VOL 102 No 1
March 2011

SAIEE Africa Research Journal

CAT-7 CABLE EVALUATION FOR SQUARE KILOMETRE ARRAY ANALOGUE SIGNAL TRANSPORT

P.K. Langat*, P van der Merwe*, T Ikin, and H C Reader***

* *Department of Electrical and Electronic Engineering, University of Stellenbosch, 7600 Stellenbosch, South Africa E-mail: hcreader@sun.ac.za*

** *Jodrell Bank Observatory, University of Manchester, Macclesfield, Cheshire, UK*

Abstract: Data-cable systems are essential for an evolving Square Kilometre Array (SKA) phased-array demonstrator project. Loss, dispersion and channel coupling are the characteristics of interest. We evaluate a Category 7 (CAT-7) cable system, which incorporates a 20 m CAT-7 cable, balun transformers, and ARJ45 connectors. Measurements from 30 MHz - 1.5 GHz are made using a calibrated Vector Network Analyzer (VNA). Through, Reflect, Line (TRL) as well as Short, Open, matched Load, Through (SOLT) calibration techniques were performed, which separately test the cables and the entire system performance respectively. A further measurement isolates the baluns. The connectors and baluns are mounted on customised printed circuit boards. We investigated phase dispersion, return loss, attenuation, near-end crosstalk (NEXT) and far-end crosstalk (FEXT). The best system was found to have a linear phase response in the 30 MHz to 1.4 GHz band, with NEXT and FEXT levels below -50 dB and -63 dB respectively. Such performance makes this cable class a candidate for high-volume data streams anticipated for the SKA.

Keywords: Baluns, calibration, connectors, CAT-7 cables, EMC, far-end crosstalk, near-end crosstalk, Square Kilometre Array.

1. INTRODUCTION

The Square Kilometre Array (SKA) [1] is the next generation of radio telescope aiming to attain one million square metres of effective collecting area in a frequency range of 0.1-25 GHz. It will have large instantaneous fields-of-view and two orders of magnitude greater sensitivity than the current telescopes. Although the technology definition continues to evolve, the SKA's system design proposes the use of phased aperture array technology as the primary collector type for the frequencies below 1 GHz. The 12-15m dish antennas will be used for the higher frequencies. Many analogue links will be used to connect these collector systems to an electromagnetically screened station processing area (bunker), which mitigates self-induced radio frequency interference (RFI).

Two signal transmission technologies have been proposed [2]. Optical fibre could be used as the high frequency link from the dishes. Category 7 (CAT-7) networking cables, which use ARJ45 connectors, could transfer the mid and low-frequency data from the aperture array elements to the processing nodes in the bunker. The CAT-7 cable shown in Fig. 1, designed to support the 10 Gb/s Ethernet applications, has a metal braided shield over four twisted pairs, which are individually screened with metallic foil. This two-tier screening configuration is used to minimize crosstalk in the cable thus reducing RFI coupling and/or leakage. These cables are used in conjunction with the ARJ45 connectors [2] (Fig. 2) and are designed to meet the CAT-7 performance specifications given by the IEC 11801 [3] cable standard and the IEC 60603-7-7 connector standard [4].

The conductor pairs provide inherently balanced performance, reducing the cable's susceptibility to electromagnetic interference (EMI). The antennas and equipment connecting to the cables are coaxially-based, so balun transformers are needed between the ARJ45 and coaxial connectors. We refer to the CAT-7 cable, ARJ45 connector, and the baluns as the CAT-7 cable system.

Due to the large number of connections anticipated in the SKA system, cost-effective solutions are critical, hence the interest in CAT-7. However, because of the SKA RF sensitivity, the cable system must maintain good analogue signal integrity, with bandwidth and non-dispersive transmission being important. For full electromagnetic compatibility (EMC), attention must also be given to low emission and crosstalk. This paper presents the measured performance of two selected commercial CAT-7 cable systems.

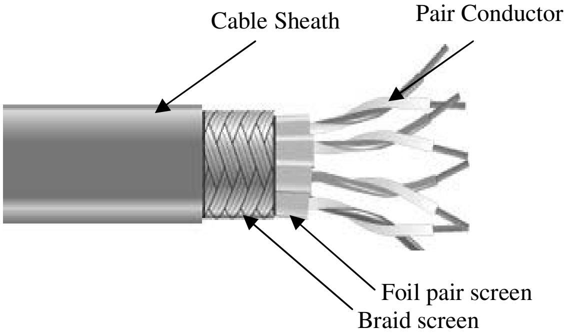


Figure 1: CAT-7 Cable structure

Figure 2: The ARJ45 connector plug and jack.

2. EXPERIMENTAL SETUP

The CAT-7 cable system we tested included 20 m CAT-7 cable lengths, ADTL2-18 balun transformers, and ARJ45 connectors. Two types of cables were made available which had solid and stranded cores respectively. Many independent measurements have been made on these cables and repeatable results have been achieved. We present our findings on a representative set of each only. The two cable types are identified as 1AB and 1AO, where the former has the solid core.

Figure 3 shows the measurement and cable system setup. As can be seen from Figure 3, the VNA ports numbered N1 and N2 are connected to the reference ports *i* and *ii* through phase-stable semi-rigid coaxial cables. The latter ports are connected to the sub-miniature version A (SMA) connectors A-D, which will allow measurements for each individual pair of the four-pair cables. The wideband ADTL2-18 baluns are used to enable the network analyzer, which is a coaxial single-ended system, to measure the characteristics of the balanced pairs of the cables. These 50:100 Ω baluns, B1-B4 in Figure 3, are mounted on two identical printed circuit boards (PCB1 and PCB2). They are connected via short 100 Ω strip-lines to the SMA connectors A-D and the ARJ45 connectors C1 and C2. The CAT-7 cable under test is connected between the connectors C1 and C2.

The VNA was switched on for an hour prior to calibration to ensure thermal stability and to reduce drift. All connectors were cleaned with n-butyl acetate, dried with compressed air, and tightened with a calibrated torque wrench.

3. CALIBRATION

To examine the CAT-7 cable system and the cables on their own, using the VNA, both SOLT and TRL calibration schemes were performed respectively. The transmission response, return loss, NEXT and FEXT are the performance metrics measured and are presented here in decibels. The transmission response is the attenuation in the cable system measured as an S_{mn} parameter on the same cable pair near-end (port n) to far-end (port m). It is both length and frequency dependent and should meet the requirements (given as insertion loss, IL) derived by the following equation [3] for each pair in category 7 with two connections:

$$IL_k = \frac{L}{100}\left(1.8\sqrt{f} + 0.01f + \frac{0.2}{\sqrt{f}}\right) + 0.04\sqrt{f} \qquad (1)$$

where k is the number of pairs being measured, L is the length of the cable in metres, and f is the frequency in MHz.

NEXT is a S_{mn} coupled interference signal between adjacent cable pairs at the near-end of the transmitting source [5]. The maximum NEXT limit for each pair in category 7, as specified in [3], is given by:

$$-20log_{10}\left(10^{\frac{-(102.4-15\log(f))}{20}} + 10^{\frac{-(102.4-15\log(f))}{20}}\right) \qquad (2)$$

where the first part of (2) is the NEXT loss for the pair-to-pair cable and the second part is that of the two connectors.

FEXT is the S_{mn} coupled interference between one pair on the near-end and another pair on the far-end of the cable. Since FEXT measurement is attenuation dependent, and thus cable length dependent, this crosstalk is represented as Equal Level Far-End Crosstalk (ELFEXT) where the attenuation is removed. According to [3], ELFEXT$_{ik}$ of pairs *i* and *k* is computed as:

$$ELFEXT_{ik} = FEXT_{ik} - IL_k \qquad (3)$$

where, i is the number of the disturbed pair, k is the number of the disturbing pair, FEXT$_{ik}$ is the far-end crosstalk loss coupled from pair *i* into pair *k* and IL$_k$ is the insertion loss of pair *k*. The ELFEXT of each pair combination in CAT 7 with two connections will meet the requirements derived by the following equation given in [3]:

$$ELFEXT_{ik} = -20log_{10}\left[10^{-\left(\frac{94-20log(f)}{20}\right)} + 2 * 10^{-\left(\frac{90-15log(f)}{20}\right)}\right] \qquad (4)$$

The return loss is a measure of the reflection at the reference port caused by the impedance mismatches of the components in the cable system. This loss will, according to [3], meet the requirements derived by the following equation:

$$RL = 34 - 10log_{10}(f);\ \text{for } f \geq 40MHz \qquad (5)$$

Both NEXT and FEXT were measured with one pair (orange) of conductors in the cable taken as the source of coupled interference signal and the other three pairs as the victims. Each cable has orange, brown, green, and blue pairs of conductors. The unused ports on the printed circuit boards were terminated with 50 Ω SMA matched loads.

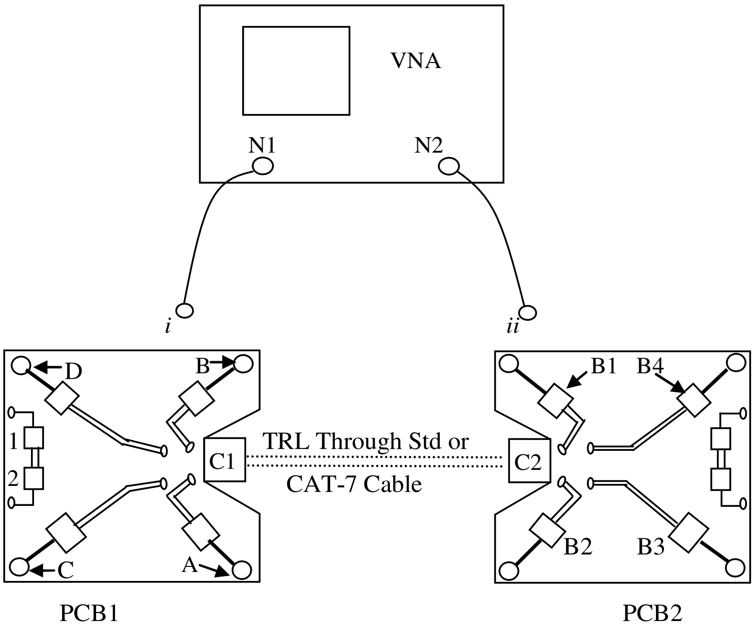


Figure 3: The VNA and cable system measurement setup with reference ports identified.

3.1 SOLT calibration

An APC-3.5 calibration kit was used to perform a full two-port SOLT calibration whose reference plane was set at ports *i* and *ii* in Fig. 3. This calibration was used to make measurements of the whole cable system (from SMA-SMA input connectors on the PCBs) by taking 201 equally spaced points from 45 MHz-1.5 GHz. We set the averaging factor to 128 and reduced the Intermediate Frequency (IF) bandwidth to 100 Hz so as to extend the dynamic range.

3.2 TRL calibration

The TRL calibration sets reference planes at the ARJ45 connectors C1 and C2 as shown in Fig. 3. Through, Reflect and Line standards, which we constructed from lengths of the 1AO cable, were used. The Through and the Line standards were terminated with ARJ45 plugs at both ends whereas the Reflect was terminated with a plug on one end and left open on the other. The Through standard was 94.5 mm long and the Reflect was made to be half of this length. The two Line lengths were also made to be 519 mm and 68.5 mm longer than the Through length respectively. The first Line was defined for the lower frequency range (30-200 MHz) and the second line for the higher frequencies up to 1.5 GHz. The electrical lengths of these standards were determined using a SOLT procedure. Their time delay values were inserted into the definition of our custom TRL calibration kit. From this procedure, the performance of the CAT-7 cable alone was measured from 30 MHz-1.5 GHz, as the PCB connectors and the baluns are effectively calibrated out.

4. MEASURED RESULTS AND DISCUSSION

In the measurements that follow, the ISO/IEC 11801 [3] cable standard class F or F_A limit lines are plotted for reference in some instances. The empirical formulas describing these lines, given in section 3 above, extend to 600 MHz (indicated as the class F consolidation point) and 1000 MHz (class F_A, given only for channels). The interest here reaches to 1.5 GHz, which exceeds existing CAT-7 bandwidth specifications. The SKA systems will need to extract maximum performance in most applications.

4.1 SOLT Calibration for System Evaluation

Immediately after the SOLT calibration, the S_{11} for the re-connected Through standard was better than -48.5 dB across the band. This gives an indication of the achievable dynamic range of subsequent reflection measurements. Evaluating S_{21} with the Through standard connected showed little loss, being at most 0.03 dB at the highest frequencies. The transmission dynamic range was found to be better than -100 dB across most of the given frequency range, which is appropriate for our measurements.

Balun performance: The performance of the baluns alone was measured using baluns 1 and 2, which are the extra back-to-back connected baluns on PCB1 and PCB2 respectively. Fig. 4 shows the S_{11} for both baluns which is higher than -10 dB for frequencies between 260-600

MHz in this configuration. This performance improves if there is a reasonable length of cable between the baluns. The reflection responses are similar since the baluns are of the same type (ADTL2-18).

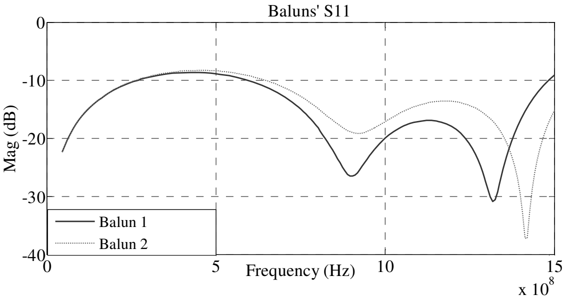


Figure 4: Comparison of back-to-back balun reflection coefficients (dB) on PCB boards (ports 1 to 2).

Figure 5 shows the S_{21} response of the back-to-back baluns, which are very close, showing a maximum transmission loss of 2.7 dB at higher frequencies. The baluns therefore work satisfactorily over the selected bandwidth of up to 1.5 GHz.

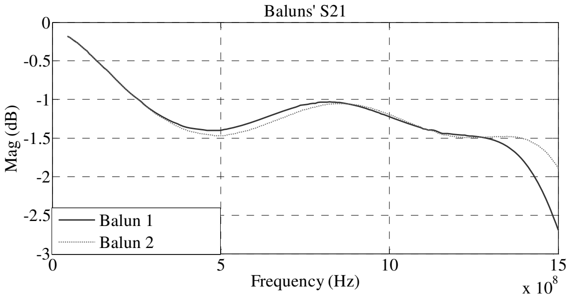


Figure 5: Comparison of back-to-back balun S_{21} (dB) on PCB boards.

Cable system performance: The SOLT calibration takes the reference plane to the SMA connectors on boards PCB1 and PCB2. These entire system measurements incorporate the cable properties along with the SMA connectors, balanced striplines, baluns and ARJ45 plugs.

Figure 6 shows the SOLT S_{11} cable system response with the two 20 m cables. The equivalent IEC 11801 Class F standard for terminated cables, and the short Through cable (94.5 mm long) response, are included for comparison. The 1AO cable system has a return loss better than 10 dB in its operational band. Both the 1AO and the short Through are better than 15 dB between 700-1200 MHz. This specific 700-1200 MHz response is not seen during the TRL cable measurements, and is attributed to the components on the PCB. The following connections show increasing S_{11} improvement in the order presented.

- balun-balun;
- SMA connectors, board, short cable (TRL's Through standard), to board, SMA connectors;
- SMA connectors, board, long cable, to board, SMA connectors;

This indicates that the balun is best loaded by a cable of some length which presents a better matched 100 Ω load. The 1AO cable clearly has the better in-band S_{11} response.

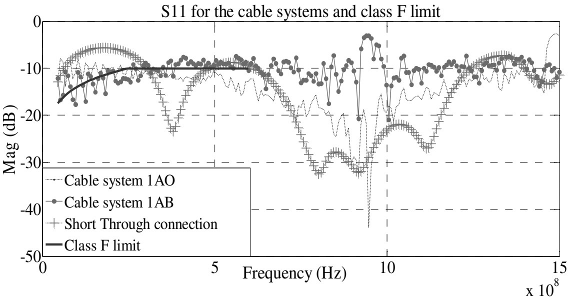


Figure 6: SOLT measured S_{11} for the cable systems 1AO and 1AB with Class F standard and short THROUGH connection for comparison.

Figure 7 shows the transmission response for the two cable systems with Class F cable limit lines for both solid and stranded cores. It can be seen that the cable system 1AB (solid-core) is bandwidth limited due to the sharp resonance at 900 MHz whereas cable system 1AO (stranded-core) has a higher limit at 1.4 GHz. The 1AO stranded cable system both meets its S_{21} standard and extends beyond the expected bandwidth.

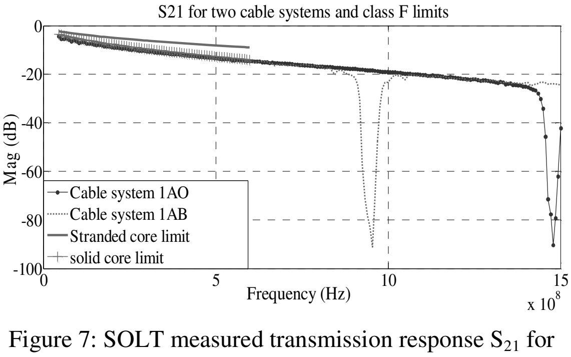


Figure 7: SOLT measured transmission response S_{21} for the two cable systems and the Class F standards for solid and stranded cables.

Cable manufacturers put a specific number of twists per unit distance into each cable pair. The twist ratios are different for each pair to minimize constructive coupling. However, there is a periodicity about the twist sequence which leads to resonances. This, in principle, could be optimized to obtain the widest operational bandwidth.

Figure 8 gives the detrended form of the phase responses for the two cable systems. This shows that the responses are linear over the depicted bandwidths. These results cannot be compared to a standard as none is available, but it is of importance for SKA digital signal transport, where in-band dispersion would be unacceptable.

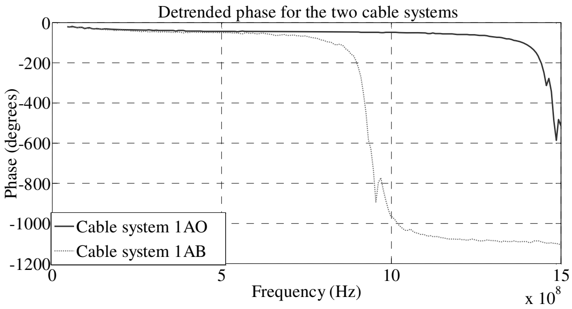
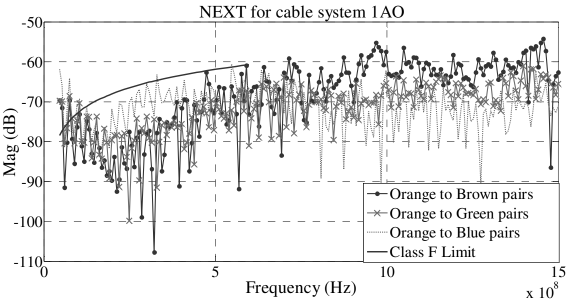


Figure 8: SOLT measured detrended phase responses for the two cable systems.

The NEXT measurements were taken for the cable system 1AO and the results are given in Fig. 9. The NEXT, from the orange pair to the other three pairs, is at most -55 dB. In-band, this only occurs around 1 GHz, well beyond the Class F specification. We also measured the cable system 1AB NEXT and found it to be much closer to the Class F limit, but do not present the result here.



cable system 1AO along with equivalent Class F limit.

The FEXT measurements for the 1A0 cable system are given in Fig. 10. The far-end cross coupling is better than -60 dB for cable 1AO across the given frequency range. The 1AB cable system FEXT was additionally tested and performs best here, being lower than – 65 dB in its band, although in every other respect its performance does not match that of the 1AO system. These results are significantly better than the Class F requirements in the frequency range up to 600 MHz.

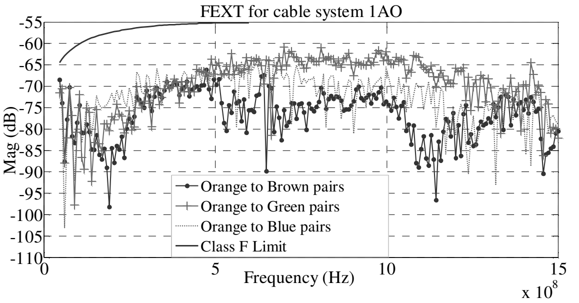


Figure 10: SOLT measured FEXT for the cable system 1AO along with equivalent Class F limit.

4.2 *TRL calibration for cable evaluation*

As discussed in section 3.2, it is possible to isolate the behaviour of the cables on its own through the TRL procedures. After performing the TRL calibration, the S-parameters for the Through standard were observed to determine the system's dynamic range available. The S_{11} was much better than -10 dB and the S_{21} was close to 0 dB until 1.3 GHz where resonance is seen. This is attributed to the limitation of the second line used for calibration since it was made from the 1AO cable. The transmission dynamic range was found to be better than -95 dB at lower frequencies and -70 dB at higher frequencies and any measurements above this noise floor are considered to be valid.

The cables' S_{11} responses, given in Fig. 11, show that cable 1AO has a lower reflection at frequencies below 1.2 GHz compared to cable 1AB. The high S_{11} at around 900 MHz for cable 1AB manifests in Fig. 12 as a substantial transmission loss, confirming the limit of these cables. Cable 1AO has a transmission frequency limit of 1.4 GHz as shown from S_{21} in Fig. 12. These results are similar to those obtained for the SOLT calibration in Fig. 7. The attenuation is less than 20 dB up to 1.4 GHz for stranded-core 1AO cable.

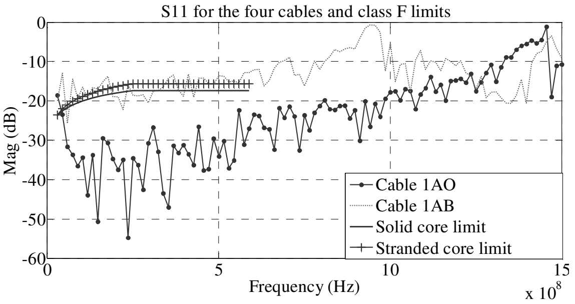


Figure 11: S_{11} for the cables 1AO and 1AB with TRL calibration.

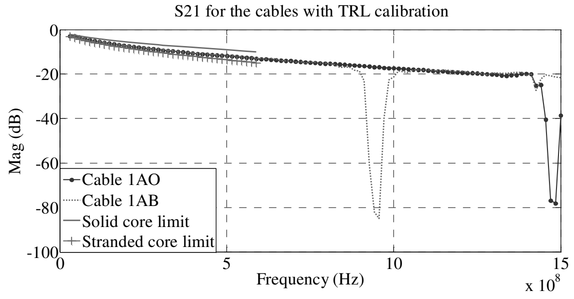


Figure 12: The four cables' S_{21} with TRL calibration along with equivalent Class F limit.

The NEXT for cable 1AO is less than -60 dB below 0.6 GHz, easily meeting the specified limit for the Class F. Equivalent results were obtained for cable 1AB. These results also compare well with those obtained using the SOLT scheme implying that the other PCB components of the cable system do not significantly contribute to the amount of crosstalk.

The FEXT for cable 1AO, given in Fig. 14, is less than -63 dB everywhere. Cable 1AB similarly showed lower levels. The SOLT technique yields comparable levels and these are well below the limits.

4.3 Discussion

The measured cable and the entire cable system performance have been presented. We gave more attention to the higher-performing 1AO cabling as it became clear that the 1AB system was band-limited to 900 MHz. The in-house TRL standards, using the identifiably better 1AO cable, produced a calibration scheme with sufficient dynamic range to test the cable characteristics to their operational limits. The line TRL standard lengths were originally chosen with a 1.5 GHz measurement bandwidth in mind. In the event, this range exceeded the frequency limit of the better cable class by a small margin.

5. CONCLUSION

The results for the CAT-7 system with the two cable types (solid-core and stranded-patch cables) have been evaluated using both SOLT and TRL calibrations. The performance of the baluns, and then the entire cable system, were examined with the SOLT calibration. The cables' performance alone was characterised using the TRL method.

The system performance is not significantly degraded by the inclusion of the SMA connectors, baluns, PCB tracks and ARJ45 jack. With respect to analogue signal integrity, the reflection and transmission results show that the solid-core cables work up to 0.9 GHz, and the stranded-patch cables up to 1.4 GHz. This is related to periodicity of twist ratios in cable pairs. Both cable types showed low-dispersion phase responses.

Not surprisingly the commercial cable systems meet the specified limits for the Class F standards. However, in terms of EMC performance for anticipated sensitive SKA applications, our measurements show that at least one cable type performs substantially better. For the 1AO cable alone, NEXT and FEXT are found to be below -50 dB and -63 dB respectively. Coupled with the phase linearity and acceptable transmission loss up to 1.4 GHz, the SKA system designers could consider this cable class as a candidate for the high-volume data streams anticipated. Final cable choice would have to include such factors as phase stability, total cost of deployment and durability.

6. ACKNOWLEDGEMENTS

The authors would like to thank Yakov Belopolsky of Bel Stewart Connector for providing the cables and connectors for measurements and also for helping to make the TRL standards. Andrew Faulkner is thanked for originating the collaboration between the authors and for commenting on the paper. The SKA (South Africa) Postgraduate Bursary programme is thanked for PhD bursaries for Philip Kibet Langat and Paul van der Merwe. The South African Bureau of Standards is acknowledged for supplying cable and connector standards.

7. REFERENCES

[1] P. J. Hall, "The Square Kilometre Array: An international engineering perspective," *Experimental Astronomy*, vol. 17, no. 1–3, pp. 5–16, Jun. 2004

[2] R. Bolton *et al.*, "SKADS Benchmark Scenario Design and Costing – 2," *SKADS Memo Series*, Oct. 2008.

[3] International Organization for Standardization, *ISO/IEC 11801- Information technology — Generic cabling for customer premises,* Edition 2.1, Jul. 2008.

[4] International Electrotechnical Commission, IEC 60603-7-7- *Connectors for electronic equipment — Detail specification for 8-way, shielded, free and fixed connectors, for data transmissions with frequencies up to 600 MHz*, Part 7-7, 2nd edition, Jun. 2006.

[5] P. Kirawanich, J. R. Wilson, and N. E. Islam, "Minimizing crosstalks in unshielded twisted-pair cables by using electromagnetic topology techniques," *Progress In Electromagnetics Research*, PIER 63, pp. 125–140, 2006

IDENTIFYING OPPORTUNITIES FOR DETERMINISTIC NETWORK CODING IN WIRELESS MESH NETWORKS

M.J. Grobler[*] **and A.S.J. Helberg**[*]

[*] *School for Electrical, Electronic and Computer Engineering, North-West University, Private Bag X6001, Potchefstroom South Africa, E-mail: leenta.grobler@nwu.ac.za, albert.helberg@nwu.ac.za*

Abstract: Recent advances in methods to improve network utilisation has lead to the introduction of Network Coding, a technique that can reduce local congestion in a network by combining information sent over the network. In this paper the use of deterministic Network Coding in a Wireless Mesh Network (WMN) is proposed. A method to determine where Network Coding can be implemented in a WMN is presented. It is shown that the inherent properties of WMNs provide good opportunities for the implementation of this method

Key words: Capacity, Network Coding, Network Topology, Network Utilisation, Quality of Service, Wireless Mesh Networks.

1. INTRODUCTION

The universal need for better control over resources in communication networks is a problem that is studied continuously. The network's maximum capacity needs to be defined and then utilised to ensure that as much information as possible is delivered in the most beneficial manner. One way in which this can be achieved, is by implementing a relatively new method called Network Coding. Research on Network Coding to date has lead to a wide variety of theoretical results, mainly applicable to wired networks, where Network Coding is implemented deterministically.

The capacity and resource control problem in wireless networks also needs to be addressed, but the identification of Network Coding opportunities is a bit more complicated, especially in the case of Wireless Mesh Networks (WMNs). WMNs are complex networks of which the topology may change unpredictably due to nodes joining and leaving the network, or moving around (in the case of a mobile ad-hoc network). One documented implementation of Network Coding in wireless networks is Random Network Coding [1]. Deterministic Network Coding however holds a few advantages over Random Network Coding, which will be discussed later in this paper and therefore it would be advantageous to find a way to implement Deterministic Network Coding in wireless networks.

In this paper, a method for the identification of opportunities for the implementation of deterministic Network Coding is presented. Further it is shown that the inherent properties of WMNs provide good opportunities for the implementation of this method.

The remainder of this paper is structured as follows: An introduction to WMNs is given in section 2. The maximum throughput capacity of a network in terms of the Min-cut Max-flow theorem is briefly discussed in section 3. Network Coding and the advantages it holds are presented in section 4. A new method to identify opportunities for the implementation of deterministic Network Coding in WMNs is presented in section 5. This method is evaluated in section 6. The impact on throughput capacity of WMNs is briefly discussed in section7. Finally, a conclusion is drawn in section 8.

2. WIRELESS MESH NETWORKS

WMNs are wireless ad-hoc networks, consisting of radio nodes that form a mesh topology [2]. These nodes can be either mobile or stationary and generally accommodate multiple hops. A WMN may operate standalone or be connected to other networks (like the Internet or a wireless sensor network). The nodes in WMNs can be either routers or clients and they may be self configuring. The management of WMNs can be either centralised or decentralised. The advantages and disadvantages of Wireless Mesh Networks can be summarised [2],[3] as follows:

Advantages:

- Multiple hops increase the area that the network can cover and therefore No Line Of Sight (NLOS) communication is possible.
- The mesh topology increases the robustness of the network against link failures, because there exist multiple paths to a given node.

Disadvantages:

- Multiple hops increases delay in the delivery of information.
- Routing protocols associated with WMNs tend to be either very complex or overly simple.

3. MIN-CUT MAX-FLOW THEOREM

The Min-Cut Max-Flow theorem states: *The maximum flow of information in a network is equal to the sum of the cut of the link capacities* [4].

Using the theorem, the network in figure 1 is cut, separating the sender node S and receiver nodes X and Y to cross as few of the links as possible to determine the minimum cut.

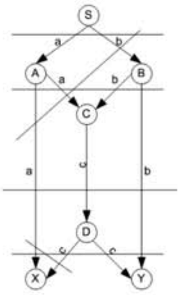


Figure 1: Min-Cut Max-Flow theorem

When each of the links have unit capacity it can be seen that the minimum cut and therefore the maximum flow (or maximum throughput capacity) of this network is equal to 2. The only way that this maximum flow can be achieved, is through the implementation of Network Coding [5],[6].

4. NETWORK CODING

Network Coding is a concept that was first introduced in [6] as a method to utilise the maximum capacity of a network and maximise the flow of information in that network. It suggested coding at packet level in wired peer-to-peer networks.

4.1 The Concept of Network Coding

The Butterfly network from [7], as depicted in figure 2, is used to explain the concept of Network Coding. The links in the figure all have unit capacity and messages *a* and *b* are binary.

Two nodes, A and B both need to transmit their messages to Nodes X and Y. Each of the nodes can forward their own message to the node that is directly connected to it, but have to route their messages through the network to reach the second node. Using traditional routing (figure 2a), node C simply replicates the information it receives from the previous sender node. In this case the two messages *a* and *b* will reach node C simultaneously. Node C will send out message *a* and then after it has been sent, message *b*. Thus, at the end of a single round of transmissions, only node Y will have received both messages, while node X still has only message *a*. This results in a throughput of 1.5 (three of the original two messages were delivered.)

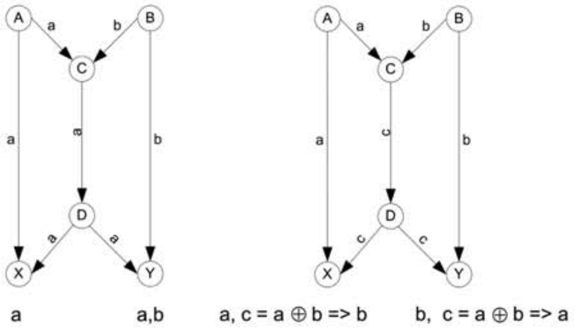


Figure 2: Butterfly Network a) without Network Coding b) With Network Coding

When Network Coding is implemented (figure 2b), node C will have the capability to transmit a linear combination (logical XOR) of the binary messages *a* and *b*. Message *c* is the same length as message *a* and *b* and is transmitted via node D to nodes X and Y. Nodes X and Y then have the capability to decode message *c* by using the message already received from the directly connected source node to solve the two linearly independent equations. In this special case, it merely means another logical XOR of the message that has already been received and the encoded message *c*. In this instance, at the end of a single round of transmissions, nodes X and Y received both messages. Four messages were delivered to two nodes in the communication round, giving a throughput of 2 [6]. This correlates to the result obtained with the Min-Cut Max-Flow theorem for a butterfly network. In effect an extra "virtual"link is created in the network, which improves the capacity of the network.

This method however changes the way node C works, because it has to form linear combinations of the messages it receives before forwarding the linear combination of these messages as a single message. It also requires nodes X and Y to have knowledge of the network topology and how the messages that reach it were encoded in order to deduce the two original messages from the messages received.

4.2 Benefits of Network Coding

The use of Network Coding in a network may provide the following benefits:

1. Throughput [6], [7], [9]: The improved throughput in networks was the first major result of Network Coding. If we refer to the throughput achieved with Network Coding in the deterministic example above, it is clear that the maximum throughput as calculated using the Min-cut Max-flow theorem has been achieved.

2. Robustness [7], [9], [10]: The robustness of the network refers to the ability of the network to remain functioning even though a link has failed completely.

3. Adaptability [9], [11]: Adaptability is an important benefit when looking at WMNs, as this refers to the ability of the network to cope with nodes constantly joining and leaving the network, resulting in a constantly changing topology.

4. Security [9]: The security benefit is an inherent benefit, seeing that linear combinations of data are sent over the network and not the actual data. This benefit while useful, is however not sufficient. If a malicious entity listens long enough and receives sufficient messages to decode the information, the information can still be eavesdropped.

One approach to make Network Coding suitable for wireless networks is Random Network Coding [1]. Random Network Coding allows the nodes in a network to randomly form linear combinations of the data received. These linear combinations are then forwarded through the network (with the information of how they were combined stored in a header field) until it reaches the receiver node. The receiver node then has to wait for enough linear independent combinations to reach it before it can decode the original messages. Deterministic Network Coding however holds a few advantages over Random Network Coding:

- Less decoding delay - Decoding nodes have a predetermined number of messages to wait for and don't have the uncertainty of waiting until enough linear independant messages are received.
- Less overhead - The decoding nodes know how the message was encoded earlier in the network. Therefore no combination vector has to be sent along with the message.

We propose the opportunistic use of a predefined topology to identify localised Network Coding opportunities in WMNs. This approach may be useful to improve network utilisation of a WMN, because the deterministic approach to Network Coding requires little overhead.

5. METHOD OF IDENTIFYING NETWORK CODING OPPORTUNITIES

A method to indicate where in a WMN or how exactly at that specific location, Deterministic Network Coding could be implemented was not found in available literature. We proposed that opportunities for the implementation of Network Coding could be found by searching for "known Network Coding topologies"within larger networks.

The identification of a known network coding topology as a subset of a larger network also defines which nodes should change their forwarding function and which nodes should have the topology information necessary to enable decoding of the linear combinations of messages.

The following method to achieve local throughput gains in a WMN when using predefined Network Coding topologies, is presented:

1. Select a Network Coding topology of which the gain and capacity is known;
2. Derive the connection matrix of the larger network from a suitable distance vector routing algorithm;
3. Search the larger network matrix for the known topology structure;
4. Implement Network Coding at the appropriate nodes;
5. Re-iterate steps (3) and (4) after a routing update.

Three different search methods were examined to implement step 3.

5.1 Iterative looping

The first method was to translate the known Network Coding topology to an adjacency matrix and search for that specific smaller matrix within the adjacency matrix of the larger network.

The simplest method to do this is iterative looping. The fundamental concept of iterative looping is a nested looping structure. This effectively moves a window across the network adjacency matrix and continually compares the data in the current window to that of the desired data.

This method was implemented in MATLAB® to search networks of between 10 and 100 nodes for only one specific Butterfly adjacency matrix. The implementation therefore did not locate any Butterflies that were numbered or orientated differently.

Although this method was able to accurately locate a specific Butterfly adjacency matrix within the larger adjacency matrix, it scaled poorly and was computationally intensive.

5.2 Cross-correlation

The second method was based on the two dimensional cross correlation of the network's 1-hop adjacency matrix and the Butterfly adjacency matrix. The cross correlation of two datasets can be seen as the similarity of those two datasets and is commonly used to search for patterns in a random dataset by forming the cross-correlation.

This method was implemented in MATLAB® to search networks of between 10 and 100 nodes for all permutations of the Butterfly adjacency matrix. The concept of cross-correlation was extended to two dimensions with the network adjacency matrix being used as the random dataset and the Butterfly adjacency matrix being used as the known pattern. The result of the cross-correlation operation *xcor(adjacency, butterfly)*, a function that is based on the *xcorr* function found in the signal processing toolbox of MATLAB®, resulted in a new matrix M with values indicating the level of cross-correlation. In order to extract the correct values, the auto-correlation of the butterfly *xcor* matrix was formed. The location of the Butterfly matrix could be determined by scanning M for the correlation value.

Advantages of using the cross-correlation method is scalability, as the method remains computationally efficient when using larger matrices. A further advantage is the ability to detect where partial Butterfly structures occur within the network connection matrix. These partial structures will have the potential to become butterflies if the topology of the network changes slightly.

This method was compared to the Iterative looping method presented in section 5.1 for both accuracy and computational efficiency. For small matrices of less than 40x40 both methods performed in the same order in terms of computation time. For larger matrices of 100x100, the cross-correlation method's computation time was significantly less than that of the iterative looping method. As the matrix dimensions increase, the computational advantages of the cross-correlation method also increases.

All permutations of an adjacency matrix have to be pre-programmed as known patterns to search for in the cross-correlation method, making it cumbersome. It was however decided to implement another method because the first two methods took very long to complete a search of a network.

5.3 Concatenation

Graph theory is used to explain the third method: Concatenation.

The network is described as a directed graph *G(V,A,R)* where V represents the set of nodes in the network, A is the adjacency matrix describing the topology of the nodes V in $R = [r(e)]_{e \in A}$ that describes the link capacities. The adjacency matrix $A = (a_{ij})$ is an $n \times n$ matrix (n is the number of nodes in the network), with $a_{ij} \in \{1,0\}$. The rows and columns represent the nodes in the network and the *n'th* row and the *n'th* column represent the same node. A connection of one node to another is indicated by a *1* in the matrix and and a *0* indicates that a node is not connected (or within transmission range) to another node. In a network each node is at least connected to itself, so the main diagonal of the matrix will contain only *1's*. If these characteristics are combined, then:

$$A_{n \times n} = (a_{ij}) = \begin{bmatrix} 1 & \dots & a_n \\ \vdots & \ddots & \vdots \\ a_n & \cdots & 1 \end{bmatrix} \qquad (1)$$

If the identity matrix I_n is subtracted from (1) then $B_{n \times n}$ is the true representation of the interconnectivity of the nodes:

$$\begin{aligned} A_{n \times n} - I_n &= \begin{bmatrix} 1 & \dots & a_n \\ \vdots & \ddots & \vdots \\ a_n & \cdots & 1 \end{bmatrix} - \begin{bmatrix} 1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & & 1 \end{bmatrix} \\ &= \begin{bmatrix} 0 & \dots & a_n \\ \vdots & \ddots & \vdots \\ a_n & \cdots & 0 \end{bmatrix} \\ &= B_{n \times n}, b_{ij} \in \{1,0\} \end{aligned} \qquad (2)$$

This representation of the network's topology is used to find all the paths of a certain length within a network, through concatenation. Every row of the matrix is examined to find all the *1's* present. Each time a 1 is found, a 1-hop route exists and this information is stored in an $r \times 2$ matrix $O_{r \times 2}$, where r is the number of 1-hop routes that exist, so that:

$$O_{r \times 2} = [o_{r1} o_{r2}] \qquad (3)$$

where $\begin{matrix} o_{r1} = i \; \forall \; b_{ij} = 1 \\ o_{r2} = j \; \forall \; b_{ij} = 1 \end{matrix}$ when $1 \leq r \leq [(n-1)!]$ and $i < j$.

In some of the literature the Butterfly topology is called the canonical example of Network Coding, emphasising that the Butterfly typology is simple yet significant, without loss of generality. It was therefore decided to use it as a known Network Coding topology. Referring to figure 2 in section 4, it can be seen that there exists a 1-hop and 3-hop path between two sets of nodes. The two 3-hop paths share one common link, therefore to find the Butterfly topology, the 1-hop and 3-hop paths of the adjacency matrix are constructed in order to be used in the search algorithm.

The 1-hop paths are constructed and stored in (3). In order to construct the 3-hop paths, the 2-hop paths first are constructed through concatenation of the 1-hop paths. A $s \times 3$ matrix $T_{s \times 3}$ exists where s is the number of 2-hop paths, so that:

$$T_{s \times 3} = [t_{s1} t_{s2} t_{s3}] \qquad (4)$$

where $[t_{s1} \; t_{s2} \; t_{s3}] = [o_{r1} \; o_{r2} \; o_{h2}]$ where $(o_{r2} = o_{h1})$ and $r < h \leq | O_{r \times 2} |$

The 3-hop paths are stored by creating a $u \times 4$ matrix $D_{u \times 4}$, where u is the number of 3-hop paths that exist, so that:

$$D_{u \times 4} = [d_{u1} d_{u2} d_{u3} d_{u4}] \qquad (5)$$

where $[d_{u1} \quad d_{u2} \quad d_{u3} \quad d_{u4} \;] = [o_{q1} \quad o_{q2} \quad t_{s2} \quad t_{s3} \;]$ where $(o_{q2} = t_{s1}) \; \forall \; q < s \leq |O_{r \times 2}|$,

or $[\; d_{u1} \quad d_{u2} \quad d_{u3} \quad d_{u4} \;] = [t_{s1} \quad t_{s2} \quad t_{s3} \quad o_{q2} \;]$ where $(t_{s3} = o_{q1}) \; \forall \; q < s \leq |T_{s \times 3}|$.

The conditions in equation (5) respresent a concatenation of an 1-hop path with a 2-hop path or a concatenation of a 2-hop path with a 1-hop path.

To find the Butterfly topology, equations (3) and (5) are used to create a $c \times 4$ matrix $H_{c\times4}$, where c represents the number of 3-hop paths that have first and last entries that correspond with those of 1-hop paths:

$$H_{c\times4} = [d_{u1}d_{u2}d_{u3}d_{u4}] \tag{6}$$

where $(o_{l1} = d_{u1})$ and $(o_{l2} = d_{u4})\ \forall\ u < l \leq |O_{r\times2}|$.

A search is performed in (6), the "wings"of the butterfly, i.e. the entries where:

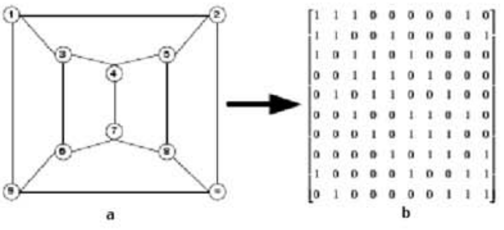
$$(d_{u2} = d_{p2}), u < p \leq |H_{c\times4}| \tag{7}$$

which would confirm the existence of a shared link between sets of wings. An example of this method is presented in the following section.

5.4 Example

In this example, the Butterfly topology is used as the known topology that is the objective of the search in a random 10 node network.

One such random network, where each node is connected to exactly three other nodes (C = 3) is presented in figure 3a and the 10 x 10 connection matrix is shown in figure 3b.



$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix}$$

Figure 3: a) Random network b) Connection matrix of the network in (a)

Finding the known topology: The $B_{n\times n}$ matrix is created from the adjacency matrix presented in figure 3b. The 1-hop paths are obtained and stored in (8). There may be more paths if the algorithm allowed cyclic paths, but for the purpose of this example only non-cyclic paths are allowed.

$$O_{15\times2} = \begin{bmatrix} 1 & 2 \\ 1 & 3 \\ 1 & 9 \\ 2 & 5 \\ 2 & 10 \\ 3 & 4 \\ 3 & 6 \\ 4 & 5 \\ 4 & 7 \\ 5 & 8 \\ 6 & 7 \\ 6 & 9 \\ 7 & 8 \\ 8 & 10 \\ 9 & 10 \end{bmatrix} \tag{8}$$

The 3-hop paths are found by concatenation, using the conditions in equations (4) and (5):

$$D_{u\times4} = \begin{bmatrix} 1 & 3 & 6 & 9 \\ 1 & 3 & 6 & 7 \\ \vdots & \vdots & \vdots & \vdots \\ 3 & 4 & 7 & 6 \\ 4 & 3 & 6 & 7 \end{bmatrix} \tag{9}$$

Equations (5), (6) and (7) are used to determine whether there are any 1-hop paths and 3-hop paths that share the same start and end nodes. The information is stored in (10):

$$H_{u\times4} = \begin{bmatrix} 1 & 3 & 6 & 9 \\ 2 & 5 & 8 & 10 \\ 3 & 4 & 7 & 6 \\ 4 & 3 & 6 & 7 \\ 4 & 5 & 8 & 7 \\ \vdots & \vdots & \vdots & \vdots \\ 5 & 4 & 7 & 8 \end{bmatrix} \tag{10}$$

This matrix is then explored to determine whether any two of the 3-hop paths in (10) contain a shared second hop:

[1 3 6 9] and [4 3 6 7] share [3 6]; [4 5 8 7] and [2 5 8 10] share [5 8]; and [3 4 7 6] and [5 4 7 8] share [4 7].

It can therefore be concluded that there are 3 butterflies present in the network in figure 3b. The three butterflies offer three opportunities for the implementation of Network Coding. However, the three cannot co-exist, as they are not disjoint.

If the middle butterfly in nodes [3 4 7 6] and [5 4 7 8] is selected node 4 should have encoding capability and nodes 6 and 8 should have decoding capability.

6. RESULTS

The method as described in section 5.3 was implemented in networks consisting of between 10 and 20 nodes. A total of 1000 random connection matrices were generated for each size network. The created random matrix generator generated 1000 unique matrices, ensuring that a particular connection matrix is not used more than once, but not necessarily that it was not a permutation of a matrix that was already tested.

6.1 Butterfly Implementation

The generated matrices had connectivity (C) of between 3 and 7, (C = 3 meaning that each node was connected to 3 other nodes). A connectivity of 3 is the minimum connectivity that will allow the possibility of a Butterfly network to exist, but does not guarantee the existence of a butterfly network in large networks. The results of this implementation are shown in figures 4 - 6.

As can be seen in figure 4, when C = 6 or higher, all examined networks contained butterflies.

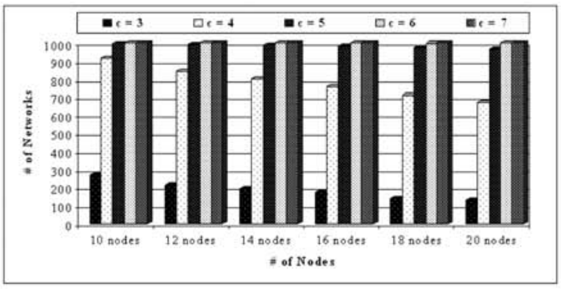


Figure 4: Number of Networks containing Butterflies

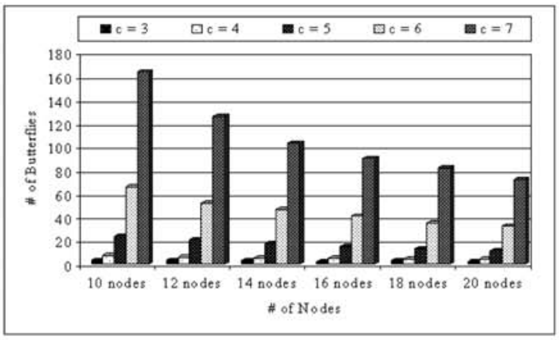


Figure 5: Number of Butterflies per Network

Further, when comparing figures 5 and 6, it can be seen that in all of the networks where butterflies were found and where C = 3, the butterflies were all disjoint. Further it was found that for low connectivity (C = 3 to 4), the number of butterflies stays relatively constant for all size networks, but with higher connectivity (C = 6 to 7), the number of butterflies decrease logarithmically with increasing network size. This may be due to the higher possibility of "islands"forming in larger networks.

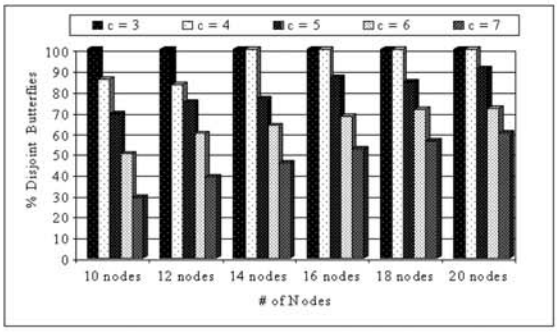


Figure 6: Percentage of Disjoint Butterflies per Network

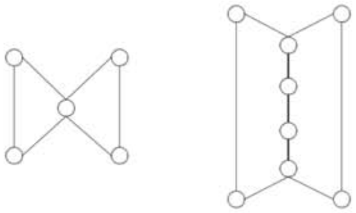


Figure 7: a) Bowtie b) Extended Butterfly

Two variations on the Butterfly topology (which test for a shared link) were also implemented. Firstly the topology was simplified to test for a shared node (refer to figure 7a), from here on referred to as a Bowtie. Secondly the topology was generalised to test for a shared path (refer to figure 7b), from here on referred to as an Extended Butterfly.

6.2 Bowtie implementation

To search for a Bowtie the parameters were selected as follows: C = 4 and higher and the shared 1-hop and 2-hop paths were used to find this topology. Again 1000 random connection matrices with between 10 and 20 nodes were explored.

Although most networks contained at least one of these Bowties, the number of Bowties per network was not very high. A connectivity of C = 6 yielded the best results in terms of presence of Bowties. For all other connectivities the number of networks containing Bowties decreased as the network size increased. This may be due to the complex nature of this topology (each node has to be connected to exactly four other nodes, two of which also have to be connected to each other on each side).

In figures 8 and 9 it can be seen that for low connectivity (C = 4 to 5), the number of bowties stay relatively constant for all size networks. With higher connectivity (C = 6 to 7) however, the number of bowties decrease logarithmically with increasing network size. Once again this phenomenon is attributed to the higher possibility of "islands"forming in larger networks.

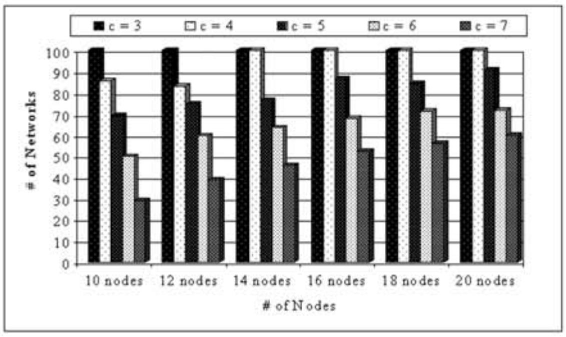


Figure 8: Number of Networks containing Bowties

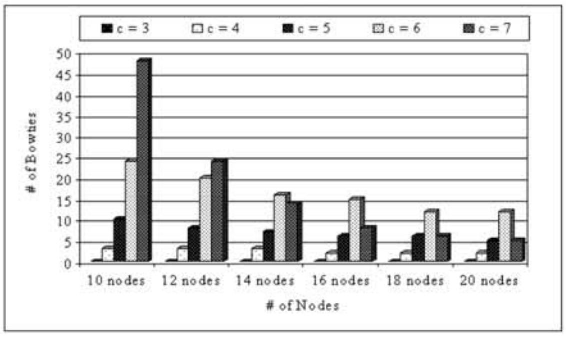


Figure 9: Number of Bowties per Network

6.3 *Extended Butterfly Implementation*

Finally, the method was implemented to find the Extended Butterfly topology. For this implementation, C = 3 and higher was selected and the shared 1-hop and 4-hop paths were used to locate this topology. The same size and same number of random networks as for the previous two simulations were used. The results of this implementation can be seen in figures 10 and 11.

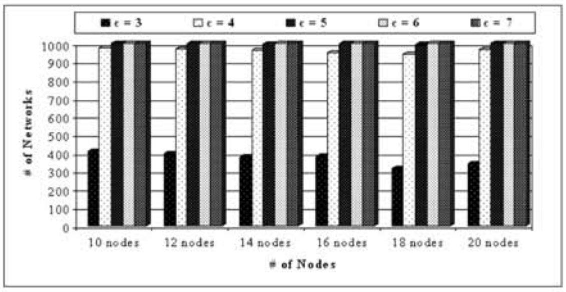


Figure 10: Number of Networks containing Extended Butterflies

This topology showed the same tendencies as the normal Butterfly, but it was found that more networks contained Extended Butterflies than normal Butterflies. When the connectivity C = 3, twice as many networks contain Extended Butterflies, but as can be seen in figure 11, there were not more Extended Butterflies than Butterflies per Network.

This may be because there are not very many opportunities for 4-hop paths in a sparsely connected network. All networks with C = 5 and higher contained Extended Butterflies. All three topologies tend to respond the same to connectivity and network size.

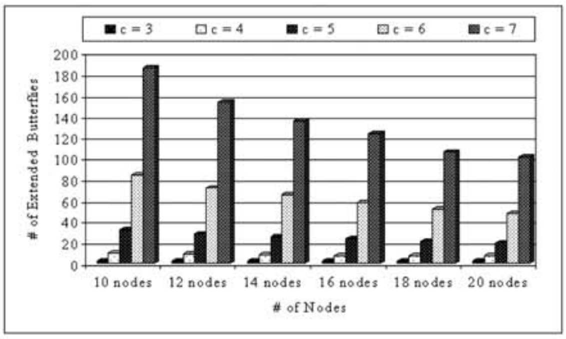


Figure 11: Number of Extended Butterflies per Network

In general it was observed that for all three topologies when connectivity C = 6, a significant number of predefined network coding topologies are probable in a random network. This seems to support [13] which suggested a connectivity of 6 for optimal throughput capacity of a mesh network.

7. THROUGHPUT CAPACITY OF A WMN

Calculating the capacity of a WMN is a complex problem due to the numerous factors that influence it [2], [3]. Studies have shown that a throughput capacity of $0.0976\sqrt{n}$ (where n is the number of nodes) is achievable for a network when C = 6 [12]. Analytical results have shown that the throughput capacity per node decreases as the node density increases [3].

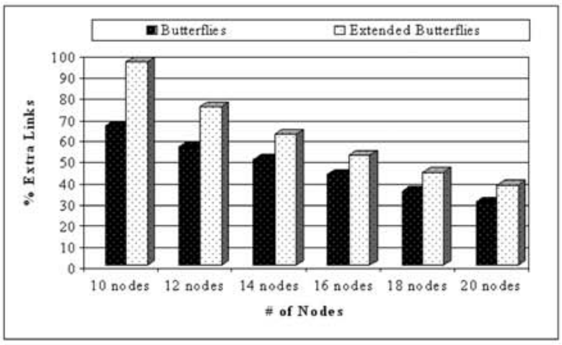


Figure 12: Percentage Extra Virtual Links

The implementation of Network Coding in a WMN can improve the throughput capacity of the network by better utilization of the available capacity. In figure 12 the improvement is shown in terms of the percentage of extra virtual links that are created in a network where C = 6 when using the Butterfly topology or Extended Butterfly topology.

8. CONCLUSION

In this paper a method was presented to determine where and how in a WMN, Network Coding can be implemented. It was implemented by means of three different appraches. The concatenation approach was more efficient than the iterative looping and cross-correlation methods, both in terms of finding topologies and in execution time. The concatenation approach was evaluated for three different topologies.

The implementation of the new method, which identifies opportunities for the implementation of network coding using predefined network coding topologies, provides more available links in the network. For each disjoint butterfly network present (or variation on this topology), one extra virtual link becomes available. Further anticipated benefits of the implementation of this concept are:

- An improvement in the total throughput (better utilisation of the network's capacity).
- Lower occupation of the total network.
- Improved Quality of Service (QoS), because of a lower delay in the network.

The implementation cost of using this method is:

- Clever nodes that can encode and decode received messages
- Route establishment delay
- A high connectivity is required
- Routing Protocol with knowledge of the whole network's topology
- Centralized network control

These costs can be overcome if the characteristics of the typical WMN are utilised, especially when the WMN is formed by nodes with significant computational power that are located within close vicinity of each other. Using laptops or mesh routers will ensure that the nodes have power to perform the computations for these clever nodes and will be able to compensate for the route establishment delay.

The nature of WMNs offer many possibilities for the opportunistic implementation of Network Coding. It is difficult to guarantee a specific increase in capacity due to the varying topology of the WMN. Theoretically this increase in topology can be as high as the maximum throughput capacity, but in practice this is seldom seen.

REFERENCES

[1] T. Ho, R. Koetter, M. Medard, D. Karger and M. Effros, "The benefits of coding over routing in a randomized setting", In Proceedings of the IEEE International Symposium on Information Theory June 2003, page 442, Yokohama, Japan, 2003.

[2] J. Jun and M. L. Sichitiu, "The Nominal Capacity of Wireless Networks", IEEE Wireless Communication. Magazine, vol. 10 pp. 8-14, 2003.

[3] I.F. Akyildiz, X. Wang and W. Wang, "Wireless Mesh Networks: A survey", Computer Networks, vol. 47, pp. 445 487, 2005.

[4] P. Elias, A. Feinstein and C. E. Shannon, "Note on maximum flow through a network", IRE Trans. Information Theory, vol. 2, pp. 117119, 1956.

[5] S. A. Aly, V. Kapoor, J. Meng and A. Klappenecker, "Bounds on the Network Coding Capacity for Wireless Rondom Networks", Information Theory and Applications Workshop, Jan. 29 2007-Feb. 2 2007, pp. 231-236.

[6] R. Ahlswede, N. Cai, S.-Y. R. Li and R. W. Yeung, "Network information flow", IEEE Transactions on Information Theory, vol. 46, July 2000, pp. 1204-1216.

[7] S.-Y. R. Li, R. W. Yeung and N. Cai, "Linear network coding", IEEE Transactions on Information Theory, February 2003, vol. 49, pp. 371- 381.

[8] A. G. Dimakis, P. B. Godfrey, M. J. Wainwright and K. Ramchandran "The Benefits of Network Coding for Peer-to-Peer Storage Systems", NetCod Workshop, January 2007.

[9] C. Fragouli, J. Widmer and J. Le Boudec "Network Coding: An instant primer", ACM SIGCOMM Computer Communication Review, January 2006, vol.36, pp. 63 - 68.

[10] R. Koetter and M. Medard, "Beyond routing: an algebraic approach to network coding", In Proceedings of the Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies, vol.1, pp. 122- 130, 2002.

[11] C. Fragouli, J. Widmer and J. Le Boudec "On the Benefits of Network Coding for Wireless Applications", Netcod, 2006.

[12] L. Kleinrock and J. Silvester, "Optimum transmission radii for packet radio networks or Why six is a magic number", in: Proceedings of the IEEE National Telecommunications Conference, Birmingham, Alabama, pp. 4.3.14.3.5, 1978.

MITIGATION OF ATMOSPHERIC TURBULENCE DISTORTIONS IN LONG RANGE VIDEO SURVEILLANCE

P.E. Robinson* and W.A. Clarke**

* *Faculty of Engineering and the Built Environment, University of Johannesburg, PO Box 524, Auckland Park 2006, South Africa*
E-mail:philipr@uj.ac.za
** *SAIEE, IEEE, Faculty of Engineering and the Built Environment, University of Johannesburg, PO Box 524, Auckland Park 2006, South Africa*
E-mail:willemc@uj.ac.za

Abstract: This paper explores the problem of atmospheric turbulence in long range video surveillance. This turbulence causes a phenomenon called heat scintillation or heat shimmer which introduces distortions into the video being captured. The nature of these distortions is discussed and a number of possible solutions explored. Using these solutions, three algorithms are implemented to attempt to mitigate the effects of heat shimmer. Within this field there is very little subject matter on the topic of objectively comparing the performance of heat shimmer reduction algorithms. A set of possible metrics is proposed in this paper and used to compare the performance of the implemented algorithms. These results provide insight into the nature of the algorithms and the effectiveness of the metrics under consideration.

Keywords: Atmospheric Turbulence, Scintillation, Heat Shimmer, Terrestrial, Optical Flow, Deblurring, Quality Metrics, GPU.

1. INTRODUCTION

The rapid advancement of digital image capture technology has resulted in the development of many technologies devoted to making use of these captured image and video signals [1, 2]. These signals can be captured in a variety of circumstances and more often than not contain a variety of distortions and noise. This degradation is the result of various factors, including the nature of the hardware used to capture the signal and external environmental factors such as lighting and camera jitter [2]. These signals are often used not only by a human operator who could deal with poor quality video, but also automated processing systems that make use of the video signal to automatically gather information about the scene. For these systems quality degradation is a serious problem [1, 2, 3].

Imaging systems are capable of very high magnifications. These kinds of systems are used for long range video surveillance. In the case where the target scene is being viewed from a range of more than 1 km the effects of atmospheric turbulence become apparent [1, 4].

Turbulence in the atmosphere causes pockets of air of varying temperatures and thus densities to move in a random fashion. This movement is caused by the varying densities of the air pockets, wind and terrain. Light from the target scene must travel through this turbulent atmosphere to reach the imaging system. The varying densities of the air pockets cause this light to be refracted by varying degrees and in a continually changing manner. This results in the target scene appearing blurred and to be wavering or shimmering. This means that objects in the scene will appear to be moving even when stationary. This effect is dubbed heat shimmer [1, 3, 4, 5, 6, 7].

Heat shimmer severely limits the range of long range imaging systems, and as such, mitigating the effects of atmospheric turbulence is a major concern when designing these systems [1, 4, 6, 7].

There are two main schools of solution to this problem. The first is to make use of a mechanical adaptive optics system to physically compensate for the effects of the atmospheric turbulence on incoming light rays [11]. The second solution is to make certain assumptions about the nature of the distortion and make use of an image processing approach to digitally enhance the video signal to attempt to reduce the distortion. A few proposed image processing methods are the direct Discrete Fourier Transform (DFT) solution [4], image registration and fusion [1], Adaptive Control Grid Interpolation [6, 7], Image Time Sequence Registration [8], Neural Network approach based on the Monte Carlo [9] and Homomorphic and Power Spectrum approach [10].

While a number of techniques have been proposed to deal with the problem of heat shimmer there has been little work done to find methods of comparing the effectiveness of these techniques. This is primarily due to the fact that it is very difficult to obtain ground truths of real world heat shimmer distorted video. This paper describes the implementation of three algorithms that mitigate the effects of heat shimmer. A number of performance metrics are proposed for use in comparing the effectiveness of these algorithms. Using the proposed metrics the performance of the implemented algorithms are compared. The results will show which of the proposed metrics are viable measures of performance for these types of algorithms.

The other focus of this paper is to start on the road to developing a real-time implementation of a heat shimmer mitigation algorithm. These algorithms are very computationally intensive and there is no account in the literature of how a real-time implementation could be achieved. This paper will explore the use of Graphics Processing Units (GPUs) as tools to accelerate the processing time of these types of algorithms.

The remainder of this paper will be arranged as follows. Section II will give an overview of the implemented algorithms. Section III will describe the details of a GPU implementation of one algorithm. Section IV will describe the proposed performance metrics and Sections V and VI will described the experiments and results. Finally VII will present the conclusion.

2. Overview of Algorithms

The algorithms implemented in this paper classify the effects of heat shimmer on long range video into two forms of distortion as described in [6, 7]. The first effect of heat shimmer is to cause photometric distortion or blurring of the scene being viewed. The second effect is geometric distortion which produces an apparent wavering motion of elements in the scene. An assumption is made that the geometric distortion is quasi-periodic. This means that any given point in the scene should oscillate periodically around its true position. The algorithms in this paper seek to reduce the effects of heat shimmer by tackling each of the above described distortions.

1.1 Averaging and Wiener filtering algorithm

This algorithm was developed in conjunction with [23] with the intention of developing the simplest possible technique to mitigating the problem of heat shimmer. The algorithm tackles each of the distortions introduced into the captured video by heat shimmer independently. Firstly the algorithm seeks to reduce the amount of geometric distortion in the video sequence by exploiting the assumption that the geometric distortion is quasi-periodic. To do this a simple ratio averaging scheme is used to maintain a running average of the frames in the sequence as shown in the following equation:

$$g(x,y,t) = \alpha f(x,y) + (1-\alpha) g(x,y,t-1), \tag{1}$$

Where:
x, y = the pixel coordinates in an image frame
t = the current frame in the sequence
g(x,y,t) = updated ratio frame average
g(x,y,t-1) = the previous frame average
f(x,y) = the current incoming frame
α = a scalar value between 0 and 1

α is chosen for the algorithm and controls the weighting of the ratio of the current frame and the previous frame average that appears in the new frame average. The quasi-periodic motion in the sequence is thus averaged out to zero. The down side of this process is that the averaging operation introduces blur into the sequence in addition to the photometric distortion already present.

The second stage of the algorithm addresses the photometric distortion. This step makes use of a model of the blurring effect of the atmosphere and a Wiener filter that allows one to filter out the blur represented by this model from the video sequence to produce a sharper image. The following equation shows the transfer function of the Wiener filter [2, 13]:

$$H_w = \frac{H^*}{|H|^2 + {S_N}/{S_F}}, \tag{2}$$

Where:
S_N = power spectrum of the additive noise
S_F = power spectrum of undistorted the image
H = the model of the distortion
H_w = the Wiener Filter based on the specified model

The modulation transfer function (MTF) used as a model for this process was presented by Hufnagel and is shown below [12]:

$$H(u,v) = e^{-\lambda(u^2+v^2)^{5/6}}, \tag{3}$$

Where:
λ = parameters that controls the amount of blur present in the model
u and v = the 2 dimensions of the spatial frequencies.

The noise-to-blurred signal ratio is estimated as the difference between the global variance of the whole frame and the average local variance. This is due to the fact that the noise and the blurred image are uncorrelated. The NSR is thus estimated using the following equation:

$$NSR = \frac{Local\ Variance}{Global\ Variance - Local\ Variance}, \tag{4}$$

The primary issue with this process is that the nature of blurring effect of the atmosphere is completely unknown and needs to be determined using only the distorted video sequence.

This is done by defining a search space of plausible values for λ and applying the Wiener filter to the current frame for each of these values and then using a metric to determine which of the λ values results in the sharpest output. The sharpness metric used to measure the sharpness of the resulting frame is based on the Laplacian operator which is defined as follows [14]:

$$\nabla^2 I = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2}, \tag{5}$$

Where:
I = the 2-dimensional intensity map of the image.

The effect of this operator can be approximated by the following discrete kernel [11].

-1	-1	-1
-1	+8	-1
-1	-1	-1

Figure 1: Laplacian operator approximation kernel

This kernel is applied to the image and accentuates areas of high spatial frequencies such as edges and reduces the intensities of areas of uniform colour. After the operator is applied the average intensity is found of the result. The more high frequency content there is in the image the higher this metric will be and thus the sharper the image appears. This allows us to find which value of λ resulted in the sharpest image and use that frame as the output of this stage of the algorithm. In [7] it is proposed to use the kurtosis of the image as a sharpness metric.

Both the sharpness metric mentioned above are time domain metrics which is a disadvantage when working with the Wiener Filter as the Wiener Filter is a frequency domain filter. This means that an incoming frame needs to have the Fast Fourier Transform (FFT) performed on it before it can be filtered. Then after the frame is filtered for each value of λ it will need to have the Inverse Fast Fourier Transform (IFFT) performed before the time domain sharpness metrics can be used.

It is proposed that a frequency domain sharpness metric be used so that only one FFT and IFFT operation needs to be performed per frame. The frequency domain sharpness metric designed for this task measures the power present in the high frequency bands of the image. This is done by finding the magnitude of the FFT values in the area described in figure 2.

In the figure the DC value is assumed to be at the centre. The range of frequency values measured fall into the light grey area described by the normalized coordinates. This area of measurement was found experimentally to give the most reliable sharpness measurements. The flow diagram of the entire algorithm is presented in figure 3.

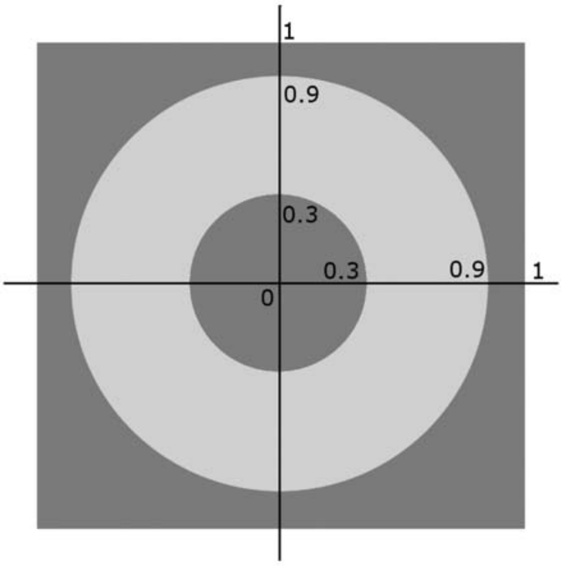


Figure 2: Frequency domain sharpness metric area of measurement. DC value assumed to be at centre.

2 WIENER FILTERING AND OPTICAL FLOW BASED DEWARPING ALGORITHMS

The following two algorithms are based on the work of [6, 7] and are chosen due to the fact that they follow the same paradigm as the first algorithm. This paradigm treats the effects of heat shimmer as two separate distortions, photometric and geometric. These approaches first deal with the photometric distortion and adhering to the same paradigm means we can make use of the Weiner Filtering and Sharpness metric based method used in the first algorithm.

These algorithms however use a different approach to dealing with the geometric distortion as suggested in [6, 7]. An optical flow technique is used to map the movement of all the pixels from one frame to the next in the video sequence. Optical flow techniques produce a vector field that describes the motion of all the pixels between two video frames. The two optical flow algorithms implemented were the Lucas-Kanade [14] and Horn-Schunk [15] algorithms, which are the most common optical flow techniques. Both of these algorithms make use of differential methods to determine the optical flow between two frames.

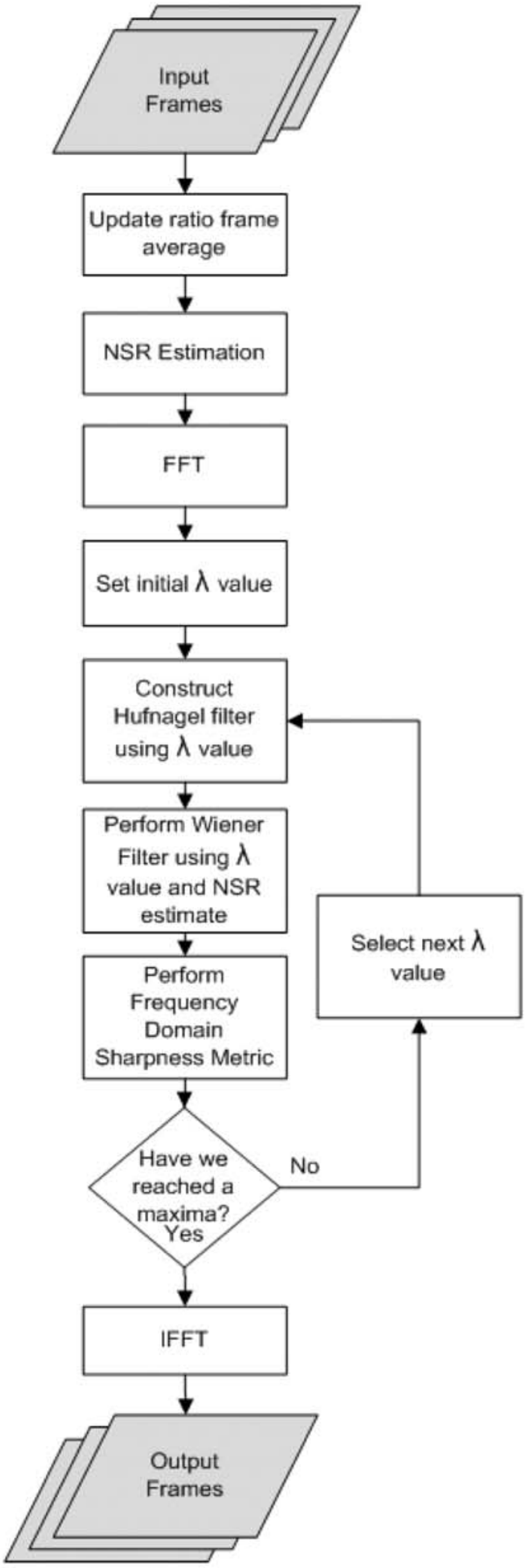

Input Frames
Update ratio frame average
NSR Estimation
FFT
Set initial λ value
Construct Hufnagel filter using λ value
Perform Wiener Filter using λ value and NSR estimate
Perform Frequency Domain Sharpness Metric
Have we reached a maxima?
No
Yes
Select next λ value
IFFT
Output Frames

Figure 3: Averaging and Wiener filtering algorithm block diagram

The general assumption made by these algorithms is that while pixels are moving from frame to frame, their intensities should not change significantly. Based on this assumption each algorithm defines a constraint equation that can be solved to find the optical flow for the given two frames which are captured at times t and $t+\delta t$. The motion for every pixel position is calculated by these techniques.

The Lucas-Kanade algorithm [14] makes use of the following constraint equation:

$$I(x,y,t) = I(x + \delta x, y + \delta y, t + \delta t), \tag{6}$$

Where:
$I(x,y,t)$ = the pixel intensity at the spatial coordinates x,y and the temporal coordinate t.
δx, δy, δt = the distance that the pixel has moved between the two frames.

The equation states that the pixel intensity in the first frame will be the same at its new position in the second frame [14, 16]. Optical flow algorithms generally suffer from the Aperture problem, which means that the information contained within the two frames under consideration is not enough to solve for the velocity of every pixel in both the x and y dimensions. In the Lucas-Kanade algorithm this problem is overcome by assuming that the optical flow inside a small sub window of the frames centred around the current pixel under consideration is constant. By using a Taylor series expansion of equation 6 and using the least squares method, the following equation can be derived which describes how to calculate the x and y velocities of the pixel at the centre of the current window.

$$\begin{bmatrix} V_x \\ V_y \end{bmatrix} = \begin{bmatrix} \sum I_{x_i}^2 & \sum I_{x_i} I_{y_i} \\ \sum I_{y_i} I_{x_i} & \sum I_{y_i}^2 \end{bmatrix} \begin{bmatrix} -\sum I_{x_i} I_{t_i} \\ -\sum I_{y_i} I_{t_i} \end{bmatrix}, \tag{7}$$

Where:
i = 1...n where n is the number of pixel elements in each window centred at x and y.
V_x, V_y = the velocities of a given pixel in the x and y directions.

We also define the following to represent the differentials used in this method:

$$\frac{\partial I}{\partial x} = I_x, \frac{\partial I}{\partial y} = I_y \text{ and } \frac{\partial I}{\partial t} = I_t, \tag{8}$$

The Horn-Schunk algorithm [15] is similar to the Lucas-Kanade method in that it makes use of the differentials between the two frames but it overcomes the Aperture problem by introducing a global smoothness constraint α.

The following equation is the energy function, in two spatial dimensions, that is minimized to find the optical flow [15]:

$$E = \int\int \left(I_x V_x + I_y V_y + I_t\right)^2 dxdy + \alpha \int\int \left\{ \left(\frac{\partial V_x}{\partial x}\right)^2 + \left(\frac{\partial V_x}{\partial y}\right)^2 + \left(\frac{\partial V_y}{\partial x}\right)^2 + \left(\frac{\partial V_y}{\partial y}\right)^2 \right\} dxdy \tag{9}$$

Using the Jacobi method to minimize this equation iteratively the following equations are found:

$$V_x^{k+1} = \overline{V_x^k} - \frac{I_x(I_x\overline{V_x^k} + I_y\overline{V_y^k} + I_t)}{\alpha^2 + I_x^2 + I_y^2}, \tag{10}$$

$$V_y^{k+1} = \overline{V_y^k} - \frac{I_y(I_x\overline{V_x^k} + I_y\overline{V_y^k} + I_t)}{\alpha^2 + I_x^2 + I_y^2}, \tag{11}$$

Where:
$\overline{V_x^k}$, $\overline{V_y^k}$ = the average of these velocity values in the local neighbourhood.

The Horn-Schunck method produces dense optical flow fields but is fairly sensitive to noise. The above techniques can be used to produce vector fields describing the motion of every pixel between two frames. If we make the assumption that the geometric distortion caused by heat shimmer is quasi-periodic [6, 7], we can use the optical flow data to find the true position of the pixels in a video frame. To do this for a specific frame, a window of frames around that frame is chosen and the optical flows between the current frame and each other frame in the window is calculated.

If the motion present in the frames is quasi-periodic, taking an average of the optical flow vectors should reveal the true position of each pixel as that pixel would have been wavering around its true position. This data is then used to move the pixels to their correct positions. This process can leave holes in the resulting image, which are filled in using the mean intensity in the current windows of frames at that position. The following figure shows the flow diagram of the algorithm.

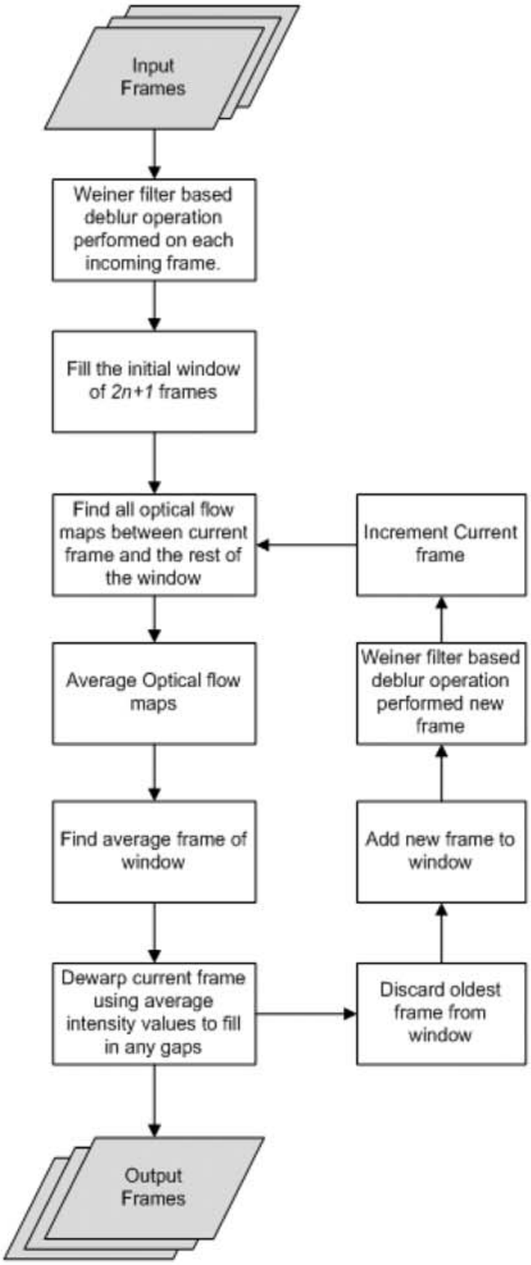

Input Frames
Weiner filter based deblur operation performed on each incoming frame.
Fill the initial window of 2n+1 frames
Find all optical flow maps between current frame and the rest of the window
Increment Current frame
Average Optical flow maps
Weiner filter based deblur operation performed new frame
Find average frame of window
Add new frame to window
Dewarp current frame using average intensity values to fill in any gaps
Discard oldest frame from window
Output Frames

Figure 4: Wiener filtering and Optical flow based algorithm block diagram

3 GPU IMPLEMENTATION

Image processing technology is maturing very quickly and we have been seeing its real world application for a few years now. However, the standard CPU based computing platforms are still struggling to provide enough performance in order to implement image processing theory in a practical way. This is even true of the newest multi-core processors that are now available. This lack of performance has driven the use of non-standard platforms, such as the FPGA, for these tasks, and these platforms are expensive and difficult to develop for.

In recent years the Graphics Processing Unit (GPU) has emerged as a possible solution to this dilemma. GPUs are found on the commercial off-the-shelf graphics cards that we have been using for years for the express purpose of rendering graphics for video games. Recently GPUs have become user programmable and able to support floating point precision calculations. It has become apparent that for applications that can make use of their parallel stream processing architecture that GPUs can far outstrip CPU in price and performance.

A GPU can be seen as a parallel stream processor. This means that the GPU contains a number of processing pipelines each capable of processing an element of data independently and in parallel with the other pipelines. This architecture is useful when performing the same kernel or operation on each element of data in a set of data. As can be seen in Figure 5 this architecture has far less cache and control overhead than that of a traditional CPU. This is because each pipeline is self contained and once data is in the pipeline it does not need any further input until after the data emerges from that pipeline.

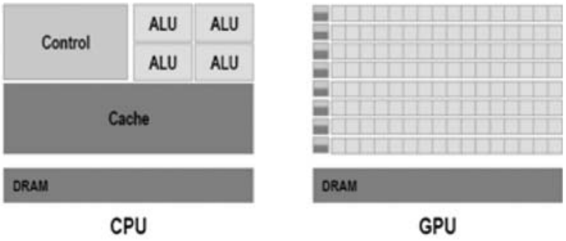


Figure 5: CPU vs. GPU architecture [17]

In the GPU data is represented as vertices, which are points in the co-ordinate system defined in the GPU, and textures or colours which are mapped onto the vertices. The graphics pipeline in a typical GPU is roughly summarised in figures 6 and 7 [18].

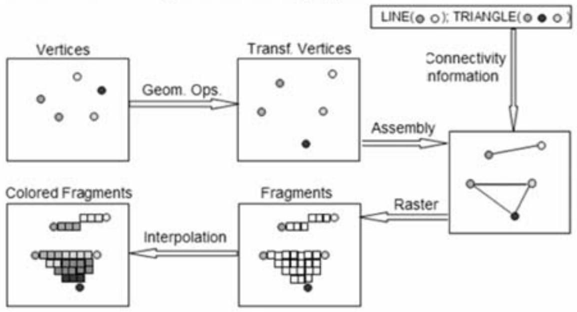


Figure 6: Visualisation of pipeline functions [18]

The parts of the pipeline that we are most interested in are parts whose functionality we can replace with our own programs. These are the vertex transformation stage and fragment texturing and colouring stage [18].

The process begins by taking the vertices, which are points in space that have colours, normals and texture coordinates associated with them, and transforming their characteristics. The next step is connecting each of the related vertices into lines and polygons. The next step is the step where the majority of the processing will be done. In this step the frame being rendered is rasterized. This means that the frame is divided into elements or fragments each representing a pixel in the frame. The texture and colour information of the vertices is then used to define the colour for each fragment in the scene. We will replace the fixed functionality with our own program. These programs are called the vertex and fragment shaders. The respective shaders are executed once for each vertex in the frame and once for each fragment in the frame [18].

The vertex and fragment shaders are invoked by rendering a quad, which is a rectangle with a vertex at each corner, which is exactly the size of the texture we want processed. The vertex shader is called once for each corner of the quad and the texture is rasterized in such a way that each fragment corresponds exactly to a pixel in the texture. Our fragment shader is executed exactly once for each of these fragments [18, 20].

For this project OpenGL was selected as the API used to interface with the GPU hardware. GLSL was selected as the shader scripting language. These selections were made primarily due to the platform portability of OpenGL and GLSL [19].

By default OpenGL will render its output to the screen but it is possible to use the FrameBuffer extension to redirect the rendered output to a frame buffer which has been linked to a texture. This allows one to use that output as the input for another rendering pass or to send that data back to the CPU after it has been processed [21]. This is critical for recursive algorithms.

The Lucas-Kanade based Wiener filtering and optical flow algorithm was implemented on the GPU to give an idea of the performance increases the GPU provides. The majority of the algorithm was easily translated into the parallel architecture of OpenGL as kernel convolutions are easily implemented to be executed for each fragment. The main issues encountered while porting the algorithms to OpenGL were the implementation of recursive algorithms, calculating the mean value of a texture and the dewarping stage of the algorithm.

The main recursive stage in the algorithm was the implementation of the decimation-in-time Fast Fourier Transform (FFT) [22]. The FFT is a recursive algorithm with a number of stages which are dependent on the previous iteration of the algorithm. This recursion is achieved using the ping pong techniques described in [21]. This uses two frame buffers which switch roles as the input to the current iteration and the storage space for the output of that iteration. This allows us to perform recursion with only two frame buffers.

While calculation of the mean of an image is relatively trivial in a serial architecture on a parallel architecture it presents a problem. A given shader program will be executed for each fragment of a texture independently and the shader program can not know the results of its execution on any other fragments during a same rendering step. This means there is no way to read the values of all the fragments in a way that will exploit the parallel nature of the GPU. To achieve the mean calculation in parallel a recursive down-sampling algorithm was used. In each successive stage of the algorithm the texture was down-sampled by a factor of 2 resulting in a texture a quarter the size of the original. This is repeated until the image is down-sampled into a single fragment which will be average value of the entire texture.

The most significant issue encountered during the GPU implementation was the lack of support for scattering operations in OpenGL 2. The Dewarping stage of the algorithm required that a given pixel be moved to a new location based on the average optical flow map. This is not possible using only fragment shaders as when a fragment shader is executed it can only affect the value of the pixel it was executed for. This means that it is impossible for a fragment shader to read from the average optical flow map where the current fragment should be moved to, and proceed to change the value of the destination fragment. This problem was solved by making use of the vertex shader. A grid of point vertices was created, one for each pixel in the texture to be dewarped. A Vertex shader was then written that is executed once for each vertex being rendered. The shader reads the vector describing where the current vertex should be moved to from the average optical flow map. The shader then modifies the position of the vertex accordingly. The vertex is finally mapped with the colour from the original position in the texture inside the fragment shader.

4 OVERVIEW OF METRICS

Metrics and methods are used to measure frame per second (FPS) performance and output image quality improvement of the algorithms. The metrics and methods used are:

- Measurement of FPS performance;
- Image sharpness measurement;
- Aberration measurement; and
- Modulation Transfer Function (MTF) measurement.

4.1 FPS performance

The FPS performance of the algorithms is related to the algorithms complexity. The greater the complexity of the algorithm, the more resources and processing time it will require and will result in a lower FPS measurement.

4.2 Image sharpness

The image sharpness measurement uses the Laplacian Operator Sharpness Metric described in equation 5 to obtain a value indicative of the image sharpness. A larger value indicates a greater level of image sharpness.

4.3 Aberration

The aberration measurement measures the horizontal and vertical displacement present in a video, this relates to the geometric distortions due to atmospheric turbulence. The horizontal and vertical displacement lengths are recorded and compared to the other algorithms and unprocessed video footage.

4.4 Modulation Transfer Function (MTF)

The modulation transfer function across spatial frequencies is determined using:

$$MTF = \frac{I_{max} - I_{min}}{I_{max} + I_{min}} / \frac{W - B}{W + B}, \qquad (12)$$

Where:
I_{max} = the maximum intensity value within that frequency
I_{min} = the minimum intensity.
W = the maximum luminance for white areas
B = the minimum luminance for the dark areas.

A higher modulation index is more desirable because within a grey level image, the darkest and lightest grey areas will be further apart and details will be more pronounced.

5 EXPERIMENTAL SETUP

Each experiment required atmospheric turbulence affected video footage. Using a constructed imaging system as shown in figure 7, video footage was captured of target charts setup 1.2km away.

The constructed imaging system consisted of a telescope coupled to an internet protocol surveillance camera. The telescope became the ‘lens’ of the surveillance camera. The telescope used was a Celestron Nexstar 8 SE (8 inch Schmidt Cassegrain telescope), the large diameter allows a large amount of light to be collected, thus helping to reduce exposure times. An Arecont AV3100 surveillance camera was coupled to the telescope using a CS to 1.25 inch connector.

Each experiment made use of different target charts. In order to control the level of uncertainty in the results, all video footage was captured within one hour period to

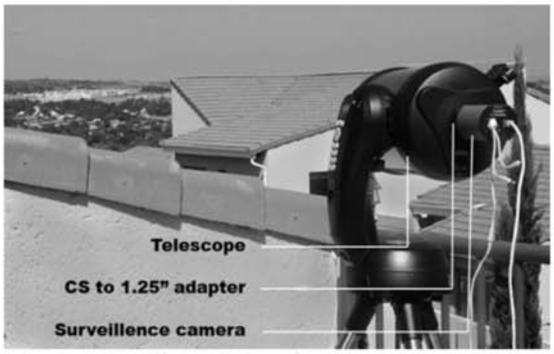


Figure 7: Imaging System

avoid dramatic changes in atmospheric conditions. Video footage was captured on a cloudless day to ensure consistent lighting conditions and shorter exposure times. Wind speed and direction was consistent over the period.

Each experiment made use of different target charts. In order to control the level of uncertainty in the results, all video footage was captured within one hour period to avoid dramatic changes in atmospheric conditions. Video footage was captured on a cloudless day to ensure consistent lighting conditions and shorter exposure times. Wind speed and direction was consistent over the period.

CPU	Intel Core 2 Duo E6750 @ 2.66 GHz
Motherboard	Asus P5K-Deluxe
RAM	2 GB DDR2 800 MHz

Table 1: Test PC Platform Specifications

5.1 *FPS performance*

The FPS performance of the algorithms is calculated using the time taken to process 100 image frames. This process was repeated 10 times for each video size and the average presented here. The same video footage is used for each of the algorithms so that results may be comparable.

5.2 *Image sharpness*

Sharpness measurements are taken across 50 image frames and an average calculated. Like the FPS performance experiment, the same video footage is processed by each of the algorithms and the sharpness measurement is then taken of the outputted videos so that results may be comparable.

5.3 *Aberration*

The aberration measurement is performed using a checkered chart of which video footage is captured through turbulent atmosphere. The video is then processed by the algorithms to determine their abilities to stabilize the footage. This method is similar to that used in [23]. The checkerboard target used consists of 20cm in [23]. The checkerboard target used consists of 20cm sized black and white squares. All the algorithms processed the same video footage to ensure exposure to and suppression of the same atmospheric turbulence conditions. After processing the video footage, maximum intensity values of the image are recorded for 50 image frames and then a grey level threshold is taken to identify square edges. The midpoint threshold value is determined from the maximum and minimum intensity values of the white and black blocks respectively. Horizontal and vertical lengths of the squares are measured and an average value calculated.

5.4 *Modulation Transfer Function (MTF)*

The MTF measurement experiment is similar to the method used in [24]. The spatial frequency charts used are viewed through atmospheric turbulence and the video footage captured and processed by each algorithm. An example of a spatial frequency chart containing various spatial frequency sizes is shown in Figure 8.

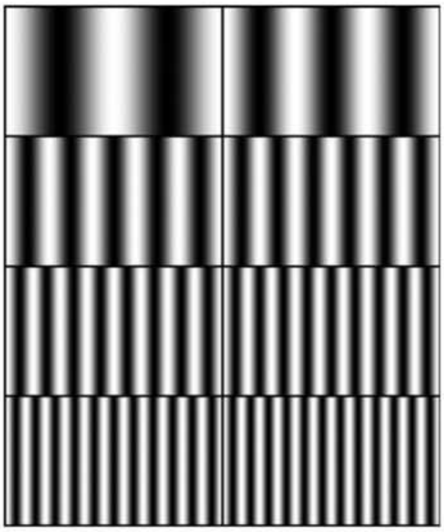


Figure 8: Spatial frequency chart example

Spatial frequencies used ranged from 415mm/lp (millimetre per line pair) to 10mm/lp. The captured video footage is processed by each of the algorithms and MTF measurements taken for each processed video using equation 12. The *W* and *B* intensity values are obtained from the lowest frequencies.

To account for the geometric distortions that vary across the video frames, a frame containing the maximum intensity values and one containing the minimum intensity values across 50 image frames are recorded. Measurements are taken from these two frames. More than 20 maximum intensity values are recorded and I_{max} for that frequency is calculated from the average. I_{min} is found in the same way for each spatial frequency except using the minimum intensity image that was constructed from the recordings.

6 RESULTS

For each of the video quality metrics human assessment was used to evaluate the effectiveness of the metric, and whether the metric was providing a pertinent insight into algorithm performance.

6.1 FPS performance

In this experiment a single 800x600 video sequence was resized to create a set of video sequences of various sizes. The CPU algorithms were run 10 times for each of the video sequences and the average frames processed per second was measured and recorded. The GPU implementation of the Lucas-Kanade based Wiener filtering and optical flow algorithm was executed on a number of different nVidia GPUs and one AMD/ATI GPU. The results of the experiment are presented in figure 9, a logarithmic scale had to be used to represent the results.

It can be seen that the CPU algorithms exhibit an exponential increase in processing time as the size of the videos increase exponentially. This was expected and demonstrates the serial nature of the CPU implementations. The Wiener filtering and Averaging algorithm performs the best out of the CPU implementations which was also expected, due to the fact that it is a much simpler algorithm than the optical flow based algorithms. The Lucas-Kanade based dewarping algorithm is faster than the Horn-Schunck based algorithm which is one of the reasons why it was chosen for the GPU implementation.

The GPU implementation exhibited very little change in processing time as the video sizes changed. This is due to the fact that the FFT and filtering steps of the algorithm need to use power-of-two sized textures and as such a texture of size 1024x1024 or 512x512 was used for the most intensive steps in the algorithm, and thus the processing times were very similar. The jump from the 1024x1024 textures to the 512x512 textures can be seen on the graph. This flat response is also due to the GPU's parallel nature allowing the GPU to scale with the increased video size much more effectively than in a serial architecture. At the largest resolutions the GPU implementations provide a performance increase of 3 orders of magnitude over the CPU algorithms.

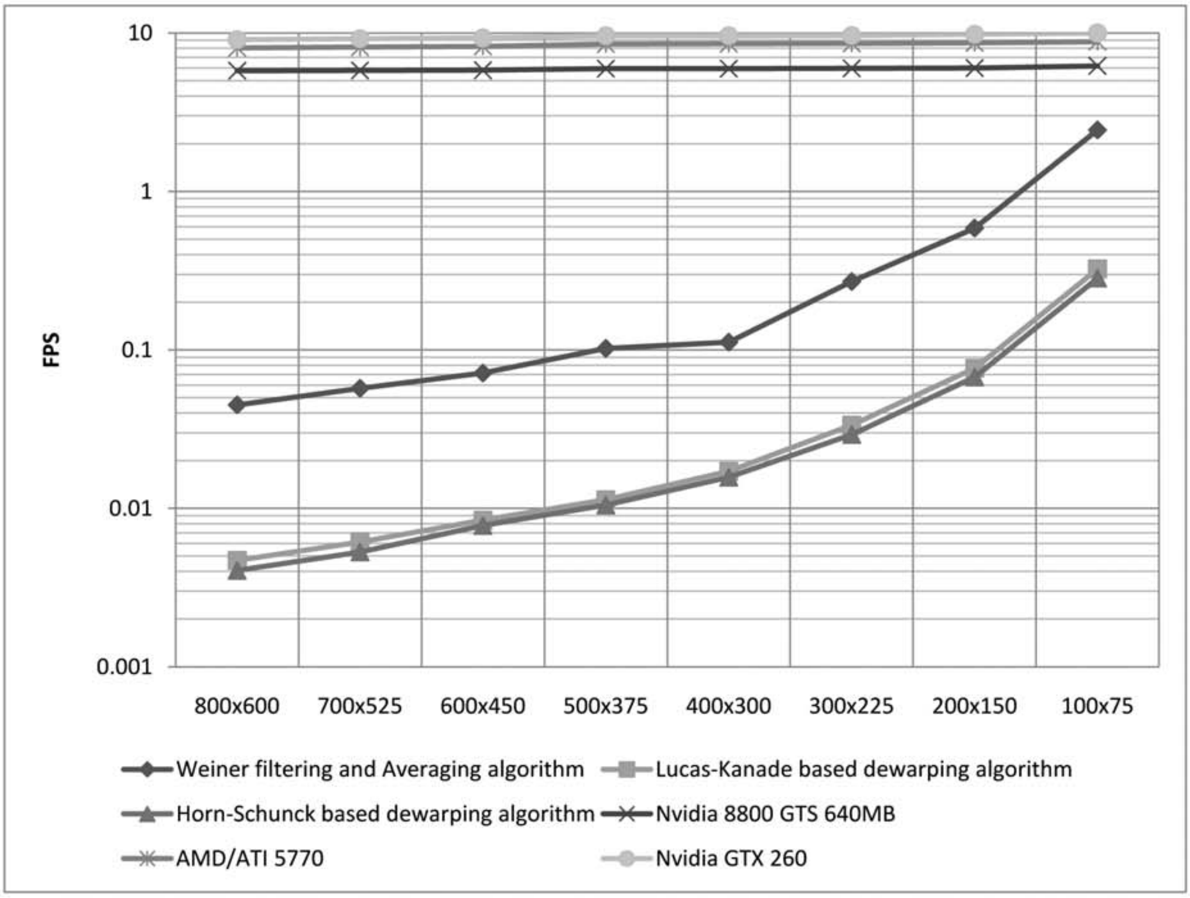


Figure 9: FPS Performance test results

6.2 Image sharpness

This experiment makes use of the Laplacian operator based sharpness measurement. A higher metric value indicates a sharper image.

Figure 10 shows the sharpness measurements for video processed using the Wiener filtering and Averaging algorithm making use of the shown range of ratios used in the ratio averaging portion of the algorithm. Also shown is the sharpness measurement for the unprocessed video

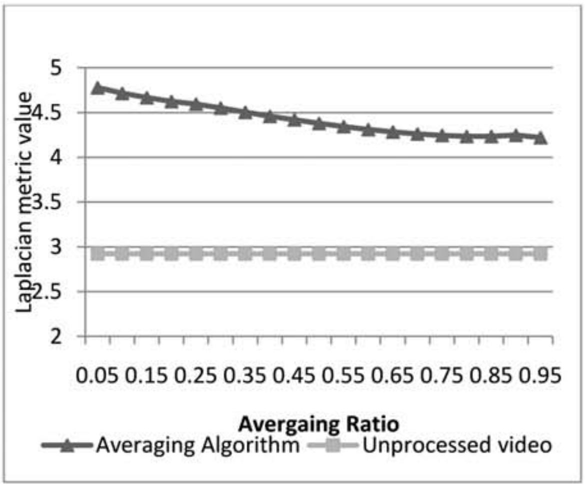


Figure 10: Wiener filtering and Averaging algorithm sharpness results

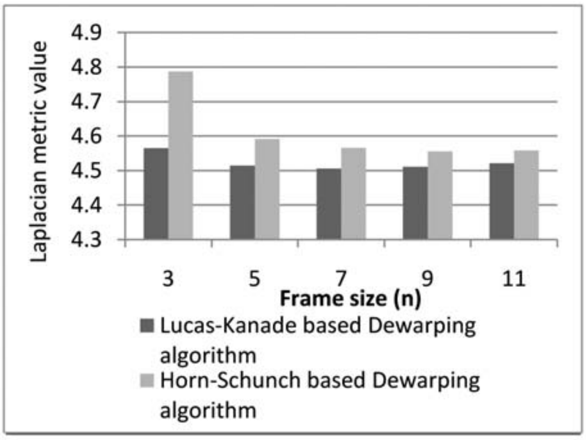


Figure 11: Optical flow based dewarping algorithm sharpness results

It is apparent that as the Averaging Ratio gets larger the sharpness of the resulting video decreases. This is due to the fact that the ratio dictates how much of the average frame is retained as each frame of the video is processed. It is apparent that the processed video is significantly sharper than the unprocessed video.

Figure 11 shows the sharpness measurements of the Lucas-Kanade based Dewarping algorithm and Horn-Schunck based dewarping algorithm. In this graph *n* indicates the size of frame being used to find the average optical flow that will be used to dewarp each frame. The smallest frame size results in the sharpest image as the dewarping operation occurs over fewer frames.

From figure 11 it is apparent that the Horn-Schunck based algorithm produces better results when using this metric. This however demonstrates a flaw with the metric being used as when the output videos are assessed by a human for both the Lucas-Kanade and Horn-Schunck based algorithms it is apparent that the Horn-Schunck video contains far more high frequency noise and is in fact less sharp. The sharpness metric measures the high frequency content in the frames and thus produces a strong response to this high frequency noise. This gives us these skewed results.

6.3 Aberration

The results of this experiment should illustrate the severity of the geometric aberrations present in the video sequence under consideration. The larger the displacements measured the more severe the aberrations in the video are.

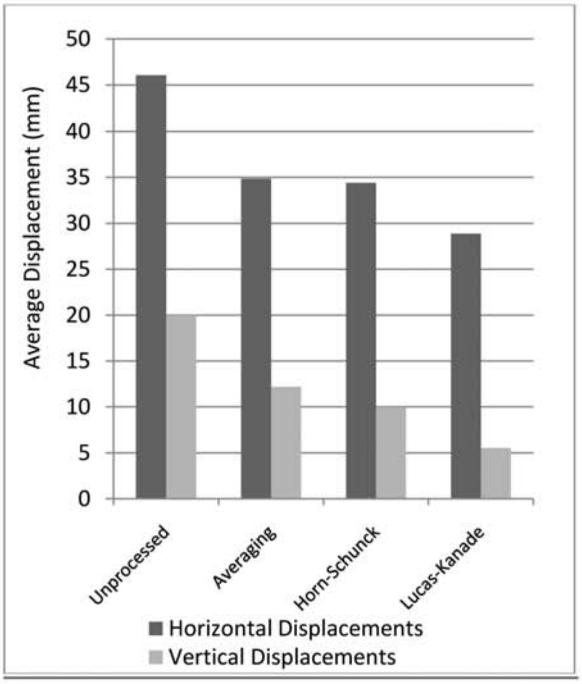


Figure 12: Displacement results

From figure 12 it can be seen that as expected the horizontal displacements are far higher than the vertical displacements due to the lateral effect of the wind. The results show that all the algorithms produce a reduction of the measured aberrations in the video and thus are stabilizing the geometric distortion present in the video. It can be seen from the results that the Lucas-Kanade based dewarping algorithm reduces the measured aberrations in the processed video the most dramatically out of the 3 algorithms. While the Averaging algorithm can potentially stabilize the video the most effectively out of the three algorithms this results in a blurring effect which can be detected by the aberration metric.

6.4 Modulation Transfer Function (MTF)

The blurring effect of the atmosphere suppresses the higher spatial frequencies that can be seen in an image. MTF analysis allows one to measure how much various spatial frequencies are degraded by the atmosphere. This determines the level of spatial detail that is visible in a given image or video.

The MTF (Modulation Transfer Function) provides the spatial frequency response of the video sequence in question. A higher response will mean that the specific spatial frequencies are more visible. From the figures it is apparent that the MTF curve of the unprocessed video exhibits a steeper gradient than the MTF curves of the processed video sequences.

In the lowest frequencies there is not a striking improvement for any of the three algorithms but it can be seen that the three algorithms do improve the MTF curve at these frequencies somewhat. The Horn-Schunck based algorithm performed the best at the lowest frequencies.

The frequencies that we are the most interested in are the high spatial frequencies, as these frequencies are the worst affected by blurring and contain the spatial detail we are trying to restore. It is apparent that these frequencies also show the biggest variation in results between the three algorithms.

The Wiener filtering and Averaging algorithm performs the worst out of the three algorithm and at times seems to worsen the MTF response when compared to the unprocessed video. This is due to the blurring effect of the averaging portion of the algorithm which will degrade the higher frequencies present in the video. It is clear that the Lucas-Kanade based algorithm has the shallowest curve and highest response at the higher frequencies. The Horn-Schunck based algorithm has the second best response and shows an increase in the MTF curve for the entire range of spatial frequencies.

6.5 Final remarks

From the various metrics it is apparent that the Lucas-Kanade based Dewarping algorithm appears to produce the sharpest and most stable video results of the three algorithms under consideration. It can also been seen that the Laplacian operator sharpness metric is over sensitive to high frequency noise and as such is not a useful measure of sharpness on its own.

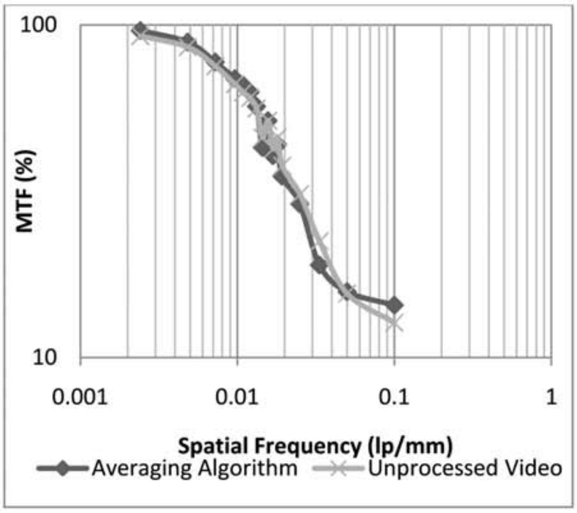


Figure 13: MTF Curve of Averaging Algorithm

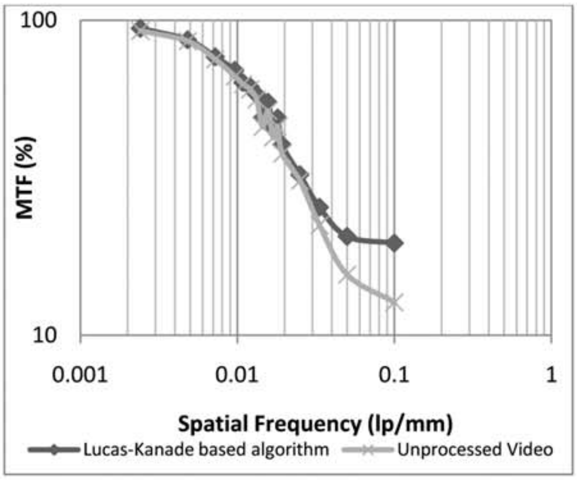


Figure 14: MTF curve of Lucas-Kanade based dewarping algorithm

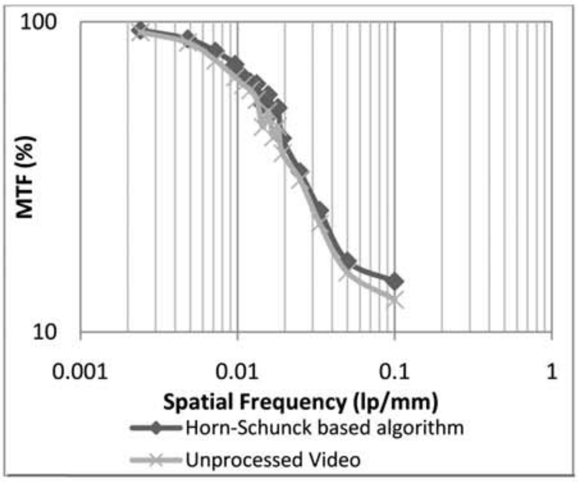


Figure 15: MTF curve of Horn-Schunck based dewarping algorithm

7 CONCLUSION

Long range surveillance systems have numerous advantages within the military and civil fields, however at long distances, the atmospheric turbulence caused by the heat of the earth's surface distorts the received image making object identification difficult. There are a number of approaches to mitigating the effect of the turbulent atmosphere and numerous documented algorithms in the literature. There are however no clear comparisons between algorithms and the metrics that could be used to compare the performance of these algorithms

In this paper three algorithms were implemented based on techniques described in the literature. A number of performance metrics were proposed and used to attempt to compare the performance of these algorithms. Human assessment was used to determine whether the metrics were providing realistic and pertinent information about the video scenes under consideration.

The performance metrics all seemed to provide valuable insight into the performance and natures of the algorithms. However it was found that the Laplacian sharpness metric was overly sensitive to high frequency noise and as such is not a useful metric when used in isolation.

Using the proposed metrics, we found that the Wiener filtering and Averaging algorithm does stabilize the geometric distortion present in the video, but it does so at the price of introducing further blurring. The Wiener filtering and Optical flow based methods seemed to provide stable and sharper results. Of these two algorithms the Lucas-Kanade based algorithm seemed to perform the best resulting in the most stable and sharp output out of the three algorithms.

The GPU implementation of the Lucas-Kanade based algorithm was fairly successful with only a few stages that required intensive redesign to allow for better exploitation of the parallel GPU architecture. The issue of recursion was easily solved using a ping pong approach and a vertex shader was used to implement the scattering operation used for dewarping.

The Averaging algorithm does, however, seem to have the lowest computational requirements of the three algorithms, but the GPU implementation of the Lucas-Kanade based algorithm did provide a significant speed increase and promises easy scalability to achieve real-time processing speeds in the future.

8 FUTURE WORK

The purpose of this study was to provide an entry point into the topic of mitigating the effects of atmospheric turbulence on video captured over a long range. As such there is much more possible work to be done to take the first steps presented in this study and produce a fully functional real world system to address the problem of heat shimmer.

A few specific areas that need improvement are as follows:

- Develop a more advanced dewarping scheme,
- Improve the accuracy of the optical flow methods,
- Develop a non-uniform deblurring scheme,
- Improve the GPU implementation.

9 REFERENCES

[1] W. Zhao, L. Bogoni, M. Hansen, "Video Enhancement by Scintillation Removal", *icme,* , 2001 IEEE International Conference on Multimedia and Expo (ICME'01), 2001, pp. 71.

[2] A. Bovik, Handbook of Image and Video Processing, Academic Press, San Diego, 2000.

[3] S.D. Ford[1], B.M. Welsh[1], M.C. Roggemann[2], "Constrained least-squares estimation in deconvolution from wave-front sensing", Optics communications, [1]Department of Electrical and Computing Engineering, Air Force Institute of Technology, 2950 P Street, Bldg 640, Wright-Patterson Air Force Base, OH 45433-7765, USA, [2]Department of Electrical Engineering, Michigan Technological University, 1400 Townsend Drive, 121 EERC bldg, Houghton, MI 49931-1295, USA. 1997.

[4] B.R. Frieden, "An exact, linear solution to the problem of imaging through turbulence", *Optical sciences centre,* University of Arizona, Tucson, AZ 85721, USA, 1997.

[5] G. Barnard. Restoration of Turbulence Degraded Images, Literature study report, Doc. No. QT-RA02-D-P047-001, Thales Advanced Engineering (PTY) LTD., October 1990.

[6] D.H. Frakes, J.W. Monaco, M.J.T. Smith, "Suppression of Atmospheric Turbulence in Video Using an Adaptive Control Grid Interpolation Approach", School of Electrical and Computing Engineering, Georgia Institute of Technology, Atlanta, GA, 2001.

[7] D. Li, R. Mersereau, D. H. Frakes, M. J. T. Smith, "New method for suppressing optical turbulence in video", Proc. EUSIPCO, 2005.

[8] G. Thorpe[1], A. Lambert[2], D. Fraser[2], "Atmospheric Turbulence Visualization through Image Time-Sequence Registration", [1]*Boeing Australia Limited,* 363 Adelaide Street, Brisbane, QLD 4000, Australia, [2]*School of Electrical Engineering,* University College, University of New South Wales, ADFA Canberra ACT 2600, Australia.

[9] B. Cong, "Encoding Neural Networks to Compute the Atmospheric Point Spread Function", Dept. of Computer Science, California State University, Fullerton, CA 92834 - USA.

[10] I. Goss-Ross, G. Barnard, B. Coetzer. *Restoration of Turbulence Degraded Images,* Algorithm

Development, Doc. No. RP-RA02-S-C047-003, Thales Advanced Engineering (PTY) LTD., March 1991.

[11] C. Max, "Introduction to Adaptive Optics and its History", American Astronomical Society 197th meeting, 2001.

[12] R. E. Hufnagel and N. R. Stanley, "Modulation transfer function associated with image transmission through turbulence media," Journal of the Optical Society of America A , vol. 54, pp. 52–61, 1964.

[13] E. C. Ifeacher, B. W. Jervis, Digital Signal Processing: A Practical Approach, Second Edition, Prentice Hall, Pearson Education Ltd. 2002.

[14] B.D. Lucas, T. Kanade, "An iterative image registration technique with an application to stereo vision", Proceedings of Imaging understanding workshop, pp. 121-130, 1981.

[15] B.K.P. Horn, B.G. Schunck, "Determining optical flow", Artificial Intelligence, vol. 17, pp. 185-203, 1981.

[16] J.L. Barron, D.J. fleet, S.S. Beauchemin, T.A. Burkitt, "Performance of Optical Flow techniques", CVPR, 1992.

[17] D. Luebke, "The Democratization of Parallel Computing", SC07 Presentation, 2007.

[18] Lighthouse 3D, GLSL Tutorial, http://www.lighthouse3d.com/opengl/glsl/, September 2008.

[19] R.S. Wright Jr, B. Lipchak and N. Haemel, OpenGL SuperBible 4th Edition, Addison Wesley Professional, 2007.

[20] R.J. Rost, OpenGL Shading Language 2nd Edition, Addison Wesley Professional, 2006.

[21] D Göddeke, GPGPU: Basic Math Tutorial, http://www.mathematik.uni-dortmund.de/~goeddeke/gpgpu/tutorial.html, September 2008.

[22] K. Moreland and E. Angel, "The FFT on a GPU", *SIGGRAPH/Eurographics Workshop on Graphics Hardware 2003 Proceedings*, pp. 112–119, 2003.

[23] B.D. Walters, "Comparison of two terrestrial atmospheric turbulence suppression algorithms", *IEEE Africon 2007*, ISBN: 978-1-4244-0987-7, pp. 1-7, Oct. 2007.

[24] C. J. Carrano, "Speckle imaging over horizontal paths", Proceedings of the SPIE —High Resolution Wavefront Control: Methods, Devices, and Applications IV, 4825, 2002.