LOOKUP TABLE BASED REAL-TIME NON-UNIFORMITY CORRECTION OF INFRARED SCENE PROJECTORS

Kenneth G. LeSueur[1], Emil Jovanov[2], and Aleksandar Milenkovic[2]

[1] US Army, Developmental Test Command
Redstone Technical Test Center
CSTE-DTC-RT-E-SA, Bldg. 4500
Redstone Arsenal, AL 35898-8052

[2]University of Alabama in Huntsville
Electrical and Computer Engineering
Engineering Building
Huntsville, AL 35899

Abstract

The state-of-the-art in laboratory testing of Infrared (IR) missile seekers and target acquisition systems is centered on advancements with IR Scene Projectors (IRSP). IR scene projection provides realistic and repeatable operational-type test scenarios in a controlled synthetic environment. When using resistor array scene projectors, there is an inherent variability in the micro-fabrication process for each emitter element that manifests as "fixed-pattern" noise, or non-uniformity. High Performance Computing (HPC) resources provide the means to correct for the non-uniformity in real-time.

Current correction techniques use straight-line approximations to interpolate between the data points available on each resistor element. We propose a novel technique based on memory lookup tables aimed to improve the fidelity of estimations between the data points. By providing a capability to pre-calculate the correction value of each resistor for each of the possible pixel values, a purely memory based table lookup can replace the current equation solving technique. This new technique could provide faster frame rates and more accurate correction of the IRSP scenes, increasing the level of validation in the laboratory testing of IR sensors.

1 Introduction

The U.S. Military depends on Infrared (IR) sensor technologies found on missile seeker platforms and target acquisition systems to accomplish the mission in the field (Figure 1). The testing and performance assessment of IR imaging systems has traditionally been accomplished using a combination of laboratory and field-testing. Both types of testing are necessary because each provides a unique set of test stimuli, all of which are required to adequately "exercise" a system. Laboratory testing involves precise, repeatable stimuli, which are used to measure a system's performance without expending expensive missile hardware or tying up valuable test ranges.

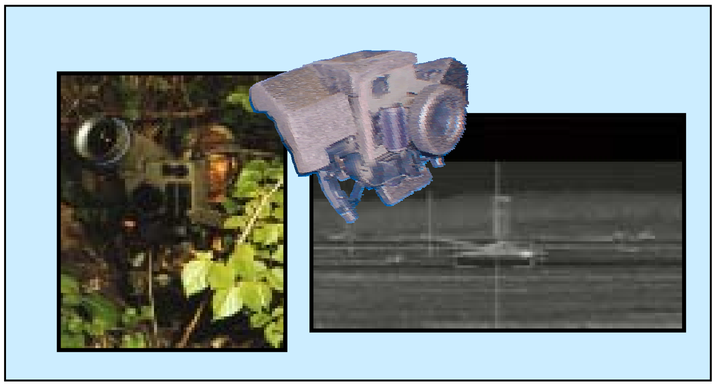


Figure 1. IR Missile and Target Acquisition System

The state of the art in laboratory testing of infrared missile seekers and target acquisition systems is centered on advancements with Infrared Scene Projectors (IRSP). Currently, the leading technology is the micro-resistor array. Such an array consists of several hundreds of thousands to one million small (~50 x 50 micron) resistors arranged in regularly spaced rows and columns on a silicon substrate (Figure 2). The resistors are electrically connected such that image information at video signal rates can be "written" to the array [5]. Electrical current flows through each resistor at a level proportional to the intensity of the input image at that location and is dissipated as heat, producing a "thermal" image on the array. This thermal image is re-imaged by a suitable optical collimating system and projected into the entrance aperture of the System Under Test (SUT).

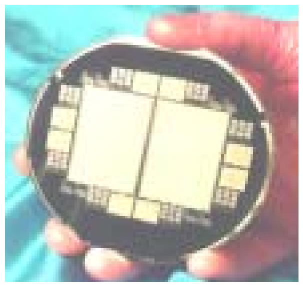


Figure 2. IRSP Wafer

A difficulty in this technology is that there can be a significant variation in the electrical resistance values for all of the micro-resistors [1][2][3]. This "non-uniformity" causes unwanted "artifacts" in the projected scene. In order to solve this problem, real-time non-uniformity correction (NUC) must be implemented. Figure 3 shows typical variations in projector element intensity versus drive voltage. The non-uniformity is largely caused by the foundry process of the wafer, but can also include Digital to Analog Converter (DAC) differences. The DAC non-uniformity is seen as different values on regularly spaced rows and columns in the projector output. To have a properly radiometrically calibrated scene the drive voltages for each of the resistors must be adjusted to produce a uniform IR output for a given commanded input.

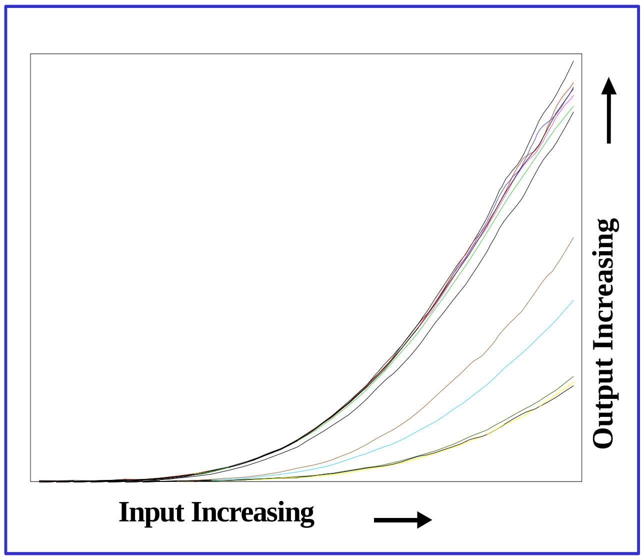


Figure 3. Resistor Characteristic Curves

Figure 4 shows an IRSP output where the top half has been corrected and the bottom half is projecting an uncorrected image.

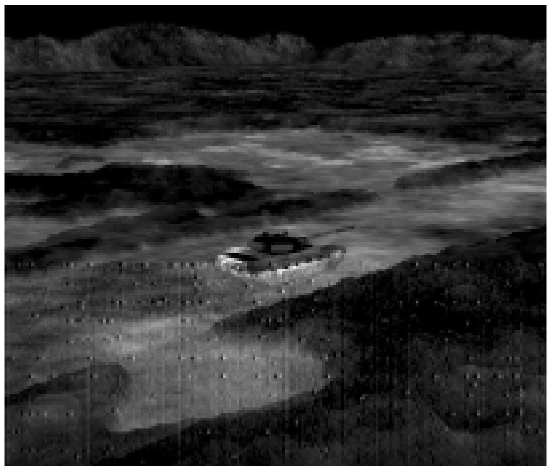


Figure 4. IR Image with Upper Half Corrected

Any usable NUC technique must be able to correct 1 Million pixels within a 30Hz (33 mSec) window. For Hardware-In-The-Loop (HWIL) laboratory testing, the frame rate is driven by the SUT and is non-negotiable. If the scene projector frame rate falls below that of the SUT, the test/simulation is invalid.

The rest of the paper is organized as follows. Section 2 surveys existing non-uniformity correction techniques, section 3 discusses the proposed solution. The experimental environment is given in section 4. Section 5 presents results of the experiments and section 6 concludes.

2 Existing Calculation Based Non-Uniformity Correction

To accomplish NUC, the first step is to gather radiometric performance data for each one of the resistors over the entire operating range [8][9]. Next, a NUC coefficient lookup table is generated for each of the projector pixels containing 16 data points. This table is used by one of the real-time NUC procedures to correct for the differences in resistor radiometric output.

One real-time NUC procedure uses custom dedicated electronics to provide a pipeline correction on each of the projector resistors as the scene data propagates through the projector drive electronics. The 16-point coefficient tables are loaded into the drive electronics memory. The electronics contain multiple Field Programmable Gate Arrays (FPGA) that access the coefficient values in memory and calculates the straight-line equation approximations given the input value from the IR video image.

The second existing solution uses general purpose processors to solve the NUC straight-line equations [4][7]. Using this approach, the video frames are copied from the computer graphics memory, after rendering, into the main memory where the image is corrected in real-time using parallel processing techniques. The corrected image is then copied back into the graphics memory to be rendered out the graphics pipeline. The processes are locked to the processors, which are given real-time priorities and isolated from system interrupts to accomplish the real-time correction.

Several drawbacks and limitations exist for each of the current solutions. Both methods suffer from accuracy limitations associated with the straight line approximations of the resistors characteristic curves. As seen in Fig. 3, the characteristic curves more resemble an exponential function. To correct a 1024x1024 projector array in real-time, using the general purpose processor procedure, 24 dedicated Silicon Graphics Incorporated (SGI™) MIPS R120000 processors are needed. Attempts to process higher order polynomial or exponential equations exceeds the frame time when using existing resources available at the Redstone Technical Test Center (RTTC). The custom electronics solution is limited to solving straight-line equations, is costly (~$250,000), and is a unique design for each type of projector system and drive electronics eliminating the possibility of reuse between different projector types.

3 Lookup Table Based Non-Uniformity Correction

The proposed solution is to replace the straight-line real-time computation based approach with a memory based NUC algorithm. Instead of calculating a straight-line approximation to the resistor characteristic curve, a 12 bit (4096 element) correction table is generated for each resistor in the array. The current IR scene generation tools and 3D IR databases are limited to 12 bit resolution. The value of the incoming video data for each pixel is an index into its associated correction table that contains a correction value for each one of the possible video levels. The correction table is calculated each time the projector system is calibrated (6 months to one year) using a non-real-time process.

By using a pre-processed correction lookup table, the interpolation between the data points is no longer limited to a straight-line approximation. A higher order polynomial or exponential line fit can be used, increasing the accuracy of the NUC. The memory based approach could also reduce the number of processors needed to meet the real-time frame rates and could possibly increase the achievable frame rates.

Typical computer generated IR imagery is analyzed to determine if scene content reveals trends that could be exploited to improve the memory based NUC real-time performance. The analysis reveals that many rows of video have image values that are repeated and are adjacent or within a few pixels of one another. Exploiting this knowledge of scene content could increase performance by reducing the number of cache misses if the correction lookup table is loaded into memory correctly.

In Figure 5, the projector incoming image value for each pixel is used as an index into the NUC lookup table associated with the resistor. The 12 bit video value addresses one of the 4096 possible entries in the precalculated NUC correction tables. Each of the 1,048,576 pixels will have a 4096 element correction lookup table that contains the NUC data. This data is essentially a photo negative of the resistor array non-uniformity.

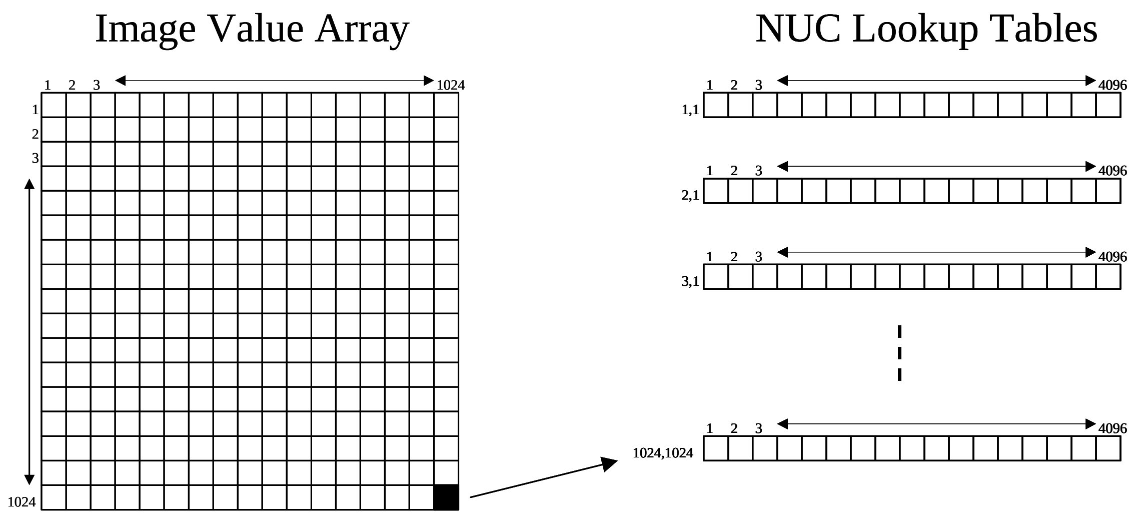
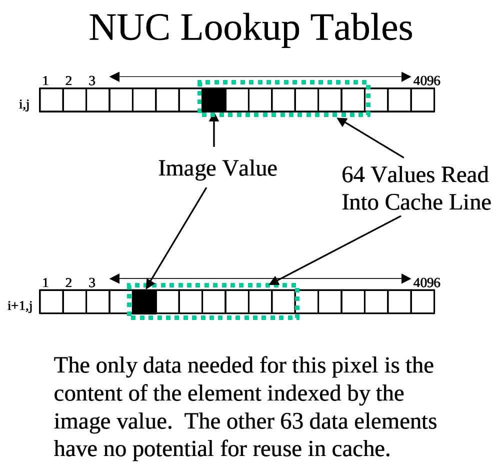


Figure 5. Video Value to NUC Table Relation

Knowing that the SGI architecture supports a 128 byte cache line and that the images have many repeated values in a row, a reduction in the number of cache misses can be achieved if the date tables are arranged as shown in Figure 6b [6]. If the data is arranged as shown in Figure 6a, there is no possibility for reuse of data in cache because the size of the lookup table greatly exceeds that of a cache line.

Where as shown in the orientation in Figure 6b, 64 pixel correction values are read or prefetched into cache, reducing the runtime of the algorithm when repeated pixel values are present in a row of video. An unsigned short integer (2 bytes) is used to store the data. Reading the 128 byte cache line pulls in the current and next 63 pixel correction values for the same intensity. The worst case scenario is when there is no repeated pixel values within the 63 adjacent pixels for each row. The worst case performance is the same for either memory orientation.



The only data needed for this pixel is the content of the element indexed by the image value. The other 63 data elements have no potential for reuse in cache.

Figure 6a. Horizontal NUC Table

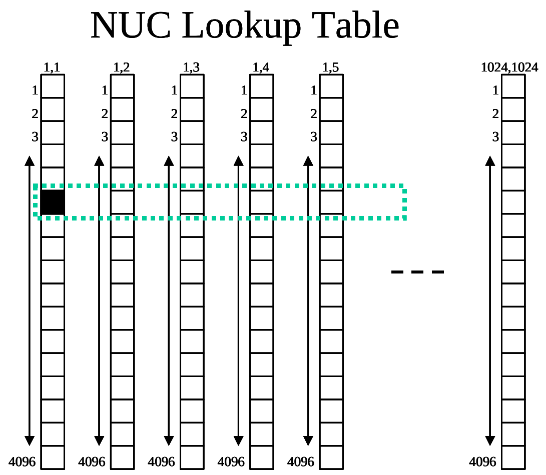


Figure 6b. Vertical NUC Table

In order to accomplish the memory based NUC all lookup table values must reside in main memory. The total space needed is:

$$TotalMemoryNeeded = (1024 * 1024) * 4096 * 2 = 8.59 Gbytes$$

The target machine to accomplish the memory based NUC is a Silicon Graphics Incorporated (SGI) Origin 2000 with 32 MIPS R12000 processors. The total system memory is 12 GB, enough to hold the lookup table.

The MIPS R12000 requires 75 to 250 clock cycles to service a secondary cache miss from main memory [10]. To determine if the approach is viable the worst case analysis is performed. The case where each video data element does not repeat within 63 adjacent pixels generates the worst case cache performance. Assuming the problem (1024x1024 resistors) is divided evenly across 32 processors, each processor is required to correct 32*1024 = 32,768 pixels per frame. The target system processor clock speed is 400MHz or 2.5 nano seconds.

$$FrameTime_A = 2.5x10^{-9} * 32{,}768 cache\_misses * 75 clockcycles = 6.144 mSec$$

$$FrameTime_B = 2.5x10^{-9} * 32{,}768 cache\_misses * 250 clockcycles = 20.48 mSec$$

Both of these values lie below the required 33 mSec requirement, making the memory based algorithm a potential solution.

4 Experimentation with Lookup Table Based Non-Uniformity Correction

The test computer system for the experimentation is a Silicon Graphics Origin 2000 system with 16 R12000 processors (8-400MHz and 8-300MHz). The total system memory is 8 GB and each processor has 8 MB of secondary cache. The operating system is IRIX 6.5 version 13 M. For the experiment, only one process and one processor is used which represents 1/32 of the problem size. The 1024 x 1024 image is divided into 32 equal row segments.

The worst case test/simulation scenario for the memory based NUC process is chosen as the test case. A missile flight profile is chosen that climbs in altitude and also closes on the target (and surrounding backgrounds) which ultimately causes every pixel in the scene to change from frame to frame. The scenario is also complicated by having the target cross the scene during the scenario furthering the change in the scene content from one frame to the next. Using the test scenario driver, sample 12 bit test video data is captured from the output of a real-time scene generator using an empirically generated 3D IR target and background database.

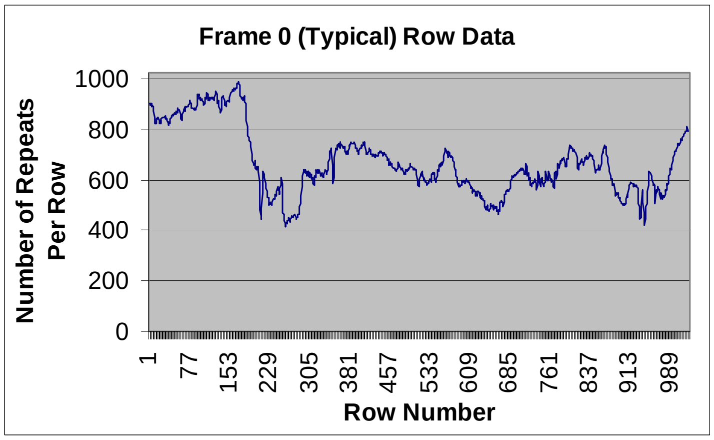
The computer system setup is modified to include real-time hooks. The real-time hooks in the test code moves the clocks to processor 0, gives the code the highest priority, sets the schedule time slice to the maximum of 10 seconds, assigns the process to the processor (#7 for this test), locks out all other processes, restricts operating system calls, turns the scheduler off, and locks the memory pages.

Timers are placed around the desired section of test code. The setup and functionality that would be provided by the scene generator are not included, only the execution time of the correction table lookup and assignments are tested.

The SGI R12000 architecture supports Processor Event Counters. There are 32 types of events that may be counted including Primary and Secondary cache misses. Perfex is used to measure the number of Secondary cache misses during the execution of the test code. Perfex is a SGI profiling tool used to tune algorithms [11].

5 Experimentation Results

The first step in the evaluation is to analyze the IR Scene data. The IR test video files are scanned and each time pixel (i) is the same value as pixel (i+1) for a given row, a counter is incremented. This data is captured for each row of each frame of the video sequence. The 1024 rows of each frame are averaged and presented as frame averaged data in Figure 9. There are a total of 234 frames in the video sequence. As seen in Figure 7 and 8 the number of value repeats can vary significantly from frame to frame and row to row. Frame 0, shown in Figure 7, is a typical frame within the test scenario. Frame 88 (Figure 8) is the worst case frame for the number of repeated data values.



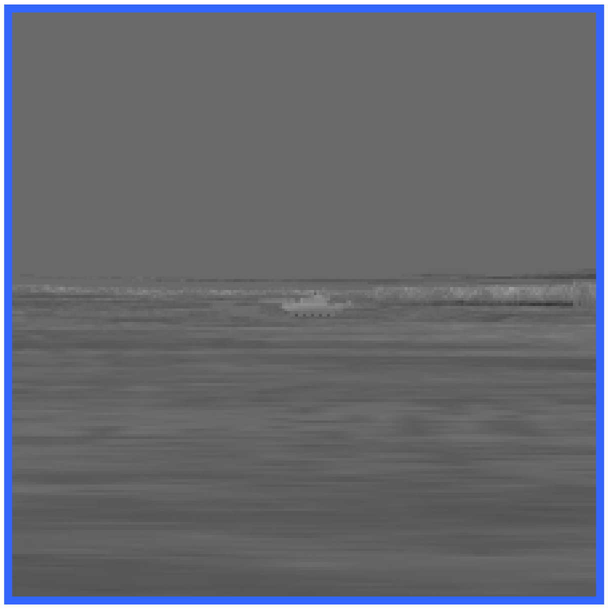
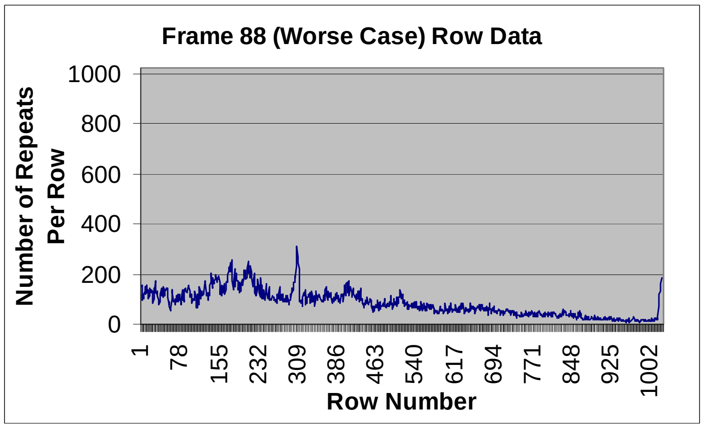


Figure 7. Number of pixel value repeats per row for Frame #0



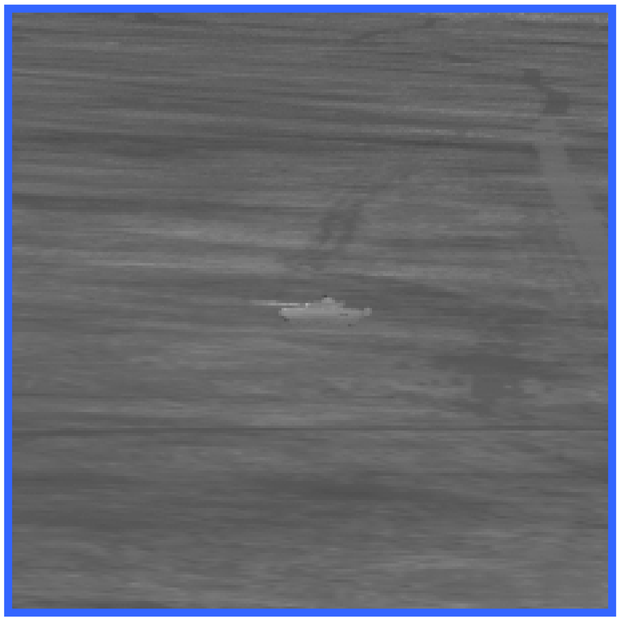


Figure 8. Number of pixel value repeats per row for Frame #88

When reviewing the scene data, it is noticed that there are sizable sections of the scene that contained a very large number of repeated values along the rows. Structures such as the sky (seen in the top part of frame # 0 in Figure 7) in the sensor field of view made many of the top rows of equal values. As part of a load balancing effort, the row assignments to the processors is modified to every 32nd row. This allowed each of the processors to take advantage of the highly repeated row values. These highly repeated row values will process much faster because of the decrease in the number of cache misses through the way the data is placed into memory. It is advantageous to balance the processes across the processors because the NUC procedure will be throttled by the slowest of these processes.

Figure 9 shows the averages of the individual frames plotted over time or frame sequence. The data shows that early in the flight there are many repeated values,

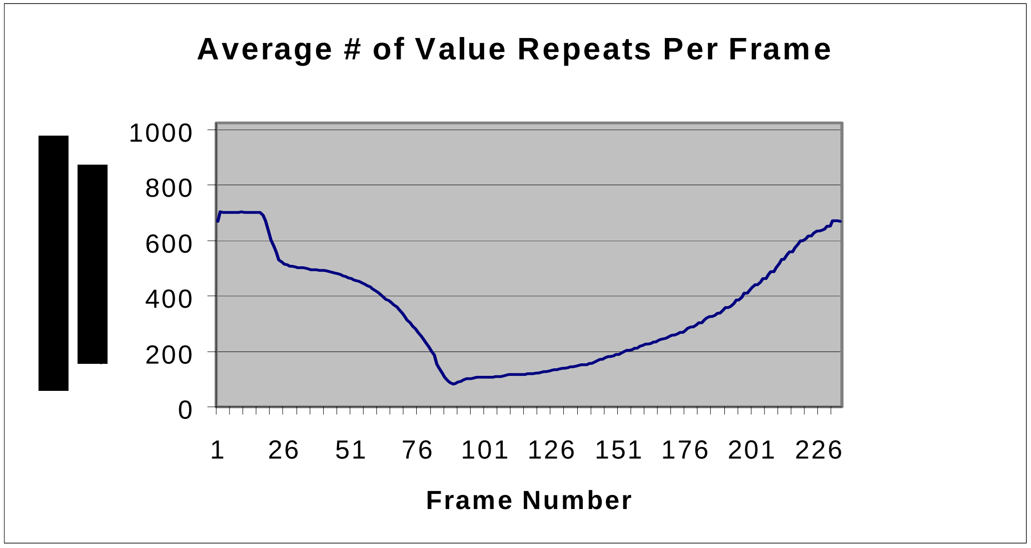


Figure 9. Average number of pixel value repeats for entire 8 second video sequence

primarily due to the sky in the background. Once the missile gains altitude and the sensor does not have sky in the background, the number of repeats drops fairly significantly. Later in flight the number of repeats grow due to the target taking up a larger portion of the scene. Frame 200 is seen in Figure 10 which shows the large uniform sections that make up the vehicle.

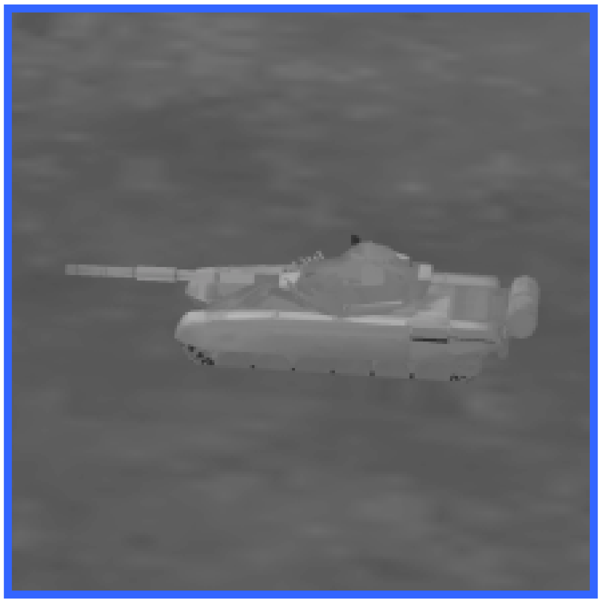


Figure 10. Typical terminal phase video frame

Next, the time needed to access the appropriate correction values is analyzed. The data shown in Figure 11 presents the results of the testing performed. When initiating the test, the profiling tool “perfex” is used to log the number of cache misses that occur during the execution of the program. The baseline values used are generated by commenting the test loop out of the code. The cache performance is then measured for frame #0 (typical frame), frame #88 (worst case for the scenario), and finally an artificially generated worst case frame that each element in the frame is incremented allowing no utilization of the SGI system 128 byte cache line.

	Time	Delta Cache Misses vs. Baseline
Baseline (no loop)	1-2 uSec	0
Frame 1	9.2 mSec	11,155
Frame 88	16.8 mSec	27,608
Worst Case	19.5 mSec	33,134

Figure 11. Timing and Cache Test Results

The timing results show that for all cases the Memory Based NUC technique solves the problem within the required frame time of 33 mSec. The execution time directly correlates with the number of cache misses.

6 Conclusions

The initial data shows that the Memory Based NUC approach is viable and shows potential for increased accuracy, faster frame rates, and possibly a reduction in the number of processors needed. The technique will be fully implemented, performing the non-uniformity correction for each of the different projector systems at RTTC.

7 References

1. M. Thomas, D. Newman, M. Frolli, D. Pritchett, “*Nonuniformity correction of cryogenic 512^2 emitter arrays: The 5 minute 5% NUC using FIESTA”*, Proceeding of SPIE – The International Society for Optical Engineering, Vol. 4366, 2001 Orlando, Technologies for Synthetic Environments: Hardware-in-the-loop Testing VI.

2. C. Stanec, L. Ewing, D. Moore, “*Considerations and algorithm development for scene-based nonuniformity correction (NUC) in dynamic infrared projectors”*, Proceeding of SPIE – The International Society for Optical Engineering, Vol. 3697, 1999 Orlando, Technologies for Synthetic Environments: Hardware-in-the-loop Testing IV. http://spie.org/scripts/toc.pl?volume=3697&journal=SPIE

3. E.M. Olson, R. L. Murrer, “*Non-Uniformity Correction of a Resistor Array Infrared Scene Projector”*. Proceedings of SPIE – The International Society for Optical Engineering, Vol. 3697, 1999 Orlando, Technologies for Synthetic Environments: Hardware-in-the-loop Testing IV. http://spie.org/scripts/toc.pl?volume=3697&journal=SPIE

4. Bruce E. Tucker and Kenneth W. Zabel, *“Parallel Image Processing Applications for Hardware-in-the-Loop Testing”*. High Performance Computing Workshop proceedings, June 1999. http://www.dtc.army.mil/hpcw/1999/tucker/index.html

5. K. R. Allred, E. E. Burroughs, W. Pickard, K. G. LeSueur, *“Real-time Synthetic Environments for HWIL Testing“*. High Performance Computing workshop proceedings, June 1999. http://www.dtc.army.mil/hpcw/1999/burrough/index.html

6. James Laudon and Daniel Lenoski “*The SGI Origin: A ccNUMA Highly Scalable Server”* http://www.sgi.com/origin/images/isca.pdf

7. J. Brent Spears and Brett N. Gossage, “*An Object-Oriented Software Framework for Execution of Real-Time, Parallel Algorithms”* 2001 International Conference on Computational Science, May 28

8. O. Williams, L. Swierkowski, “*Search for optimal infrred projector nonuniformity correction prodedures: II*”, Proceedings of SPIE – The International Society for Optical Engineering, Vol. 4027, 2000 Orlando, Technologies for Synthetic Environments: Hardware-in-the-loop Testing V. http://spie.org/scripts/toc.pl?volume=4027&journal=SPIE

9. R. Robinson, A. Irwin, J. Oleson, “*MIRAGE: Calibration Radiometry System”*, Proceedings of SPIE – The International Society for Optical Engineering, Vol. 4027, 2000 Orlando, Technologies for Synthetic Environments: Hardware-in-the-loop Testing V. http://spie.org/scripts/toc.pl?volume=4027&journal=SPIE

10. Silicon Graphics Incorporated, “*MIPS R10000 User’s Manual*”, version 2.0, Oct 2000. ftp://ftp.sgi.com/sgi/doc/R10000/User_Manual/t5.ver.2.0.book.pdf

11. Silicon Graphics Incorporated, “*Real Time Programming*”, Silicon Graphics Technical Education, August 1999.