2013 NDIA GROUND VEHICLE SYSTEMS ENGINEERING AND TECHNOLOGY SYMPOSIUM
MODELING & SIMULATION, TESTING AND VALIDATION (MSTV) MINI-SYMPOSIUM
AUGUST 21-22, 2013 - TROY, MICHIGAN

FINITE ELEMENT OPTIMIZATION FOR NONDESTRUCTIVE EVALUATION ON A GRAPHICS PROCESSING UNIT FOR GROUND VEHICLE HULL INSPECTION

Victor U. Karthik
ECE Department
Michigan State University
East Lansing, MI 48824.

Sivamayam Sivasuthan
ECE Department
Michigan State University
East Lansing, MI 48824.

Arunasalam Rahunanthan, Ph.D.
University of Toledo
Toledo, OH 43606.

Paramsothy Jayakumar, Ph.D. & Ravi S. Thyagarajan, Ph.D.
US Army TARDEC
Warren, MI 48397.

S. Ratnajeevan H. Hoole, D.Sc. (Eng.), Ph.D.
Michigan State University (MSU)
East Lansing, MI 48824.

ABSTRACT

Shape reconstruction for nondestructive evaluation (NDE) of internal defects in ground vehicle hulls using eddy current probes provides a rationale for determination of when to withdraw vehicles from deployment. This process requires detailed finite element optimization and is computationally intensive. Traditional shared memory parallel systems, however, are prohibitively expensive and have limited central processing units (CPUs), making speedup limited. So parallelization has never been done. However, a CPU that is connected to graphics processing units (GPUs) with effective built-in shared memory provides a new opportunity. We implement the naturally parallel, genetic algorithm (GA) for synthesizing defect shapes on GPUs. Shapes are optimized to match exterior measurements, launching the parallel, executable GA kernel on hundreds of CUDA™ (Compute Unified Device Architecture) threads to establish the efficiencies.

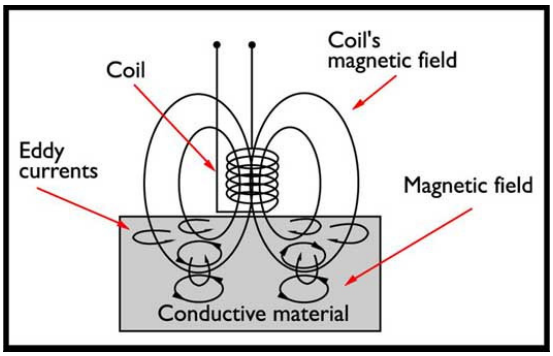
MOTIVATION AND BACKGROUND

When a hull or plate-plate weld in a ground vehicle is found to be defective, it is often wastefully taken out of service for repairs in present practice without determining if the defect warrants withdrawal. A procedure is required for defect characterization so that a decision to withdraw may be thought-out without compromise to war-fighter-safety; such a procedure would include behind-armor damage created by Improvised Explosive Device (IED) blasts. X-ray technology is not only dangerous, but also impracticable because of requiring readings within the tank's interior of X-rays that pass through the hull. NDE for corrosion, cracks, and other defects in ground vehicle armor employs eddy current probes for testing [1]. Recent changes have been towards composite materials for ground vehicles and remote measurement of defects through sophisticated measuring devices. These methods do not deal with corrosion and ignore the larger fleets of steel-hulled ground vehicles continuing to require the older NDE assessment. These conventional vehicles with steel hulls will remain in service for extended periods and be increasingly aging so that testing for, and characterizing defects are *a fortiori* important

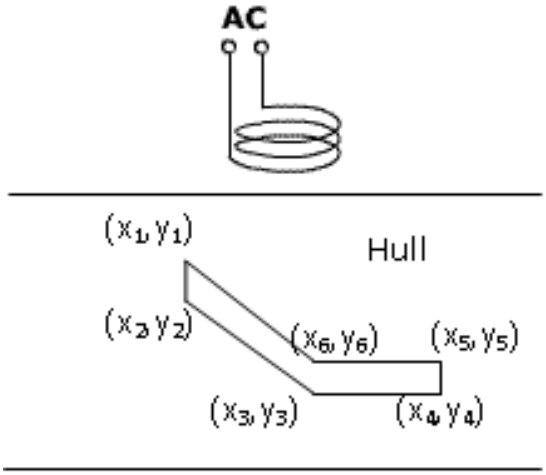
STATE OF THE ART – HOW IT IS DONE TODAY

Present methodology examines the response of the hull under test to a signal from an eddy current problem [1] (Fig. 1a). Any deviation from a known baseline response of a

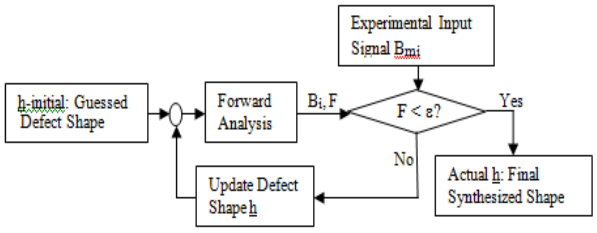
target with no defects is flagged as defective. Defect characterization is not done at present because it involves thousands of tedious eddy current computations in complex arithmetic and a three-dimensional magnetic vector potential with mixed finite elements [2] to model and optimize the defect shape h of Fig. 1b until the computed and measured fields match (Fig. 1c).



(a) The Test System

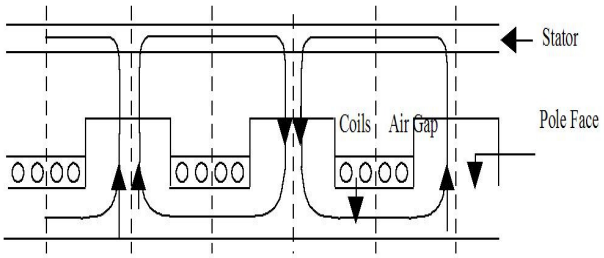


(b) Defect Modeled as $\underline{h}=\{x_1,y_1,\ldots,y_6\}$

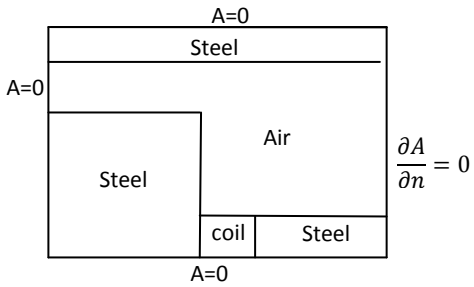


(c) Reconstructing h

Figure 1: Eddy Current Defect Testing System



a) Physical Alternating Pole System



b) Minimum Boundary Value Problem with Boundary Conditions on Magnetic Vector Potential A

Figure 2: Pole-Faces to be Shaped and Problem

THE TEST PROBLEM

Gradients-based optimization is difficult to program especially with eddy currents and would require a costly shared memory system usually limited to 16 processors because of technology limits [3]. Time is also a factor because these computations require large computer systems not readily portable for field testing. MSU's HPCC (High

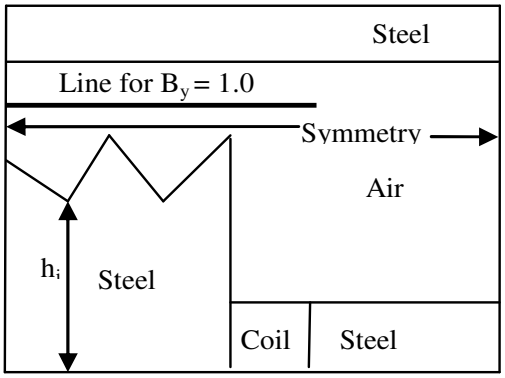


Figure 3: Parametric Model of Pole Face with Measuring Points

Performance Computing Center) Cluster provides an alternative powerful platform with parallel processing on the Graphics Processing Unit (GPU) [4, 5].

A primary question occupying our minds was which optimization method to use.

Our experience is that gradient techniques are fast in computation but slow to set up because of the programming time to have special mesh generators. Going by the literature, Preis *et al*. [6], staunch advocates of the zeroth order evolution strategy, merely say it is competitive with its higher order deterministic counterparts (which we take to mean the equivalent in time at best), but claim that its "robustness and generality" are superior. We agree with this observation because search methods will never see local minima as a problem. In contrast to what Preis *et al*. state, Simkin and Trowbridge [7] observe that simulated annealing and the evolution strategy take many more function evaluations. This is also our experience and we would add that the genetic algorithm works faster than simulated annealing. Haupt [8] advises that the genetic algorithm is best for many discrete parameters and the gradient methods for where there are but a few continuous parameters. However, we have gone up to 30 continuous parameters without problems. There seemed good reasons to go either way. But when the need for special mesh generators for high order methods is considered, a zeroth order method [9] is best.

For now therefore in this feasibility study, a zeroth order method like the genetic algorithm or simulated annealing has been selected to be the best choice. These involve two zeroth order methods where no derivative calculations are required. They avoid the difficult computation of objective function gradients and the need for special mesh generators [9].The simple test problem of Fig. 2 was selected. Fig. 2a shows the alternating poles pushing flux up and down in adjacent sets and Fig. 2b the minimal boundary value for the magnetic vector potential A with governing equation:

$$-\frac{1}{\mu}\nabla^2 A = J \qquad (1)$$

where μ is the magnetic permeability and J is the forcing current density in the coils [2].

Table 1: Hitting the 4 GB Limit at Matrix Size 10^4x10^4

Size	TE	SK(in MB)	TENSiP	SK(in MB)	TENSiS	SC(MB)
100	10000	0.038147	661	0.002521	1209	0.004612
400	16000	0.061035	8819	0.033642	2421	0.009235
900	810000	3.089905	28829	0.109974	5021	0.019153
1600	2560000	9.765625	67239	0.256496	10421	0.039753
2500	6250000	23.841858	130049	0.496098	17301	0.065998
6000	36000000	137.329101	374459	1.428448	41681	0.159000
8000	64000000	244.140625	657679	2.508846	54221	0.206836
10000	100000000	381.469727	1020099	3.891369	68021	0.259479

Key:
TE: Total Elements
TENSiP: Total number of elements that need to be stored in profile storage
TENSiS: Total number of elements that need to be stored in sparse storage
SK: Storage capacity

The object is to optimize the shape of the pole-face with GA. Thus as shown in Fig. 3, where the pole face is defined by 5 heights $h_1, h_2, \cdots, h_5$. These five heights h_i of Fig. 3 had to be optimized to get the pole shape giving us a vertical flux density 1 T along a line above the pole face. To define this formally the m measuring points were defined along the line show in Fig. 3. We allowed m to be changeable but used 3 which worked well. This performance requirement of 1 T was imposed through the least-square objective function

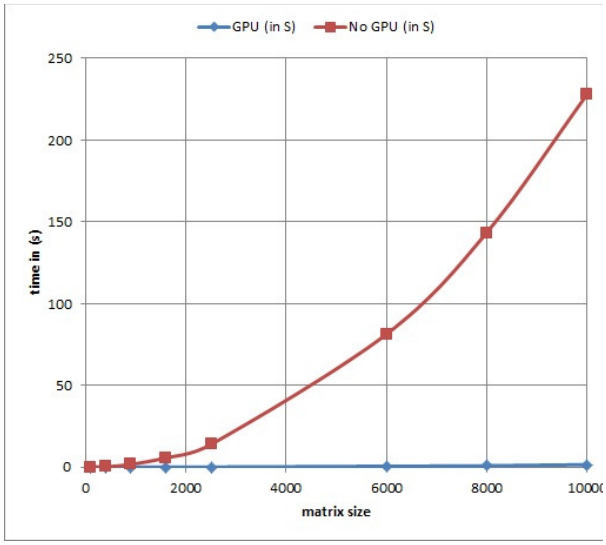


Figure 4: Timing by Element-by-Element Iterations (with GPU is the flat horizontal line

$$F = \sum_{i=1}^{m} \left(B_{Computed}^{y,i} - 1.0\right)^2 \qquad (2)$$

where B^y is the vertical flux density. Thus as $B_{Computed}^{y,i} \rightarrow 1$, F would tend to zero. The heights h, clearly would determine the value of F making

$$F = F(h_1, h_2, \cdots, h_5) \qquad (3)$$

Thus an optimization method [10-12] needs to be pressed into service to determine these values of $h_1, h_2, \cdots, h_5$ that would make F zero.

GPU PROCESSING – RESULTS

From the time Marrocco and Pironneau [10], and Gitosusastro, Coulomb and Sobonnadiere [11] applied gradients based finite elements to optimize magnetic circuits, papers have appeared using various optimization techniques [12]. Searching for the minimum requires several evaluations of the objective function F for each set $h_1, h_2, \cdots, h_5$ being tested. This means a new mesh and a large computational load. Shared memory parallel computers have been used [13, 14] but these are expensive, costly and the technology limits the number of processors that can be

used, usually to 16. However, since recently parallelization has been possible on a commonly available PC itself with 4 cores, where the code may simultaneously run on the multiple cores or the graphics processing unit (or GPU – to be more specific on an NVIDIA GPU with CUDA architecture [4]). There is however a severe memory limit – 4 GB at present. This would limit large problems as well as optimization where several problems may be attempted at once. We tested how the 4 GB translates into matrix size and the results of this study are shown in Table 1. It is seen that the limit is a 10,000x10,000 matrix size when the most efficient (sparse) storage scheme is employed [2].

For our purposes, as the project progresses, we need to solve eddy current problems in three dimensions. That is, if we are dealing with the magnetic vector potential as the state variable, it would have three components with each component a complex number. The implications to matrix storage would be severely limiting.

To overcome this we propose to use a method of the early 1980s when, working with the then new IBM PC 286, we had a memory limit of 612 kB which could not hold even a trivial matrix in memory. What we used to do was employ the Jacobi methods of matrix solution element-by-element. For example in solving the finite element matrix equation

$$[P]\{A\} = \{Q\} \tag{4}$$

just to explain the issues, the Gauss-Seidel iterative method [2], commonly used by power engineers, is an improvement on the older Gauss iterations where in each iteration m+1 we use the latest available values of the unknowns A, using equation i of (4) to compute A_i treating only A_i as the unknown and all the other variables as known and given by their latest values:

$$A_i^{m+1} = \frac{1}{P_{ii}}\left\{\sum_{k=1}^{i-1} P_{ik}A_k^{m+1} + \sum_{k=i+1}^{n} P_{ik}A_k^{m}\right\} \tag{5}$$

with obvious modifications for i =1 and i = n. Here at iteration m+1, computing A_i in the order i=1 to n, A is at values of iteration m+1 up to the (i-1)th component of {A} and at the values of the previous iteration m for values after i. The Gauss iterations using the old iteration's value for computing all A_i in iteration m+1 according to

$$A_i^{m+1}\frac{1}{P_{ii}}\{Q_i - \textstyle\sum_{k=1}^{i-1} P_{ik}A_k^{m} - \sum_{k=i+1}^{n} P_{ik}A_k^{m}\} \tag{6}$$

is inefficient in the context of sequential computations. But in this case of parallelization, if we can resort to this conventionally inefficient method, we may not form the matrix [P]. If [D] is the matrix [P] with all off diagonal elements eliminated, then Gauss's iterative method in this modified form gives,

$$[D]\{A\}^{m+1} = \{Q\} - [P - D]\{A\}^{m} \tag{7}$$

Thus without forming [P], the operation of the right hand side of (7) can be effected by taking each finite element in turn, computing the local 3x3 Dirichlet matrix $[P]^L$ and using that because

$$[P] = \textstyle\sum_{Elements}[P]^L \tag{8}$$

So as each $[P]^L$ is formed, the three values of $\{A\}^m$ may be taken and subtracted as in the right hand side of (7). We tested this and the results are shown in Fig. 4. We were able to go up to matrix sizes 100,000x100,000 and well beyond. We then applied the same idea to the more efficient incomplete Cholesky preconditioned conjugate gradients (ICCG) method [2] as laid out by Mahinthakumar and Hoole [15] for shared memory systems. In the GA kernel, there was no internal parallelization. In this work, the Incomplete Cholesky Conjugate Gradients matrix solver was parallelized on the GPU and we observed a speed-up of 146.351 for the matrix size 10,000x10,000 (Fig. 5).

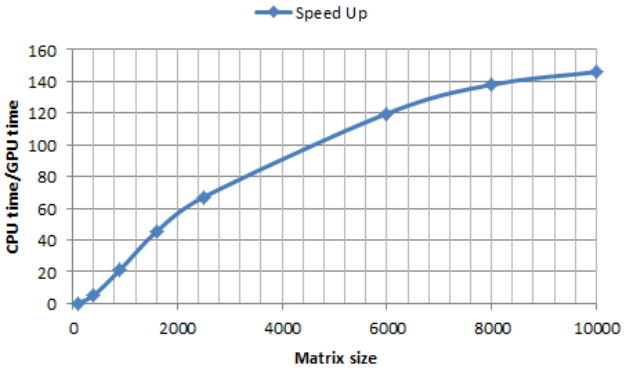


Figure 5: Conjugate Gradients Algorithm: Matrix Size Vs CPU time/GPU time

Table 2: Performance of GA

Pop. Size	Error (%)	Fitness F	Obj. Function	Time (s)
10	6.7	0.94×10^{-3}	1067.0	2.00
20	5.46	0.0018	554.55	3.88
50	2.09	0.9974	0.032	9.65

Table 3: Performance of SA

Iterations	Object Funct F	Time (s)
500	0.0448	14.25
1000	0.0146	28.22
4000	0.0282	119.12
40000	0.0075	1144.42

APPLICATION TO POLE FACE SHAPING

The pole face then shaped by optimization is shown in Fig. 6. GA was found to be effective converging faster than simulated annealing. The "fitness of the solution" for the GA, 1/(1+F), would approach 1 from below. The population size is the number of GA kernels launched as GPU threads. The optimal shape is in Fig. 5 and details in Tables 2 and 3 where the performances of the genetic algorithm and simulated annealing are respectively summarized. After 50 threads, the fitness was excellent. To compare the timings, for the genetic algorithm, a comparable object function is computed in Table 2 from the fitness F: $(1-F)/F$. It is seen that the genetic algorithm reaches a comparable object function value much faster than simulated annealing.

Seeking large population sizes and matrix size, we attempted to estimate the gain by using the GPU. The results are shown in Table 4. We observed these results for a simple electro-thermal coupled problem with the matrix size 204×204.

TEST PROBLEM OF INVERSION

To test our methodology and establish feasibility for these methods mooted, we need a special mesh generator modeling the crack defined by parametric location and shape. We created such a mesh generator just for the crack assessment problem, created a crack and computed the fields along measuring points outside the steel plate.

Thereafter, taking the results as the "measured field," we had to discover this crack as described by parameters {p}.

The results are shown in Fig. 7. We obtained a 95% match. In realistic cracks however, no model would be perfectly valid so the objective function will not go down to zero. However, the method is seen to be feasible.

CONLUSIONS

We conclude that using genetic algorithm optimization doing computations on a GPU for the reconstruction of hull defects has enormous benefits – speed through parallelism and convenience in avoiding computationally expensive gradients which are almost impossible to program as general purpose software.

Ultimately this problem would need to be done in 3-D where the computational load would be higher. Special mesh generators that maintain nodal connections would add to the burden. Additionally, many parameters must be allowed to get accurate crack shapes.

Table 4: Gain in using GPU

Population Size	No of Iterations	Time(s)		Ratio GPU/CPU Times
		CPU	GPU	
50	20	4824.94	432.38	11.16
100	20	9619.24	475.09	20.25
200	20	18322.20	667.85	27.43
400	20	37444.20	1480.32	25.29
512	20	49200.00	1756.43	28.01

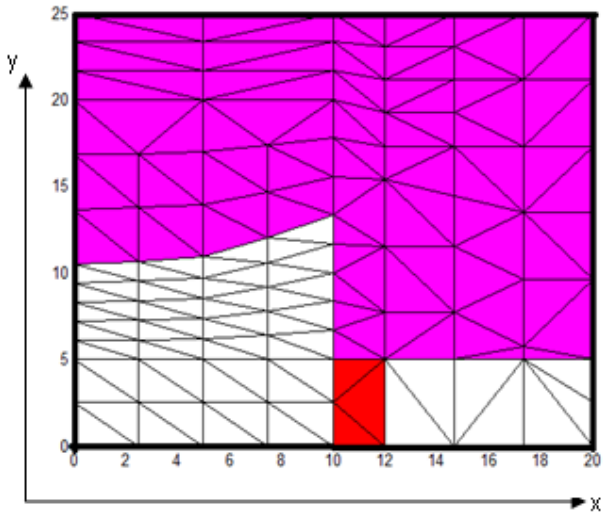


Figure 6: An Optimized Pole-face (5 parameters, 98 nodes, and 70 unknowns

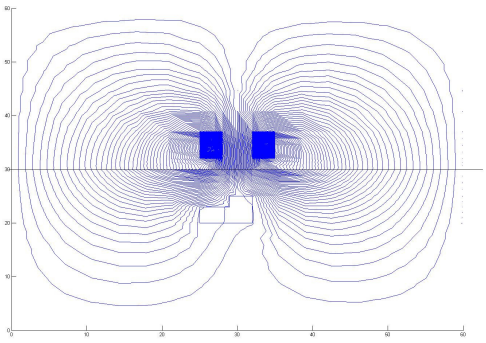


Figure 7: Crack Recreated from Exterior Fields using 10 parameters

DISCLAIMER

Reference herein to any specific commercial company, product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or the Dept. of the Army (DoA). The opinions of the authors expressed herein do not necessarily state or reflect those of the United States Government or the DoD, and shall not be used for advertising or product endorsement purposes.

REFERENCES

[1] A. Joshi, L. Udpa, and S. Udpa, "Use of higher order statistics for enhancing magnetic flux leakage pipeline inspection data," *Int. J. App. Electromag. & Mechanics*, Vol. 25, No. 1-4, pp. 357-362, 2007.

[2] S.R.H. Hoole, *Computer Aided Analysis and Design of Electromagnetic Devices*, Elsevier, NY, 1989.

[3] S.R.H. Hoole, "Finite Element Electromagnetic Field Computation on the Sequent Symmetry 81 Parallel Computer," *IEEE Trans. Magn.*, Vol. 26, No. 2, pp. 837-840, March, 1990.

[4] C. Cecka, A.J. Lew, and E. Darve, "Assembly of Finite Element methods on Graphics Processors," *Int. J. Num. Meth. Eng.,* Vol. 85, No.5, pp. 640–669, 2011.

[5] V. Ginting, F. Pereira, and A. Rahunanthan, "A Multi-stage Bayesian Prediction Framework for Subsurface Flows," *International Journal for Uncertainty Quantification*, 2013.

[6] K. Preis, C. Magele and O. Biro, "FEM and Evolution Strategies in the Optimal Design of Electromagnetic Devices," *IEEE Trans. Magnetics*, Vol. 26(5), pp. 2181-2183, Sept. 1990.

[7] J. Simkin and C.W. Trowbridge, "Optimization Problems in Electromagnetics," *IEEE Trans. Magnetics*, Vol. 27 (5), pp. 4016-4019, 1991.

[8] R. Haupt, "Comparison between Genetic and Gradient-based Optimization Algorithms for Solving Electromagnetics Problem," *IEEE Trans. Magnetics*, Vol. 31 (3), pp. 1932-1935, 1995.

[9] S. Krishnakumar and S.R.H. Hoole, "A Common Algorithm for Various Parametric Geometric Changes in Finite Element Design Sensitivity Computation," *J.Materials Proc. Tech.*, Vol. 161, pp. 368-373, 2005.

[10] A. Marrocco and O. Pironneau, "Optimum Design with Lagrangian Finite Elements: Design of an Electromagnet," *Computer Methods in Applied Mechanics and Engineering,* Vol. 15, pp. 277-308, 1978.

[11] S. Gitosusastro, J.L. Coulomb and J.C. Sobonnadiere, "Performance Derivative Calculations and Optimization Process," *IEEE Trans. Magnetics*, Vol. 25(4), pp. 2834-2837, 1989.

[12] Garret N. Vanderplaats, *Numerical Optimization Techniques for Engineering Design:* McGraw-Hill, 1984.

[13] S. Ratnajeevan H. Hoole, "Optimal Design, Inverse Problems and Parallel Computers," *IEEE Trans. Magn.*, Vol. 27, pp. 4146-4149, Sept. 1991.

[14] S. Ratnajeevan Hoole, *Finite Elements, Electromagnetics and Design*, Elsevier, Amsterdam, May 1995.

[15] G. Mahinthakumar and S. Ratnajeevan H. Hoole,"A Parallelized Element by Element Jacobi Conjugate Gradients Algorithm for Field Problems and a Comparison with other Schemes"," *Int. J. App. Electromag. in Matl*., Vol. 1, No. 1, pp. 15-28, July 1990.

.