K. Yamamoto, “On-line handwriting character recognition method with directional features and direction change features,” in *Proc. 4th Int. Conf. Document Anal. Recognit.*, Ulm, Germany, 1997, pp. 926–930.

[134] E. Oztop *et al.*, “Repulsive attractive network for baseline extraction on document images,” *Signal Process.*, vol. 74, no. 1, 1999.

[135] M. Parizeau and R. Plamondon, “A fuzzy-syntactic approach to allograph modeling for cursive script recognition,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 17, pp. 702–712, July 1995.

[136] J. Park, V. Govindaraju, and S. N. Srihari, “OCR in a hierarchical feature space,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 22, pp. 400–407, Apr. 2000.

[137] T. Pavlidis, “Recognition of printed text under realistic conditions,” *Pattern Recognit. Lett.*, p. 326, 1993.

[138] T. Pavlidis and J. Zhou, “Page segmentation by white streams,” in *Proc. 1st Int. Conf. Document Anal. Recognit.*, Saint-Malo, France, 1991, pp. 945–953.

[139] L. F. C. Pessoa and P. Maragos, “Neural networks with hybrid morphological/ rank/linear nodes: A unifying framework with applications to handwritten character recognition,” *Pattern Recognit.*, vol. 33, pp. 945–960, 2000.

[140] R. Plamandon *et al.*, “On-line handwriting recognition,” in *Encyclopedia of Electrical and Electronics Engineering*, J. G. Webster *et al.*, Eds. New York: Wiley, 1999, vol. 15, pp. 123–146.

[141] R. Plamandon and S. N. Srihari, “On-line and off-line handwriting recognition: A comprehensive survey,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 22, pp. 63–85, Jan. 2000.

[142] A. Polesel, G. Ramponi, and V. Matthews, “Adaptive unsharp masking for contrast enhancement,” in *Proc. Int. Conf. Image Process.*, vol. 1, 1997, pp. 267–271.

[143] L. Prevost and M. Milgram, “Automatic allograph selection and multiple expert classification for totally unconstrained handwritten character recognition,” in *Proc. 14th Int. Conf. Pattern Recognit.*, Brisbane, Australia, 1998, pp. 381–383.

[144] K. E. Price, “Relaxation matching techniques comparison,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-7, pp. 617–623, May 1985.

[145] L. R. Rabiner, “A tutorial on hidden Markov models and selected applications in speech recognition,” *Proc. IEEE*, vol. 77, pp. 257–286, 1989.

[146] N. V. S. Reddy and P. Nagabhushan, “A three dimensional neural network model for unconstrained handwritten numeral recognition: A new approach,” *Pattern Recognit.*, vol. 31, no. 5, pp. 511–516, 1998.

[147] J. M. Reinhardt and W. E. Higgins, “Comparison between the morphological skeleton and morphological shape decomposition,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 18, pp. 951–957, Sept. 1996.

[148] B. Ripley, “Statistical aspects of neural networks,” in *Networks on Chaos: Statistical and Probabilistic Aspects* , U. Bornndorff-Nielsen, J. Jensen, and W. Kendal, Eds. London, U.K.: Chapman & Hall, 1993.

[149] J. Rocha and T. Pavlidis, “A shape analysis model,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 16, pp. 394–404, Apr. 1994.

[150] , “Character recognition without segmentation,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 17, pp. 903–909, Sept. 1995.

[151] J. Saula and M. Pietikainen, “Adaptive document image binarization,” *Pattern Recognit.*, vol. 33, no. 2, pp. 225–236, 2000.

[152] A. W. Senior and A. J. Robinson, “An off-line cursive handwriting recognition,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 20, pp. 309–322, Mar. 1998.

[153] J. Serra, “Morphological filtering: An overview,” *Signal Process.*, vol. 38, no. 1, pp. 3–11, 1994.

[154] T. Shioyama, H. Y. Wu, and T. Nojima, “Recognition algorithm based on wavelet transform for handprinted Chinese characters,” in *Proc. 14th Int. Conf. Pattern Recognit.*, vol. 1, 1998, pp. 229–232.

[155] M. Shridhar and A. Badreldin, “High accuracy syntactic recognition algorithm for handwritten numerals,” *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-15, pp. 152–158, Jan. 1985.

[156] M. Shridhar and F. Kimura, “Handwritten numerical recognition based on multiple algorithms,” *Pattern Recognit.*, vol. 24, no. 10, pp. 969–983, 1991.

[157] J. C. Simon, “Off-line cursive word recognition,” *Proc. IEEE*, vol. 80, p. 1150, July 1992.

[158] R. M. K. Sinha, B. Prasada, G. F. Houle, and M. Babourin, “Hybrid contextual text recognition with string matching,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 15, pp. 915–925, Sept. 1993.

[159] S. Smith *et al.*, “Handwritten character classification using nearest neighbor in large databases,” *IEEE. Trans. Pattern Anal. Machine Intell.*, vol. 16, pp. 915–919, Sept. 1994.

[160] Y. Solihin and C. G. Leedham, “Integral ratio: A new class of global thresholding techniques for handwriting images,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 21, pp. 761–768, Aug. 1999.

[161] M. Sonka, V. Hlavac, and R. Boyle, *Image Processing, Analysis and Machine Vision*, 2nd ed. Pacific Grove, CA: Brooks/Cole, 1999.

[162] T. Steinherz, E. Rivlin, and N. Intrator, “Off-line cursive script word recognition—A survey,” *Int. J. Document Anal. Recognit.*, vol. 2, no. 2, pp. 90–110, 1999.

[163] C. Y. Suen, C. C. Tappert, and T. Wakahara, “The state of the art in on-line handwriting recognition,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 12, pp. 787–808, Aug. 1990.

[164] C. Y. Suen, M. Berthod, and S. Mori, “Automatic recognition of handprinted characters—The state of the art,” *Proc. IEEE*, vol. 68, pp. 469–487, Apr. 1980.

[165] H. Takahashi, “A neural net OCR using geometrical and zonal-pattern features,” in *Proc. 1st Int. Conf. Document Anal. Recognit.*, 1991.

[166] Y. Tang *et al.*, “A new approach to document analysis based on modified fractal signature,” in *Proc. 3rd Int. Conf. Document Anal. Recognit.*, Montreal, QC, Canada, 1995, pp. 567–570.

[167] Y. Tang *et al.*, “Off-line recognition of Chinese handwriting by multifeature and multilevel classification,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 20, pp. 556–561, May 1998.

[168] Y. Tao and Y. Y. Tang, “The feature extraction of Chinese character based on contour information,” in *Proc. 5th Int. Conf. Document Anal. Recognit.*, Bangalore, India, 1999, pp. 637–640.

[169] C. C. Tappert, “Cursive script recognition by elastic matching,” *IBM J.Res. Develop.*, vol. 26, no. 6, pp. 765–771, 1982.

[170] Q. Tian *et al.*, “Survey: Omnifont printed character recognition,” *Visual Commun. Image Process: Image Process.*, pp. 260–268, 1991.

[171] Ø. D. Trier and A. K. Jain, “Goal directed evaluation of binarization methods,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 17, pp. 1191–1201, Dec. 1995.

[172] Ø. D. Trier, A. K. Jain, and T. Taxt, “Feature extraction method for character recognition—A survey,” *Pattern Recognit.*, vol. 29, no. 4, pp. 641–662, 1996.

[173] W. H. Tsai and K. S. Fu, “Attributed grammar—A tool for combining syntactic and statistical approaches to pattern recognition,” *IEEE Syst. Man, Cybern.*, vol. SMC-10, pp. 873–885, Dec. 1980.

[174] S. Tsujimoto and H. Asada, “Major components of a complete text reading system,” *Proc. IEEE*, vol. 80, p. 1133, July 1992.

[175] D. Tubbs, “A note on binary template matching,” *Pattern Recognit.*, vol. 22, no. 4, pp. 359–365, 1989.

[176] J. Wang and J. Jean, “Segmentation of merged characters by neural networks and shortest path,” *Pattern Recognit.*, vol. 27, no. 5, p. 649, 1994.

[177] L. Wang and J. Mendel, “A fuzzy approach to handwritten rotation invariant character recognition,” *Comput. Mag.*, vol. 3, pp. 145–148, 1992.

[178] S. S.Wang, P. C. Chen, andW. G. Lin, “Invariant pattern recognition by moment Fourier descriptor,” *Pattern Recognit.*, vol. 27, pp. 1735–1742, 1994.

[179] K. Wang, Y. Y. Tang, and C. Y. Suen, “Multi-layer projections for the classification of similar Chinese characters,” in *Proc. 9th Int. Conf. Pattern Recognit.*, 1988, pp. 842–844.

[180] T. Watanabe, L. Qin, and N. Sugie, “Structure recognition methods for various types of documents,” *Mach. Vis. Appl.*, vol. 6, pp. 163–176, 1993.

[181] D. H. Wolperk, “Stacked generalization,” *Neural Networks*, vol. 5, pp. 241–259, 1992.

[182] S. L. Xie and M. Suk, “On machine recognition of hand-printed Chinese characters by feature relaxation,” *Pattern Recognit.*, vol. 21, no. 1, pp. 1–7, 1988.

[183] L. Xu, A. Krzyzak, and C. Y. Suen, “Methods of combining multiple classifiers and their application to handwritten recognition,” *IEEE Trans. Syst., Man, Cybern.*, vol. 22, pp. 418–435, Mar. 1992.

[184] Y. Yang and H. Yan, “An adaptive logical method for binarization of degraded document images,” *Pattern Recognit.*, vol. 33, no. 5, pp. 787–807, 2000.

[185] J. Yang and X. B. Li, “Boundary detection using mathematical morphology,” *Pattern Recognit. Lett.*, vol. 16, no. 12, pp. 1287–1296, 1995.

[186] B. A. Yanikoglu and P. Sandon, “Recognizing off-line cursive handwriting,” in *Proc. Int. Conf. Comput. Vision Pattern Recognit.*, Seattle, WA, 1994, pp. 397–403.

[187] F. T. Yarman-Vural and A. Atici, “A segmentation and feature extraction algorithm for Ottoman cursive script,” in *Proc. 3rd Turkish Symp. Artif. Intell. Neural Networks*, 1994.

[188] F. T. Yarman-Vural and E. Ataman, “Noise histogram and cluster validity for Gaussian mixtured data,” *Pattern Recognit.*, vol. 20, no. 4, pp. 385–401, 1987.

[189] B. Yu and A. K. Jain, “A robust and fast skew detection algorithm for generic documents,” *Pattern Recognit.*, vol. 29, 1996.