

# Laboratory Measurements of Load-Transfer in Geosynthetic Reinforced Soils

by

**Samir Chauhan**

B.E. Civil Engineering, M.S. University, Baroda, India, 1986

M.Tech. Civil Engineering, Indian Institute of Technology, Kanpur, India, 1989

Submitted to the Department of  
Civil and Environmental Engineering  
in partial fulfillment of the requirements for the degree of  
Doctor of Philosophy in Civil and Environmental Engineering  
at the  
Massachusetts Institute of Technology

November 1995

*[February 1996]*

© 1995 Massachusetts Institute of Technology

All Rights Reserved

Signature of Author \_\_\_\_\_

Department of Civil and Environmental Engineering

Certified by \_\_\_\_\_

Prof. Andrew J. Whittle  
Thesis Co-Supervisor

Certified by \_\_\_\_\_

Dr. John T. Germaine  
Thesis Co-Supervisor

Accepted by \_\_\_\_\_

Prof. Joseph M. Sussman

MASSACHUSETTS INSTITUTE  
OF TECHNOLOGY

Chairman, Departmental Committee on Graduate Studies

FEB 26 1996

ARCHIVES

LIBRARIES



# Laboratory Measurements of Load-Transfer in Geosynthetic Reinforced Soils

by: Samir Chauhan

Submitted to the Department of Civil and Environmental Engineering  
on November 14, 1995, in partial fulfillment of the requirements for the degree of  
Doctor of Philosophy in Civil Engineering

## Abstract

This thesis describes the measurements of tensile stresses which develop in planar steel and polymeric inclusions due to shearing of the surrounding soil mass. The experiments are performed in a laboratory device, referred to as the Automated Plane Strain Reinforcement (APSR) cell, which has been substantially modified from a pre-existing prototype. The measurements of load-transfer between the soil and tensile reinforcing inclusions are evaluated and compared with analytical predictions from an elastic shear-lag model.

The APSR cell measures the maximum tensile stress that develops at the center of single planar inclusions of half-length,  $L/2 < 0.45\text{m}$  as the surrounding soil matrix is sheared in a plane strain compression mode. The cell is fully automated with eight feedback control loops that use digital PID control algorithms to control displacements of the platform jacks, the positions of the plane strain walls and the inclusion, and the confining air pressure. The feedback control system ensures that out-of-plane strains are less than 0.001% using pressurized water diaphragms within the side walls and holds the reinforcing inclusion to  $\pm 1\mu\text{m}$  of its reference position using a custom-built infrared sensing device. The hardware and software modifications carried out during the present research have greatly improved the reliability and efficiency of the APSR cell relative to a pre-existing prototype (Larson; Ph.D., 1992), and provided a capacity for the routine measurement of load-transfer.

A comprehensive program of tests has been performed to measure the tensile stresses in planar steel and nylon 6/6 sheet reinforcements embedded in Ticino sand at a relative density of 75%, and sheared at a confining pressure of 31kPa. These experiments demonstrate the effects of inclusion length on the load-transfer behavior for two linear elastic materials whose axial stiffness represent upper and lower bounds on typical reinforcing products used in practice. The results show that tensile stresses develop in the reinforcement due to lateral strains within the soil matrix, and can be related to the maximum shear stress level imposed during shearing. At the same shear stress, tensile loads in planar inclusions depend on the axial stiffness and length of the inclusions. The elastic shear-lag model proposed by Abramanto (Ph.D., 1993) provides a reliable framework for interpreting the APSR measurements. The analyses using secant approximations to the measured non-linear stress-strain properties of the unreinforced soil matrix are in excellent agreement with the measured behavior of planar inclusions in the APSR cell.

APSR tests also have been carried out using typical polyester, woven geogrid products used in practice. Although the maximum tensile stresses are similar in magnitude to nylon 6/6 reinforcements (of comparable axial stiffness), there are significant differences in the behavior observed as a function of the inclusion length. Further experiments show that the load transfer for grids of the same axial stiffness are affected by the arrangements of the transverse grid elements. These results suggest that the grid geometry affects the tensile load distribution and provides greater reinforcing efficiency compared to a planar reinforcement. Further studies are necessary to validate these findings.

The performance of an instrumented, geogrid-reinforced soil wall at working load levels has been reported in a well documented case study in Canada. The maximum tensile strains in these reinforcements are in good agreement with results obtained under similar conditions in the APSR cell, and demonstrate the direct practical application of the load-transfer measurements described in this thesis.

Thesis Co-Supervisor: **Dr. Andrew J. Whittle**  
Title: Associate Professor of Civil & Environmental Engineering  
Thesis Co-Supervisor: **Dr. John T. Germaine**  
Title: Principal Research Associate



# Acknowledgments

I am grateful to many people who have, in so many ways, contributed to the work described in this thesis. I take this opportunity to thank the following individuals and organizations:

My thesis co-supervisor Professor Andrew Whittle whose continued commitment to the APSR research allowed me to pursue and eventually complete this work. I am grateful to him for his counsel and advise throughout this study.

Dr. John Germaine, also my thesis co-supervisor, has been a constant source of ideas and innovative solutions. I have learned a lot about taming the chaotic world of lab work from his hands-on approach to research. I am truly thankful for this unique learning experience.

Prof. Charles C. Ladd and Prof. Frederick J. McGarry for serving on my thesis committee and reviewing my work. Stephen Rudolph for excellent machining jobs in spite of my often messy and incomplete drawings. Mary Elliff for her help in editing pieces of my correspondence and this thesis at various times. Tom Collins of Huesker Corporation for providing grid samples used in the final phase of the work.

This research was sponsored in part by the Army Research Office and the National Science Foundation. Their support is gratefully acknowledged.

Life at MIT is unthinkable without the help, companionship and support I have received from my friends and fellow students. I thank: Doug Larson and Mauricio Abramento for their previous work which provided valuable background for this study; Marika Santagata, Dante Legaspi, Juan Pestana, Ute Schran, Ajay Gupta, and Parin Gandhi for sharing gossips and laughter, and being there in good and not so good times.

My friends outside MIT, including Piyush Singhal, Dipak Shah, Ketki Shah, and Tushar Nandwana, provided much needed distractions and occasional realization that there is a whole different world outside MIT. Thank you so much.

My lovely wife Falguni deserves a large part of the credit for this dissertation. Her love, understanding, and ever smiling face gave me strength during the most difficult phase of the work. She has also been a great help in proof reading numerous drafts of the thesis.

Finally, I dedicate this thesis to my mother who made countless sacrifices so that I could realize my dreams. I can never hope to repay even a small part of what she has done for me.



# Table of Contents

<b>Abstract</b>	<b>3</b>
<b>Acknowledgments</b>	<b>5</b>
<b>Table of Contents</b>	<b>7</b>
<b>List of Tables</b>	<b>13</b>
<b>List of Figures</b>	<b>15</b>
<b>Chapter 1: Introduction</b>	<b>23</b>
1.1. Background	23
1.2. Laboratory Studies of Soil-Reinforcement Interaction	25
1.3. Thesis Scope and Organization	29
<b>Chapter 2: Description of the APSR Cell</b>	<b>39</b>
2.1. Introduction	39
2.2. APSR Cell Design	41
2.2.1. Design Requirements	42
2.2.2. Size Considerations and Design of Structural Box	43
2.2.3. Major Principal Axis	45
2.2.4. Intermediate Principal Axis	46
2.2.5. Minor Principal Axis	48
2.2.6. Reinforcement System	49
2.3. Instrumentation	53
2.3.1. Boundary Displacements	53
2.3.2. Boundary Stresses	56
2.3.3. Inclusion Load and Position	57
2.3.4. Data Acquisition and Reduction	58
2.4. Computer Control Systems	60
2.4.1. Background	60
2.4.2. APSR Control Hardware	63
2.4.2.1. Pressure-Volume Controller	63
2.4.2.2. DC Motor and Servo Amplifier	64

2.4.2.3. Electro-pneumatic Regulator	65
2.4.2.4. AIC-16 Analog-Digital Card	66
2.4.2.5. AOC-8 Digital-Analog Card	67
2.4.2.6. Digital Computer	67
2.4.3. APSR Control Software	68
2.4.3.1. Structure of FlexCAT	69
2.4.3.2. APSR Interface	72
2.4.4. Control Algorithm and Performance	72
2.4.4.1. Platform Pistons	73
2.4.4.2. Active Plane Strain	74
2.4.4.3. Air Pressure	75
2.4.4.4. Reinforcement Position	75
<b>Chapter 3: Test Materials and Procedures</b>	<b>115</b>
3.1. Introduction	115
3.2. Test Materials	115
3.2.1. Ticino Sand	116
3.2.2. Reinforcing Materials	117
3.2.2.1. Steel Sheet	117
3.2.2.2. Nylon 6/6	118
3.2.2.3. Fortrac Geogrid	119
3.3. Sample Preparation and Test Procedure	121
3.3.1. Sample Preparation	121
3.3.1.1. Rubber Membrane	122
3.3.1.2. Reinforcement	122
3.3.1.3. Cell Preparation	125
3.3.1.4. Sand Deposition	127
3.3.2. Test Procedure	129
3.3.2.1. Application of Initial Boundary Stresses	129
3.3.2.2. Plane Strain Shearing	131
3.4. Shear Behavior of Unreinforced Ticino Sand	132
3.4.1. Effect of Air Pressure Confinement	133
3.4.2. Effect of Rigid $\sigma_1$ Boundary	135
3.4.3. Effect of Vacuum	136
3.4.4. Equivalent Elastic Soil Parameters	137
3.5. Conclusions	138



<b>Chapter 4: Load-Transfer in Planar Inclusions</b>	<b>165</b>
4.1. Introduction	165
4.2. Effects of Design Improvements on Test Data	166
4.3. Presentation of APSR Test Data	167
4.3.1. APSR Tests on Steel Sheet Inclusions	168
4.3.1.1. Tensile Stress in the Reinforcement	168
4.3.1.2. Shear Behavior of Reinforced Ticino Sand	171
4.3.2. Tests on Nylon 6/6 Sheet Inclusions	172
4.4. Discussion and Evaluation of Results	174
4.4.1. Outward Movements of Inclusion	175
4.4.2. Initiation of Tensile Load-Transfer	176
4.4.3. Tensile Stress in Inclusion	179
4.4.3.1. Influence of Inclusion Length	179
4.4.3.2. Effects of Inclusion Stiffness	180
4.4.4. Effect of Inclusion on Behavior of Soil Matrix	181
4.5. Comparison with Shear-lag Analysis	182
4.5.1. Review of Shear-lag Analysis	182
4.5.2. Selection on Input Parameters	187
4.5.3. Interpretation of APSR results	190
4.6. Conclusions	192
<b>Chapter 5: APSR Tests on Geogrids</b>	<b>229</b>
5.1. Introduction	229
5.2. Presentation of Grid Test Results	230
5.2.1. Overview	230
5.2.2. Tensile Loads in Inclusions	231
5.2.3. Shear Behavior of Ticino Sand	233
5.2.4. Effects of Transverse Members	235
5.3. Discussion and Interpretation APSR Data	236
5.3.1. Selection of Input Parameters for Shear-lag Analyses	237
5.3.2. Effects of Inclusion Stiffness and Length	238
5.3.3. Influence of Grid Geometry on Load-Transfer	239
5.4. Conclusions	241
<b>Chapter 6: A Geogrid Reinforced Wall Case Study</b>	<b>267</b>
6.1. Introduction	267

6.2. RMC Reinforced Retaining Walls	268
6.2.1. Description of Geogrid Reinforced Walls	269
6.2.2. Material Properties	270
6.2.3. Test Configurations and Procedures	271
6.2.3.1. Test 1: Incremental Panel Facing	272
6.2.3.2. Test 2: Propped Panel Facing	273
6.2.4. Instrumentation	273
6.3. Presentation of Test Results	274
6.3.1. Test 1: Incremental Panel Facing Wall	274
6.3.2. Test 2: Propped Panel Facing Wall	276
6.4. Results of Prediction Exercise	276
6.4.1 Methods of Analysis	277
6.4.2 Comparison with Measured Strains	278
6.5. APSR Estimates of Reinforcement Strains	279
6.6. Discussion and Conclusions	287
<b>Chapter 7. Summary and Conclusions</b>	<b>313</b>
7.1. Summary	313
7.2. Conclusions	317
7.2.1. Shear Behavior of Unreinforced Ticino Sand	317
7.2.2. Load-Transfer in Planar Inclusions	318
7.2.2.1. Measurements of Inclusion Tensile Loads	318
7.2.2.2. Comparison with Shear-lag Predictions	320
7.2.3. Behavior of Geogrids in the APSR Cell	321
7.2.4. Geogrid Reinforced Wall Case Study	323
7.3. Suggestions for Future Work	323
7.3.1. Improvements in APSR Cell	324
7.3.2. Experimental Program	325
<b>References</b>	<b>331</b>
<b>Appendix A: Refinement of Cell Design</b>	<b>341</b>
A.1. Introduction	341
A.2. Reinforcement Positioning System	341
<b>Appendix B: Digital Feedback Control</b>	<b>349</b>
B.1. Introduction	349
B.2. Control Objectives and Performance Criteria	351

B.3. Control Algorithms	353
B.3.1. Proportional Control	353
B.3.2. Integral Control	353
B.3.3. Derivative Control	354
B.3.4. Direct Digital Control	354
B.4. Tuning Procedures	356
<b>Appendix C: Data Converter Cards</b>	<b>363</b>
C.1. Introduction	363
C.2. Analog-to-Digital Conversion	364
C.2.1. AIC-16 Architecture	366
C.2.1.1. AD1170 Chip	367
C.2.1.2. Analog Section	368
C.2.1.3. Digital Section	370
C.2.2. Operation of AIC-16 Card	371
C.2.2.1. Communicating with AD1170	372
C.2.2.2. Retrieving Conversion Data	374
C.2.2.3. Advanced Features of AD1170	375
C.3. Digital-to-Analog Conversion	377
C.3.1. AOC-8 Architecture	378
C.3.1.1. AD767 Chip	379
C.3.1.2. Digital Interface	380
C.3.2. Operation of AOC-8 Card	380
C.3.3. Calibration Procedure	382
<b>Appendix D: APSR Control Software</b>	<b>395</b>
D.1. Introduction	395
D.2. APSR Interface	396
D.2.1. Sigma 1 Panel	396
D.2.2. Sigma 2 Panel	397
D.2.3. Sigma 3 Panel	398
D.2.4. Reinforcement and Graph Panels	398
D.3. FlexCAT User's Guide	399
D.3.1. Overview	399
D.3.2. System Menu	401
D.3.3. Setup Menu	402

D.3.4. Calibration Menu	404
D.3.5. Test Menu	406
D.3.6. DataAcq Menu	406
D.3.7. Functions Menu	407
D.4. Program Listings	425
D.4.1. File SUPPORT.H	425
D.4.2. File SYSTEM.H	427
D.4.3. File FLEXCAT.C	431
D.4.4. File APSR.C	501

# List of Tables

2.1	The APSR Cell Transducers	77
2.2	Specifications for the APSR Cell Linear Actuators	78
2.3	Specifications for the DC Servo Motors	78
2.4	Specifications for the Electro-Pneumatic Controller	79
2.5	Digital-to-Analog Card Channel Allocation and Range	79
2.6	The APSR Feedback Loop Parameters	80
3.1	Properties of Ticino Sand (after Larson, 1992)	141
3.2	Relative Densities of Dense Ticino Sand Specimens	141
3.3	Glass Transition Temperatures and Elastic Moduli of Some Common Polymers	142
3.4	Dimensions and Mechanical Properties of Steel and Nylon Sheet Reinforcement	143
3.5	Dimensions and Mechanical Properties of Selected Fortrac Geogrids (Source Huesker, 1994)	143
3.6	Shear Properties of Dense Ticino Sand in the Original APSR Cell (after Larson, 1992)	144
3.7	Shear Properties of Dense Ticino Sand in the Modified APSR Cell	144
4.1	Summary of APSR Proof Tests with Steel Sheet Reinforcements	195
4.2	Summary of APSR Tests with Steel and Nylon 6/6 Sheet Reinforcements	196
4.3	Magnitude of Reinforcement Centerline Loads and Strains for the Longest Planar Inclusions used in the APSR Cell	197
4.4	Linear Regression Analyses of Tensile Loads in Inclusions as Functions of the Average Lateral Strain	198
4.5	Offset Strain Corrections for Short Inclusions in the APSR Cell	199

4.6	Input Parameters for Shear-lag Predictions of Load Pickup in Steel and Nylon 6/6 Sheet Reinforcements	199
4.7	Regression Coefficients Describing Variation of Elastic Material Parameters for Unreinforced Ticino Sand	200
5.1	Summary of APSR Tests with Fortrac Grid Reinforcements	243
5.2	Summary of APSR Measurements of Load-Transfer in Fortrac 110/30-20 Geogrid Reinforcements	244
5.3	Linear Regression Analyses of Tensile Loads in Fortrac 110/30-20 Grids as Functions of Average Lateral Strain in Soil	245
5.4	Geometric Characteristics of Intact and Modified Fortrac 110/30-20 Grid Inclusions	245
5.5	Input Parameters for Shear-lag Predictions of Load Pickup in Fortrac Geogrid Reinforcements	246
6.1	Comparison of Ticino Sand with the Sand Used in RMC Trial Walls	289
6.2	Comparison of Some of the Design Methods Used by the RMC Prediction Exercise Participants	290
6.3	Summary of Predicted Reinforcement Strains for Propped Panel Wall	291
6.4	Summary of Predicted Reinforcement Strains for Incremental Panel Wall	292
6.5	Summary of Measured Reinforcement Strains	293
C.1	Important Specifications of the AD1170 Converter	383
C.2	The AD1170 Address Space Utilization	384
C.3	Interpretation of the AD1170 Status Byte	384
C.4	The AD1170 Command Set	385
C.5	AD1170 Data Format and Format Code	386
C.6	Preset Integration Periods	386
C.7	Important Specifications of the AD767 D/A Converter	387
D.1	Properties of the FlexCAT Objects	409

# List of Figures

1.1	Boundary Conditions in Pullout Tests (after Larson, 1992)	33
1.2	General Arrangement of Soil Reinforcement in Shear Box Tests	34
1.3	Typical Results of Direct Shear Tests on Leighton Buzzard Sand Reinforced by a Single Steel Grid at Different Orientations (Jewell, 1980)	35
1.4	Boundary Conditions in the Unit Cell.	36
1.5	Effect of Reinforcements on External Stress-Strain Behavior Measured in Plane Strain Shear Test (McGown et al., 1978)	37
2.1	Geometry of the APSR Ideal Plane Strain Reinforced Soil Element	81
2.2	Conceptual Design of the APSR Cell (after Whittle et a., 1991)	82
2.3	Cross Section Through the APSR Cell	83
2.4	Photographs of the APSR Cell and Control Hardware	85
2.5	The APSR Cell Platform Assembly	87
2.6	Detailed View of the Sidewall Positioning System	88
2.7	The APSR Support Cart and Sidewall Referencing System	89
2.8	Exploded View of Plane Strain Sidewalls (after Larson, 1992)	90
2.9	The APSR Cell Airbag Assembly	91
2.10	Reinforcement Entry Slot and Support Arch	92
2.11	Reinforcement Control Mechanism	93
2.12	Principle of Infrared Position Detection	94
2.13	Calibration Curve for Infrared Sensor	95
2.14	Infrared Positioning System Assembly	96
2.15	The APSR Cell Instrumentation	97
2.16	Circuit Diagram for Infrared Sensor Assembly	98

2.17	The APSR Junction Box	99
2.18	Types of Control Systems	100
2.19	Platform Feedback Control Loop	101
2.20	Schematic of the APSR Pressure-Volume Controller	102
2.21	Simplified Block Diagram of the DC Servo Motor Controller	103
2.22	Structure of FlexCAT Program	104
2.23	FlexCAT Flow Control Diagram	105
2.24	The APSR User Interface	107
2.25	Calibration Curves for Platform Pistons and Air Pressure Regulator	109
2.26	Performance of Platform Piston Control System	110
2.27	Performance of Plane Strain Sidewalls	111
2.28	Performance of Air Pressure Control System	112
2.29	Calibration and Performance of Reinforcement Control System	113
3.1	Grain Size Distribution for Ticino Sand (after Larson, 1992)	145
3.2	Molecular Structure of Nylon 6/6 and Creep Behavior of Plastics	146
3.3	Typical Stress-Strain Results for In-Isolation Tension Tests on Nylon 6/6	147
3.4	Stress-Strain Behavior of Fortrac Woven Geogrids (Source Huesker, 1994)	148
3.5	Dimensional Stability of Fortrac Grid Under Sustained Loading (Source Huesker, 1994)	149
3.6	APSR Specimen Membrane Parts and Layout	150
3.7	Details of Full Length Planar Reinforcement	151
3.8	Connection between Planar Reinforcement and Specimen Membrane	152
3.9	Details of Geogrid and Indicator Flap Connection	153
3.10	Photographs of the APSR Cell Preparation	155
3.11	The APSR Cell Raining Apparatus	157



3.12	Stress-Strain Response of Dense Unreinforced Ticino Sand in the Original APSR cell (after Larson, 1992)	158
3.13	Comparison between the Shear Behavior of Dense Ticino Sand with and without the Airbag	159
3.14	Repeatability of Externally Measured Shear Behavior in the Modified APSR cell	160
3.15	Variation of the Parameter b for Unreinforced Dense Ticino Sand	161
3.16	Effects of Rigid Major Principal Stress Boundary on the Measured Shear Behavior of Dense Ticino Sand	162
3.17	Effect of Overconsolidation due to Application of Vacuum during Sample Preparation	163
3.18	Variation of Secant Shear Modulus and Poisson's Ratio for Unreinforced Medium Dense Ticino Sand	164
4.1	Comparison of Measurements Obtained from the Original and Modified APSR Cell	201
4.2	Improvements in the Control of Steel Reinforcement Position Using the Infrared Sensor	202
4.3	Repeatability of Tensile Stress Measurements in the Improved APSR Cell	203
4.4	Initiation of Load Pickup in 180 mm Long Steel Sheet Inclusion	204
4.5	Effect of Inclusion Length on Tensile Stresses Measured for Steel Sheet Reinforcements in the APSR Cell	205
4.6	Relationship Between Inclusion Load and Average Lateral Strain in Soil Matrix for Steel Sheet Reinforcements	206
4.7	Effectiveness of Steel Sheet Reinforcement in Reducing Lateral Deformations in the Sand Matrix	207
4.8	Externally Measured Shear Behavior of Dense Ticino Sand Reinforced with Steel Sheet Inclusions	208
4.9	Beginning of Load Transfer in the Nylon 6/6 Sheet Reinforcement	209
4.10	Effect of Inclusion Length on Tensile Stresses Measured for Nylon 6/6 Sheet Reinforcements in the APSR Cell	210
4.11	Inclusion Tensile Loads as Functions of the Average Lateral Strain in Soil Matrix for Nylon 6/6 Sheet Reinforcements	211

4.12	Relationship between the Applied Stress Ratio and Lateral Strain in APSR Tests on Nylon 6/6 Reinforcements	212
4.13	Externally Measured Shear Behavior of Dense Ticino Sand Reinforced with Nylon 6/6 Sheet Reinforcements	213
4.14	Outward Movements of Nylon 6/6 Inclusions during the Initial Phase of the APSR Test	214
4.15	Possible Effects of Arching at Rear Wall of the Cell on Displacement Field within Reinforced Soil Specimen	215
4.16	Development of Tensile Loads for Steel and Nylon 6/6 Sheet Inclusions in the APSR Cell	216
4.17	Effect of Inclusion Length on Centerline Tensile Loads for Steel and Nylon 6/6 Sheet Inclusions	217
4.18	Comparison of Load-Transfer Measurements for Steel Inclusions Obtained from the Original and Modified APSR Cell	218
4.19	Plane Strain Geometry for an Element of Reinforced Soil (after Abramento, 1993)	219
4.20	Distribution of Tensile Stresses in a Planar Inclusion (after Abramento, 1993)	220
4.21	Effect of Inclusion Length and Stiffness on the Maximum Load Transfer Ratio (after Abramento, 1993)	221
4.22	Sensitivity of the Shear-lag analysis to the Values of Elastic Input Parameters	222
4.23	Variation of Secant Shear Modulus and Poisson's Ratio for Unreinforced Medium Dense Ticino Sand	223
4.24	Shear-lag Predictions vs. Measured Inclusion Tensile Loads for Steel Sheet Inclusions of Different Lengths	224
4.25	Effects of Inclusion Length on Predicted and Measured Load Transfer for Steel Sheet Reinforcements	225
4.26	Comparison of Shear-lag Predictions and APSR Measurements of Load-Transfer for Nylon 6/6 Sheet Reinforcements	226
4.27	Prediction and Measurement of Load-Transfer for Nylon 6/6 Sheet Reinforcements	227
5.1	Repeatability of Externally Measured Tensile Loads in Fortrac 110/30-20 Grid Inclusions	247

5.2	Effect of Inclusion Length on Tensile Loads Measured in the APSR Cell for Fortrac 110/30-20 Grid Reinforcements	248
5.3	APSR Measurements of Load-Transfer for Fortrac 80/30-20 Grid Reinforcements	249
5.4	Relationship between Exit Point Tensile Loads and Lateral Strains in Soil Matrix for Fortrac 110/30-20 Grid Reinforcements	250
5.5	Relationship between Applied Stress Ratio and Lateral Strain in Soil for Fortrac 80/30-20 Grid Reinforcements	251
5.6	Shear Behavior of Medium Dense Ticino Sand Reinforced with Fortrac 80/30-20 Grid Inclusions	252
5.7	Effectiveness of the Longest Fortrac 110/30-20 Grid Inclusion in Reducing Lateral Deformations in the Sand Matrix	253
5.8	Effects of Fortrac 110/30-20 Grid Reinforcements on Boundary Measurements of Shear Behavior of Ticino Sand	254
5.9	Effects of Transverse Members on Load-Transfer Behavior of Two Fortrac 110/30-20 Geogrid Inclusions	255
5.10	Relationship between Inclusion Tensile Loads and Average Lateral Strain for Intact and Modified Grid Reinforcements	256
5.11	Effect of Transverse Grid Elements on the Average Lateral Strains in the Soil Matrix	257
5.12	Effect of Transverse Ribs on Externally Measured Shear Behavior of Dense Ticino Sand for 193 mm Long Grid Inclusions	258
5.13	Effect of Transverse Ribs on Externally Measured Shear Behavior of Dense Ticino Sand for 360 mm Long Grid Inclusions	259
5.14	Comparison of Shear-lag Predictions and APSR Measurements of Load-Transfer for Fortrac 110/30-20 Grid Reinforcements	260
5.15	Comparison of Shear-lag Predictions and APSR Measurements of Load-Transfer for Fortrac 80/30-20 Grid Reinforcements	261
5.16	Effects of Inclusion Length on Predicted and Measured Load Transfer for Fortrac 110/30-20 Grid Reinforcements	262
5.17	Comparison of Shear-lag Predictions and APSR Measurements of Centerline Loads in Modified and Intact Fortrac 110/30-20 Reinforcement	263
5.18	Hypothetical Differences between Distributions of Tensile Stress within Planar and Geogrid Inclusions	264

5.19	Externally Applied Stress Ratio and Inclusion Tensile Load as Functions of Axial Strain for Smooth and Punched Nylon Sheets	265
5.20	Comparison between Load-Transfer Behavior of 360 mm Long Smooth and Punched Nylon 6/6 Sheet Inclusions	266
6.1	Principal Components of RMC Retaining Wall Test Facility (after Bathurst et al., 1987)	295
6.2	General Arrangement for Incremental Panel Wall Construction	296
6.3	Isochronous Load-Strain Curves for Tensar SR2 Geogrids at 20° C (after Bathurst et al., 1987)	297
6.4	Summary of Panel Positions for Incremental Panel Wall Test (after Bathurst et al., 1987)	298
6.5	Distribution of Geogrid Strains for Incremental Panel Wall Test	299
6.6	Comparison of Grid Strains from Strain Gage Readings and Extensometer Data for Incremental Panel Wall Test (after Bathurst et al., 1987)	300
6.7	Summary of Panel Movements for Propped Panel Wall Test (after Bathurst et al., 1987)	301
6.8	Summary of Strain Gage Readings for Propped Panel Wall Test	302
6.9	Reinforcement Strains in Incremental Panel Wall after 12 kPa Surcharge for 100 hours (after Bathurst and Koerner, 1987)	303
6.10	Reinforcement Strains in Incremental Panel Wall after 50 kPa Surcharge for 1000 hours (after Bathurst and Koerner, 1987)	304
6.11	Reinforcement Strains in Propped Panel Wall after 12 kPa Surcharge for 100 hours (after Bathurst and Koerner, 1987)	305
6.12	Reinforcement Strains in Propped Panel Wall after 50 kPa Surcharge for 1000 hours (after Bathurst and Koerner, 1987)	306
6.13	Application of APSR Cell for Estimating Tensile Loads in RMC Geogrid Reinforcements	307
6.14	Non-Linear Failure Envelope for RMC Sand	308
6.15	APSR Measurements of Load-Transfer for 360 mm Long Nylon Inclusion	309
6.16	Effect of 360 mm Long Nylon 6/6 Inclusion on Externally Measured Lateral Strain in the Soil Matrix	310

6.17	Computation of Maximum Strains in the Backfill	311
A.1	Connection from Arch to Reinforcement Grips (after Larson, 1992)	345
A.2	The Original Designs for Referencing the Inclusion Position	346
A.3	Data Showing Limitations of the Original Inclusion Referencing System	347
B.1	Block Diagram Representation of Control Systems	359
B.2	Step Response of a Typical Feedback Control System	360
B.3	Open Loop Response to a Step Function	361
C.1	The Ideal Transfer Function of a 3-Bit A/D Converter	389
C.2	Block Diagram of the AIC-16 Analog-to-Digital Converter Card	390
C.3	Architecture of the AD1170 and Voltage-to-Frequency Conversion	391
C.4	Pin Assignments of the IBM PC-XT Bus	392
C.5	Block Diagram of the AOC-8 Digital-to-Analog Converter Card	393
C.6	Basic R-2R Ladder Network and the AD767 Block Diagram	394
D.1	The APSR User Interface	411
D.2	The FlexCAT Graph Display Window	413
D.3	The Layout of the System Setup Dialogue Box	415
D.4	The Sensor Calibration Window	417
D.5	The Loop Calibration Window and Controls	419
D.6	The A/D Card Test Display	421
D.7	The FlexCAT Data Acquisition File Header	423



# Chapter 1: Introduction

## 1.1. Background

The term 'geosynthetic' refers to a wide range of polymeric material products, such as woven and non-woven textiles, high strength strips and grids, and membranes, which are extensively used to enhance the performance of soils in a wide range of civil engineering projects. Geosynthetics have been successfully used for a variety of purposes including containment, reinforcement, filtration, and drainage. In the reinforcement function, layers of geosynthetic materials are placed in a soil mass in order to provide tensile strength to soil which is otherwise relatively strong in compression but weak in tension. During the past two decades, geosynthetics have increasingly been used for soil reinforcement in retaining walls, foundations, slopes, and embankments. Over the years, advances in manufacturing techniques have steadily produced geosynthetic reinforcing materials with higher axial stiffness and lower creep.

Although, many reinforced-soil structures have been safely constructed using geosynthetic materials and are performing well, the use of geosynthetics as reinforcement has, historically, preceded the development of suitable methods for analysis and design (Rowe and Ho, 1993). Most analytical and design methods are either empirical in nature or are based on limit equilibrium approaches which do not consider deformations and make gross approximations of the interactions between the constituent materials. The mechanisms of interaction between the soil and the reinforcing inclusion which determine the magnitude of the loads carried by the reinforcement are not well understood, particularly for extensible reinforcements with

non-planar geometry, such as grids (Jewel et al., 1984). It is clear that more wide spread acceptance of geosynthetics depends upon the availability of validated, rational methods for the selection of reinforcing materials. Reliable predictions of the loads carried by geosynthetic reinforcing materials under working stress conditions are essential because all polymeric materials exhibit time dependent stress-strain properties (creep) to a varying degree. The rate at which creep strains occur in geosynthetic materials strongly depends on the fraction of the ultimate tensile strength mobilized under the sustained loading (e.g. Ward, 1971).

For the past five years, an integrated program of experimental measurements and analytical modeling has been carried out at MIT with the overall objective of improving the fundamental understanding of the composite behavior of the soil reinforced with geosynthetic materials. This research is based on the premise that the tensile stresses transferred onto a reinforcing inclusion within the soil mass can be predicted from known (interpretable) properties and geometry of the soil and reinforcement (Whittle et al., 1991). An analytical framework has been developed to model the load-transfer for a planar inclusion (Abramento and Whittle, 1993). The analysis is based on shear-lag type solutions which have been used previously in the mechanics of composite materials. The analysis expresses stresses in the reinforcement and soil matrix as closed-form, analytical functions of the soil and reinforcing material properties and geometry, and hence, provides physical insight into the mechanism of load-transfer in reinforced soil.

In addition to the analytical work, a new laboratory composite element shear device has been developed (Larson, 1992; Whittle et al., 1992) in order to obtain accurate measurements of load-transfer which can be used to evaluate predictive capabilities and limitations of the shear-lag analysis. The device, referred to as the



Automated Plane Strain Reinforcement (APSR) cell has the unique capability of measuring the maximum tensile stress that develops at the center of a single planar inclusion due to shearing of the surrounding soil in plane strain compression. The APSR cell simulates the mechanical interaction that occurs in a reinforced soil at working load levels under realistic and interpretable boundary conditions, and hence represents a significant advance over existing measurements of load-transfer.

## 1.2. Laboratory Studies of Soil-Reinforcement Interaction

Larson (1992) conducted a detailed review of the laboratory investigations of soil-reinforcement interaction reported in the literature. This section summarizes the important findings of this survey that provided rationale for the APSR cell design. The existing laboratory experiments designed for the local measurement of soil-reinforcement interactions can be roughly divided into three categories: i) pull out tests, ii) shear box tests, and iii) plane strain tests.

Pull out tests which simulate the action of the reinforcing inclusion as an anchorage have been widely used to investigate shear strength of the soil-reinforcement bond (Juran et al., 1988; Farrag et al., 1993). A geosynthetic inclusion confined in the soil is pulled out, usually at a constant rate of displacement, while the applied load and displacements of the inclusion are recorded. The test results are interpreted assuming that: a) the full frictional resistance is mobilized over the embedded surface of the geosynthetic and b) the normal traction acting at the interface is equal in magnitude to the boundary stress applied on the soil. In practice, these assumptions are overly simplistic and there are a number of limitations associated with pull out test: a) the

measurements are affected significantly by the boundary conditions of the test, b) deformations of extensible reinforcements can produce progressive failure and hence, test results are affected by the length of the inclusion. Figure 1.1 illustrates the variations on the basic pull out box design have been devised to overcome intrinsic limitations associated with boundary conditions. Based on a survey of available test data, Juran et al. (1988) concluded that the large number of parameters which influence pull out resistance make systematic comparisons of test results difficult. More recently, Farrag et al., (1993) have showed that significant variability exist in pull out load-displacement response of geogrids due to differences in equipment and testing procedures.

Modifications of the conventional box shear test has been used by a number of researchers to measure soil-reinforcement interaction characteristics (Jewell, 1980; Palmeria, 1987; Shewbridge and Sitar, 1989). Figure 1.2a shows the geometry of the Direct Shear Apparatus (DSA) used by Jewell (1980). The cell has dimensions 254x152x152 mm and contains a single reinforcement oriented at an angle  $\theta$  to the vertical direction. The specimen is consolidated under vertical stress usually applied through a rigid platen. During shear, the platen is locked to the top half of the box to prevent rotation and ensure symmetry about the horizontal shear plane. The measurements include the external stress-strain-strength behavior of the composite 'element' and the distribution of strains within the soil and inclusions using techniques of radiography (Jewell, 1980) and photoelasticity (Dyer, 1985). Figure 1.3 shows measured data for one set of DSA tests using dense Leighton Buzzard sand ( $D_r = 90\%$ ) reinforced by a steel grid oriented at  $-30^\circ \leq \theta \leq 60^\circ$  together with data for the unreinforced sand. The figure shows that for inclusions oriented at  $\theta > 0^\circ$ , there is a significant increase in the peak shear resistance, with maximum improvement  $\tau_r/\tau_{ur} = 1.55$ , measured for an inclusion rotated at  $\theta = 60^\circ$ . In contrast, there is a 10%

reduction in peak shear resistance when  $\theta = -30^\circ$ . The data also show that the reinforcement has little influence on the initial response of the composite element for horizontal displacement  $\bar{x} \leq 0.2$  mm with significant change in shear resistance for horizontal displacements in the range  $0.4 \leq \bar{x} \leq 0.7$  mm. Tensile stresses in the reinforcement are either inferred from the measured deformations of markers located on the inclusion or interpreted at the peak shear resistance using limit equilibrium analysis. Overall, box shear experiments represent a major contribution in understanding the complex interactions between the soil and reinforcement. However, the tests have a number of limitations:

1. The box shear apparatus imposes highly non-uniform stress and strain conditions and generate large shear distortions within a relatively small volume of the soil. The presence of the inclusion which intersects the failure plane introduces further non-uniformity.
2. The scale effects associated with the size of the shear zone and length of the inclusion make it difficult to apply the results of the test to prototype structures.
3. The reinforcement acts in both tension and bending resulting in a complex interaction which is difficult to interpret.

In order to overcome some of the non-uniformity problems associated with box shear tests, Hayashi et al. (1988) have used the direct simple shear apparatus with reinforcing bars embedded at various angles,  $\theta$ , in Toyura sand (Figure 1.2b). The results indicate that the reinforcement increases the shear resistance of the sand by 10 to 50% for  $-15^\circ \leq \theta \leq 20^\circ$ .

McGown et al. (1978) have used a plane strain unit cell which contains a single planar inclusion oriented in various directions to the major principal stress (Figure 1.4).

The cell accommodates a soil specimen of dimensions 152x102x102 mm which is confined by a constant vacuum pressure. The major principal stress is applied through rigid top and bottom platens with the test performed at a constant displacement rate. The rigid plane strain walls are lubricated and incorporate a thick glass platen to permit photographic measurements of internal strain. Tests were conducted using dry Leighton Buzzard sand reinforced with aluminum foil and mesh as well as extensible geotextiles. The results were analyzed in terms of boundary stresses and strains and also internal deformation patterns. Figure 1.5 shows typical stress ratio-axial strain measurements for tests with horizontal inclusions ( $\theta = 0^\circ$ ) on medium dense ( $D_r = 58$  to 65%) and loose ( $D_r = 11$  to 18%) Leighton Buzzard sand, respectively. The unreinforced, dense sand reaches a peak stress ratio,  $R = 8.1$ , at an axial strain,  $\epsilon_1 = 2.3\%$ , followed by post peak strain softening associated with shear planes formed within the specimen. The addition of inclusions, particularly the aluminum mesh and non-woven fabric (T-140), cause large changes in the externally measured composite behavior, while the behavior for the thin aluminum foil is similar to the unreinforced sand. Although the aluminum mesh and geotextile inclusions increase the peak shear resistance (e.g.,  $R_{max} = 13.0$  and 10.5 for the mesh and T-140, respectively), the reinforcements have little effect on the strain required to mobilize peak shear strength conditions. McGown et al. (1978) used internal measurements of soil strains to show that the mesh and geosynthetic reinforcements inhibit lateral deformations in the soil, and hence, reduce the tendency of the soil to dilate. The unit cell is useful for studying the overall composite material behavior but lacks the important ability to measure the tensile forces that develop in the reinforcement. Although the effects of inclusion properties and geometry on the shear behavior of soil-reinforcement composite have been studied extensively using a variety of shear devices, there is no underlying framework for estimating the strength properties as a function of the properties of the

constituent materials (soil and reinforcement) and geometry. As a result, measurements of composite behavior in small scale laboratory tests, such as the unit cell and direct simple shear apparatus (Hayashi et al. 1988), cannot be used to estimate the performance of field-scale reinforced structures.

It can be concluded that, most existing laboratory tests are directed mainly towards providing design parameters describing soil-reinforcement interaction in limit equilibrium analyses. They are not well suited for estimating load-transfer characteristics (i.e., stresses which develop within the reinforcement), especially at working stress levels. Primary limitations of existing tests are due to: a) non-uniformity of stress and strain within the soil, b) the complex boundary conditions imposed in the test, and c) lack of direct measurement of loads carried by the reinforcement. The APSR composite element test was specifically designed to overcome these limitations and provide direct, reliable measurements of load-transfer under working stress conditions.

### 1.3. Thesis Scope and Organization

This thesis describes the measurements of tensile stresses which develop in steel and polymeric inclusions due to shearing of the surrounding soil in a plane strain compression mode using the APSR cell. Larson (1992) obtained preliminary measurements of load-transfer for steel sheet using the prototype APSR device. However, the APSR cell design has been extensively modified in order to improve sensitivity and reliability of the test. Chapter 2 describes the mechanical design of various APSR sub-systems in detail. The chapter also provides a detailed account of the hardware and software aspects of the feedback control system which maintains the

necessary boundary conditions in the cell and concludes with a presentation of results showing the overall performance of the computer control system in the modified APSR cell.

Chapter 3 describes testing procedures used in the APSR research and presents the engineering properties of the matrix (soil) and reinforcing materials used in the present test program. The chapter also describes the measurements and interpretations of the shear behavior of unreinforced Ticino sand in the APSR cell.

Load-transfer measurements have been obtained for a series of APSR tests on dry Ticino sand reinforced with planar inclusions made from steel and Nylon 6/6. Chapter 4 establishes the reliability and repeatability of these measurements and evaluates the important trends in the load-transfer data. The discussion focuses on the effects of the reinforcement length and stiffness on the magnitude of tensile stresses in the planar reinforcements. The final part of the chapter includes a detailed comparison between the shear-lag predictions of load-transfer and the data from the APSR tests to show how the original linear, elastic shear-lag solution can be adapted to account for measured non-linear behavior of the matrix (sand) material.

APSR tests also have been carried out using typical polyester, woven geogrid products used in practice. Chapter 5 describes these measurements and demonstrates the similarities and differences between load-transfer behavior of geogrids and planar inclusions in the APSR cell through comparisons with the results for Nylon 6/6 reinforcements and shear-lag predictions of inclusion stresses.

The APSR cell offers a new experimental capability for evaluating the performance of prototype-scale geosynthetic-reinforced structures under working stress conditions. Chapter 6 illustrates the use of APSR measurements of load-transfer for

estimating the maximum reinforcement loads within a reinforced soil wall under working stress conditions, and compares these estimates with measurements for a well documented case study.

Chapter 7 gives a summary of the main conclusions and contributions of the thesis together with recommendations for future research using the APSR cell.





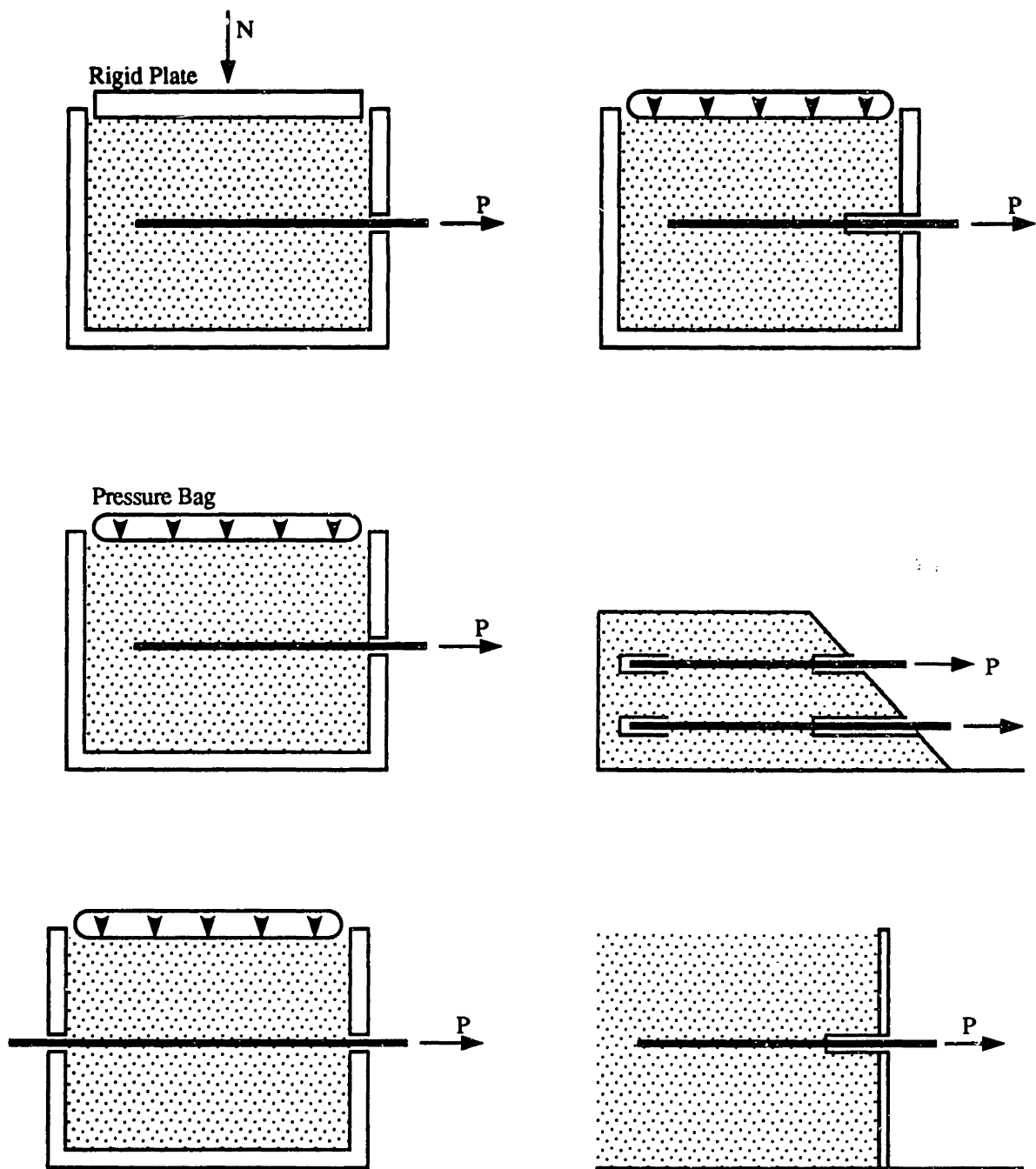
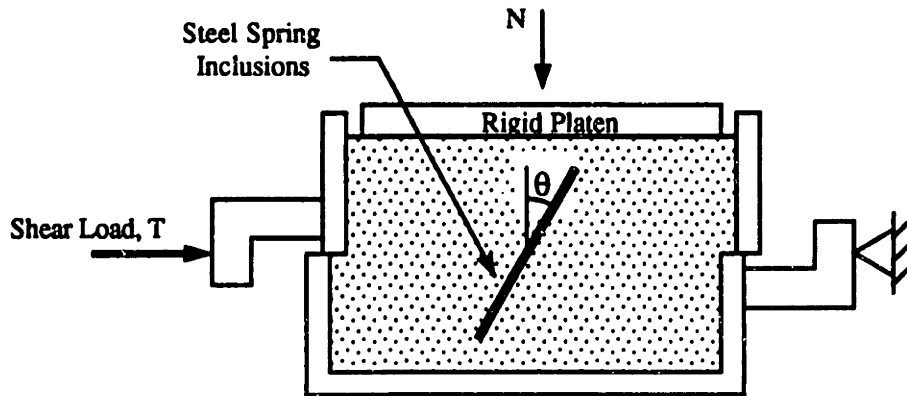
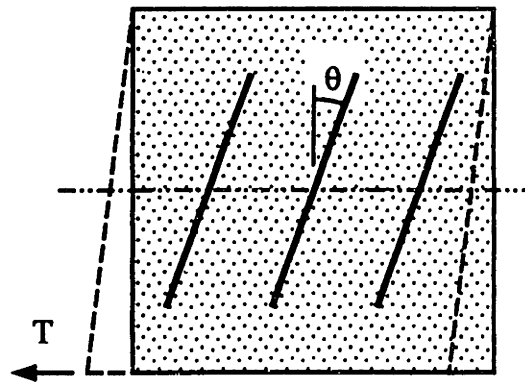


Figure 1.1: Boundary Conditions in Pullout Tests (after Larson, 1992).



(a) Schematic of Reinforced Direct Shear Box Test  
(after, Jewell, 1980).



(b) Shearing in the Direct Simple Shear Apparatus  
(after, Hayashi et al., 1988).

Figure 1.2: General Arrangement of Soil Reinforcement in Shear Box Tests.

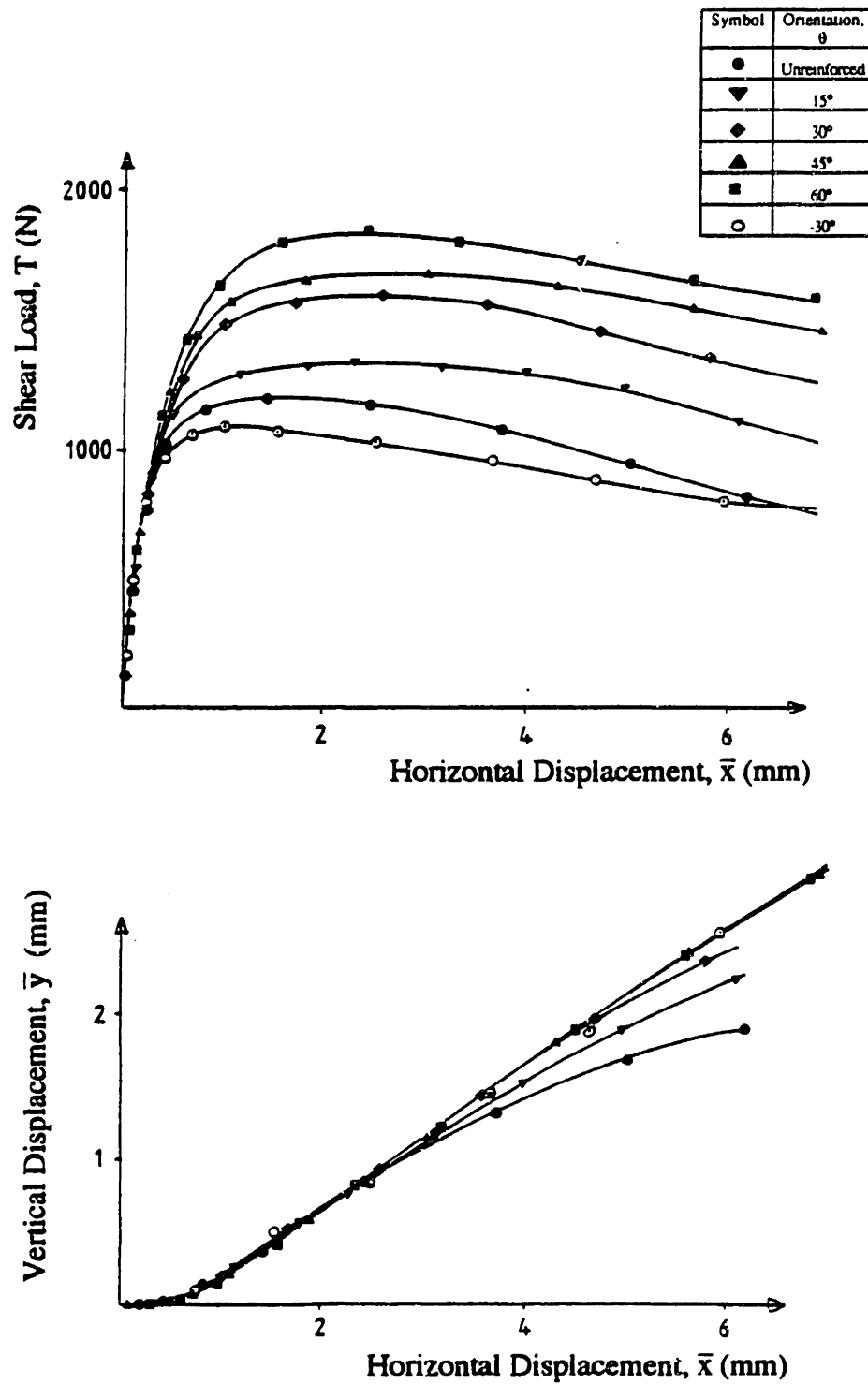


Figure 1.3: Typical Results of Direct Shear Tests on Leighton Buzzard Sand Reinforced by a Single Steel Grid at Different Orientations (Jewell, 1980).

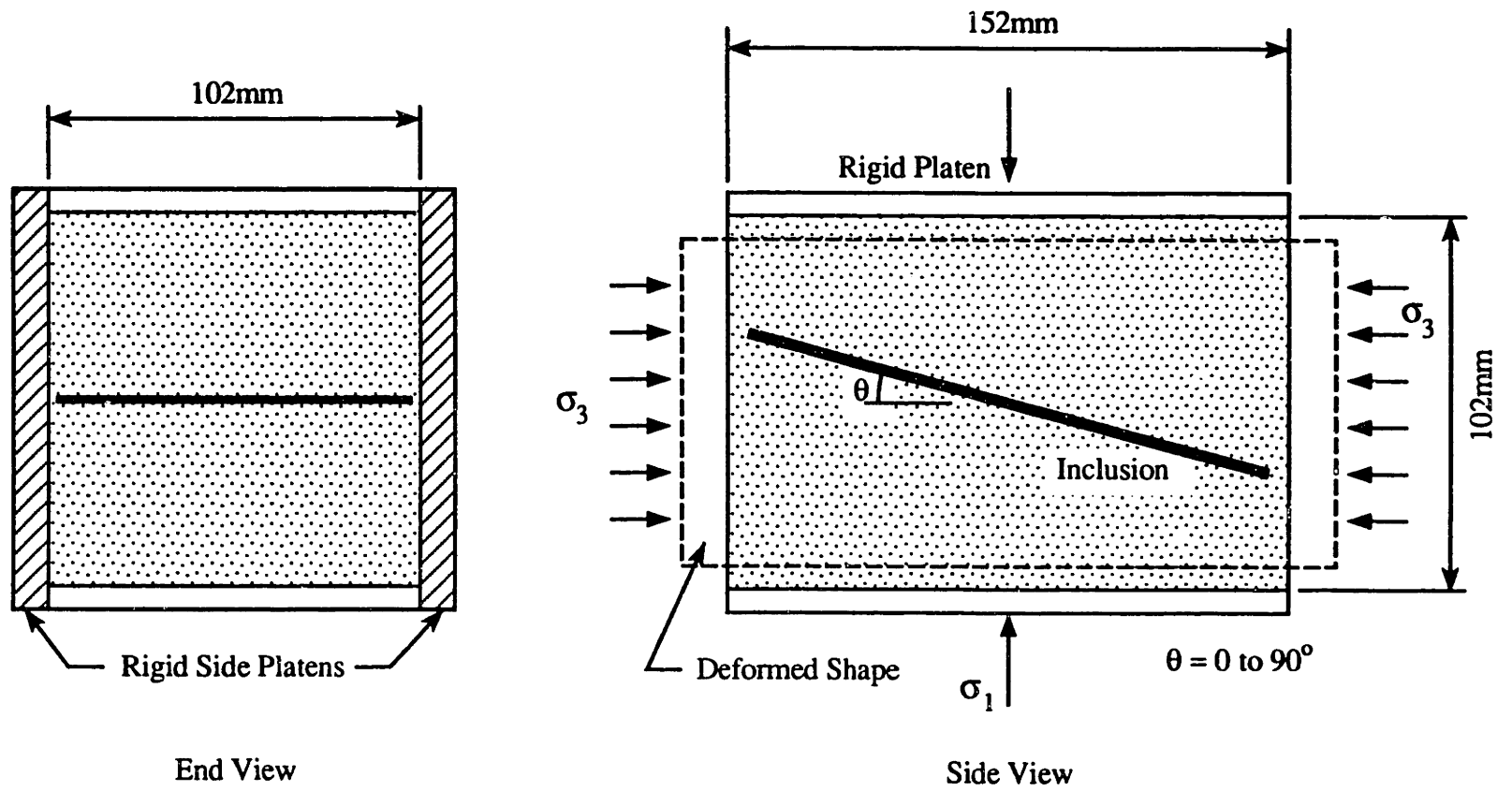
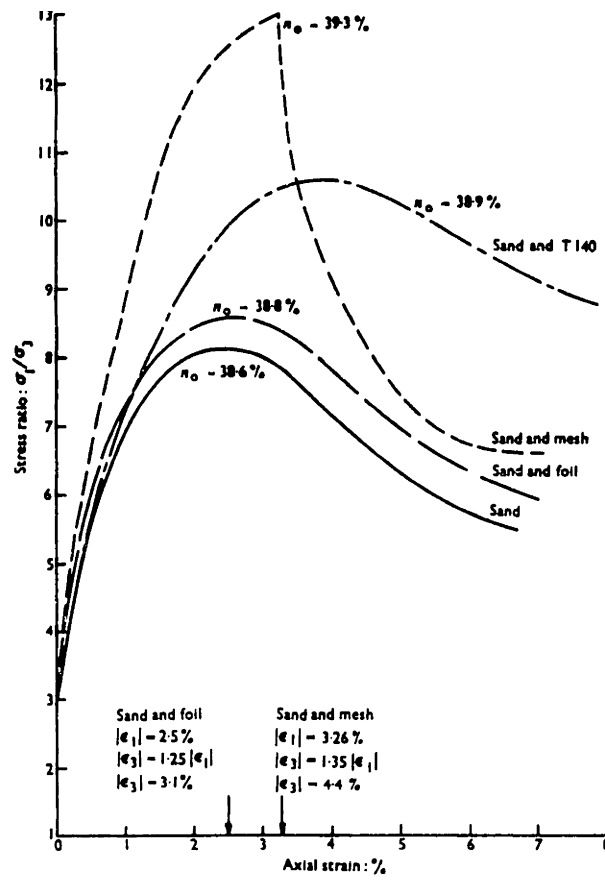
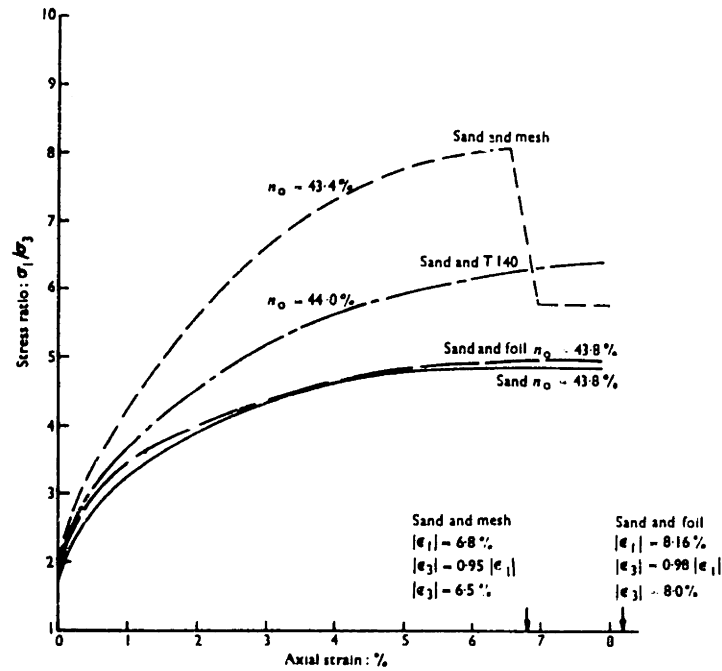


Figure 1.4: Boundary Conditions in the Unit Cell.



(a) Dense Leighton Buzzard Sand.



(b) Loose Leighton Buzzard Sand.

Figure 1.5: Effect of Reinforcements on External Stress-Strain Behavior Measured in Plane Strain Shear Test (McGown et al., 1978).



# Chapter 2: Description of the APSR Cell

## 2.1. Introduction

This chapter describes a composite element test device, referred to as the Automated Plane Strain Reinforcement (APSR) cell, originally designed and built by Douglas Larson as a part of his Ph.D. dissertation (Larson, 1992). The APSR cell is capable of measuring the maximum tensile stress that develops at the center of a single planar inclusion due to shearing of the surrounding soil in plane strain compression. The hardware and software configuration of the pre-existing prototype APSR cell has been substantially modified during the present research in order to improve the performance and reliability of the cell.

Figure 2.1 shows the idealized geometry for a composite plane strain element of soil reinforced by a planar inclusion of length,  $L$ , which is oriented parallel to the direction of the external minor principal stress,  $\sigma_3$ . As the soil is sheared in a plane strain compression mode (by increasing the major principal stress,  $\sigma_1$ ), tensile stresses are transferred to the inclusion. The plane of symmetry,  $x = 0$  (Figure 2.1) has well defined, mixed boundary conditions specified by: 1) no lateral displacement,  $u_x = 0$ , and 2) no shear stress acting along the plane,  $\tau_{xy} = 0$ . The APSR cell takes advantage of this symmetry by simulating one-half of the unit element containing an inclusion of length,  $L/2$ , as shown in Figure 2.2. The rear wall of the cell ( $x = 0$  plane, Figure 2.2) is rigid and lubricated to minimize friction. The key design feature of the APSR cell is that the inclusion is clamped externally to a load cell which measures the force in the reinforcement at a location equivalent to the centerline of an inclusion with length,  $L$ .

In order to maintain the symmetry along the rear wall, a linear actuator controls the position of the reinforcement such that there is no displacement of the inclusion at the reference entry point (X in Figure 2.2).

Figure 2.3 shows a cross section through the APSR cell. It contains a soil specimen measuring 570 mm high by 450 mm wide by 152 mm deep (plane strain direction) enclosed in a thin rubber membrane. The APSR cell can accommodate a single reinforcing inclusion in size up to 450 mm in length (equivalent to a total sample length,  $L = 900$  mm) and 150 mm in width. The inclusion passes through a slot in the rear wall of the cell and is held in position by jacking against an external support arch. A thin airbag (rubber bladder) applies uniform confining stress,  $\sigma_3$  to the front face of the specimen, while the major principal stress,  $\sigma_1$  is imposed by moving two rigid end platforms into the specimen. As the specimen is sheared, it deforms freely in the lateral direction against the airbag. The sidewalls of the APSR cell maintain plane strain condition throughout the test using a unique active control system.

Uniform specimens of dry sand are prepared using a custom-built raining apparatus. Sand specimens deposited in the z (vertical) direction initially exhibit isotropic properties for plane strain shearing in the x-y plane. This allows load-transfer behavior to be examined independent of the effects of soil anisotropy. All rigid contact boundaries are lubricated with two layers of silicone grease separated by thin rubber sheet to minimize surface friction.

The APSR cell has been extensively instrumented with a variety of electronic transducers. Displacement transducers and magnetic proximity sensors measure movements of the specimen boundaries and pressure transducers measure the applied major, minor, and intermediate principal stresses. A load cell measures the maximum tensile stresses in the inclusion, while a custom designed infrared sensor monitors its



position. Additional instrumentation can be designed to measure the local strains and/or stresses at locations along the inclusion for certain types of reinforcing materials. It is possible to measure internal soil deformations through radiography.

The APSR cell is fully automated with eight digital feedback control loops that control displacements of the platform jacks, the positions of the plane strain walls and the inclusion, and the confining air pressure. An IBM compatible personal computer controls various aspects of the test, performs data acquisition and analysis, and displays the results in real-time. Two separate carts house the APSR cell and the test control electronics (Figure 2.4). Connections between the two are quickly and easily removed and reattached so both carts can be transported separately.

The first part of this chapter discusses mechanical design of various APSR sub-systems. The next section describes the instrumentation required to provide interpretable results. The last part of the chapter describes hardware and software aspects of the control system which maintains the necessary boundary conditions in the cell. Several important changes have been made from the prototype device described by Larson (1992). Some of these modifications and the resulting improvements are documented in Appendix A.

## 2.2. APSR Cell Design

The previous section presented the conceptual design and summary of the final configuration of the APSR cell. This section provides a brief discussion of the design requirements and then describes the mechanical design of various APSR sub-systems in some detail. The APSR cell is a plane strain device capable of independent control

of three principal stresses and a reinforcing inclusion. Consequently, the discussion of its design has been conveniently divided into three principal axis systems plus the reinforcement system. The sections on design requirements and size considerations are abbreviated from earlier work, and reader should refer to Larson (1992) for a more detailed discussion on these topics.

### 2.2.1. Design Requirements

The APSR is a composite element test designed to investigate specific aspects of the soil-reinforcement interaction problem. The overall goal of the design was to develop a laboratory device capable of directly measuring load transfer behavior for a wide variety of reinforcing materials within the working stress range. The following key design requirements for the APSR cell were specified (Larson, 1992).

- a. The dimensions of the APSR cell should be sufficiently large to minimize the scale effects commonly encountered in extrapolating the results of laboratory tests to actual reinforced structures.
- b. The boundary conditions imposed by the cell should be well controlled and should represent the boundary conditions found in actual structures to allow rational interpretation of the test data.
- c. The cell should be able to develop stresses and strains of sufficient magnitude required to investigate the load transfer behavior in a wide variety of inclusion materials. This is particularly important for studying extensible reinforcing materials such as non-woven fabrics.
- d. The instrumentation should provide direct measurements of forces or deformations in the reinforcement. Remote or indirect measurements may be used to assess soil deformations within the specimen.

The following sections describe the APSR cell in detail and show how the above requirements were incorporated into the final design.

### 2.2.2. Size Considerations and Design of Structural Box

The APSR cell is designed to accommodate a soil specimen measuring 570 mm long by 450 mm wide by 152 mm deep (Figures 2.2, 2.3). The size of the specimen is a compromise between the need to obtain interpretable results representative of the field situation and the conditions that could be achieved practically in a laboratory scale test. A preliminary analysis using the shear-lag method (Abramanto & Whittle, 1993) indicated that for a reinforcement-matrix stiffness ratio,  $E_f/G_m = 200$  (typical of many geosynthetics) an inclusion with the half-length,  $L/2 = 400$  mm can develop 90% of the maximum tensile stress for an infinitely long reinforcement (refer to Chapter 4 for details on shear-lag analysis). Findings similar to this were used to arrive at the maximum inclusion half-length,  $L/2$  of 450 mm. The second factor affecting inclusion size is the characteristic dimensions for geosynthetic materials such as geogrids. For these materials, McGown et al. (1985) have shown that there is a minimum width of test specimen necessary to ensure properties which are representative of the manufactured product (based on in-isolation tests). Based on a survey of most commonly used grids in 1990, the dimension of the APSR cell in the plane strain direction was selected as 152 mm. This allows a reinforcing inclusion up to 450 mm in length (equivalent to a total sample length,  $L = 900$  mm) and 150 mm in width.

The magnitude of applied boundary stresses determine the critical dimensions of the structural box for the APSR cell. Soil properties are highly dependent on stress level, therefore it is particularly important to select stress ranges which are representative of field conditions. The design of the APSR cell assumes a maximum confining stress,  $\sigma_3 \leq 50$  kPa, which is equivalent to 3 m of overburden pressure and

is comparable to in-situ confining stresses in actual reinforced soil structures (Larson, 1992). At this confining stress, assuming a plane strain friction angle of  $50^\circ$ , the major principal stress  $\sigma_1$ , at failure is about 400 kPa. Using a conservative value of  $b = 0.5$ , where  $b = \frac{\sigma_2 - \sigma_3}{\sigma_1 - \sigma_3}$ , the intermediate principal stress  $\sigma_2$ , necessary to maintain plane strain condition was estimated to be about 200 kPa.

The main body of the APSR cell is a rectangular strong-box of size 815 mm long by 540 mm wide by 200 mm deep (Figure 2.3). It is built from aluminum plates held together by steel bolts. The box is supported only at the corners by four adjustable steel legs (Figure 2.7). It is mounted on top of a rigid frame constructed from aluminum angles. The support frame is fitted with four wheels to allow easy transportation during the set up phase of a test. The end plates of the APSR cell are 13 mm thick, and the front and back plates are 16 mm thick (Figure 2.3). Six 1/4-20 bolts attach each end plate to the front and back plates. The sidewalls not only maintain the plane strain condition, they also provide reaction against the compressive forces applied to the specimen. The design of the plane strain sidewalls is described in Section 2.2.4. The top sidewall is detachable and it is removed while setting up the test. A total of 53 1/4-20 bolts attach sidewalls onto the main body of the cell, and resist a design thrust of 56 kN with a factor of safety (FS) of about 2.

In an ideal soil element test, the state of both stress and strain should be uniform. Since it is not possible to eliminate the boundary friction completely, a device which applies uniform boundary tractions to the soil specimen cannot control the deformations and strains within the specimen. Similarly, a displacement controlled (rigid) boundary condition produces non-uniform stress field within the soil specimen. This, of course, is a common problem encountered in the design of laboratory element tests in soils, and demands a compromise. The APSR cell, like the conventional

triaxial apparatus, is a mixed boundary condition device. A set of rigid platforms impose uniform boundary displacements in the major principal stress direction while a thin flexible membrane maintains uniform boundary stress (traction) in the minor principal stress direction. Ideally all six sides of the specimen should be free of shear stresses (i.e., friction less). In practice, this is approached by lubricating all contact surfaces in the APSR cell with two thin layers of grease, separated by a thin natural rubber sheet. The grease is a mixture of Dow-Corning high vacuum grease and Compound 7 in equal proportion. The Compound 7 is a release agent which reduces the viscosity and shear strength of the vacuum grease. The proof tests designed to evaluate effectiveness of the lubrication indicate an equivalent friction angle,  $\phi_g = 0.45^\circ$  (Larson, 1992).

### 2.2.3. Major Principal Axis

As stated earlier, the major principal stress is applied through two rigid platforms which move in unison to maintain the symmetry of the test. Each platform is attached to a pair of hydraulic jacks as shown in Figure 2.5. The platforms are made from 15.7 mm thick aluminum plates with an all-around clearance of 0.8 mm to prevent the plates from binding against the outer walls of the cell. The four hydraulic jacks are made from brass and are custom designed for the APSR cell (Larson, 1992). They have a maximum stroke of 35 mm which corresponds to about 12 % axial strain ( $\epsilon_{yy}$ ). Each jack consists of a piston-cylinder assembly with 140 mm outer diameter and 127.8 mm inner diameter. An O-ring fits inside a groove near the open end of the cylinder to provide a water-tight seal. The bottom of each jack has a threaded hole to accommodate a 1/4" Swagelok® tube fitting for attaching the water supply line. The jacks are extended and retracted by pumping water into or out of them using custom designed pressure-volume controllers (refer to Section 2.4.2.1). A 6.35 mm diameter

plastic tubing connects a jack to the pressure-volume controller. The ratio of the inner diameter of a jack to that of a pressure-volume controller is about 2.37, which provides a hydraulic gain of 5.6. The water pressure inside each jack is measured by a pressure transducer attached to the supply line. The major principal stress,  $\sigma_1$ , is computed from these measurements. Displacement transducers (LVDTs) measure the position of each end of a platform (Figure 2.5). The average of all four LVDT readings is used to compute the axial strain in the specimen. The position of each jack is controlled by the test computer using a digital feedback loop (refer Section 2.4 for details). By coordinating the movements of adjacent jacks, the computer can prevent the rotation of the platforms and drive them at constant displacement rate (velocity).

In the earlier design of the APSR cell, flexible water-filled pressure bags were placed between platform plates and the soil specimen (Larson, 1992). This was done in an effort to maintain a uniform pressure distribution along the major principal stress boundary. However, this design aggravated the already difficult problem of non-uniform soil deformations due to friction along the top and bottom sidewalls. It was decided to replace the water bags with rigid plywood platforms (Figure 2.5).

#### 2.2.4. Intermediate Principal Axis

The APSR cell maintains the plane strain (no lateral displacement) boundary condition in the intermediate principal stress direction. There are many plane strain laboratory devices described in the geotechnical literature (Seah, 1990; Tatsuoka et al., 1986; McGown and Andrawes, 1977; Green and Reades, 1975). All of these devices rely on thick side platens with high bending stiffness to approximate the plane strain condition. The large surface area of the sidewalls ( $0.26 \text{ m}^2$ ) and the anticipated magnitude of the lateral stress ( $\sigma_{zz} = 215 \text{ kPa}$ ) make this approach impractical for the APSR cell. The allowable out-of-plane strain was estimated to be 0.05% which

corresponds to 38  $\mu\text{m}$  maximum permissible deflection of the sidewalls. An approximate analysis using plate theory shows that 36 mm thick steel plates are required to keep the deflections within this limit (Larson, 1992). In comparison, the active plane strain control system was able to keep the out-of-plane strain well within 0.001 % (refer to Section 2.4.4.2). In addition, the active control system offers the following advantages: a) reduced size and weight of the side walls, b) direct measurement of the intermediate principal stress,  $\sigma_{zz}$ , and c) ability to measure soil deformations using radiography. Thin sidewalls allow more X-Rays to penetrate the APSR cell and hence, shorten the exposure time. Radiography has been used in APSR cell primarily to establish the uniformity of the strains inside the unreinforced soil (Larson, 1992). The equipment available at MIT Geotechnical laboratory is able to measure strains in the plane of X-Ray film to a resolution of about 0.2 % on a 10 mm grid.

The active plane strain control system uses a sidewall which is comprised of two aluminum plates, separated by a thin layer of water (Figure 2.6). A 2.1 mm diameter feeler rod passes through a hole located at the center of the outer wall and rests against the inner wall. An O-ring around the rod forms a seal to prevent the pressurized water between the inner and outer walls from leaking out. A compression spring keeps the feeler rod in continuous contact with the inner wall. All movements of a steel target (small disc) attached to the top of the feeler rod, therefore, correspond to displacements of the inner wall. The position of the steel target is monitored by a magnetic proximity sensor. The top and bottom proximity sensors are held in position by two custom designed clamps which provide stable support (Figure 2.7). These clamps, in turn, are attached to the frame which supports the entire APSR cell. This frame provides a fixed reference point which is unaffected by the deformations of the APSR cell. As the inner wall starts to move, a feedback control system (refer to Section

2.4 for details) adjusts the water pressure in the cavity between the inner and outer walls, thereby, returning the inner wall to its initial position. Although the active control system monitors and controls wall deflections at only one central point, proof tests conducted by Larson, (1992) indicate that the entire inner wall remains flat throughout the test.

Figure 2.8 taken from Larson, 1992 shows the construction of the plane strain walls. The outer wall actually consists of three pieces: a heavy outer plate, a shim, and a sheet metal liner to contain the water layer. The water cavity is sealed by an O-ring in the outer plate. The shim is made from 1.6 mm thick aluminum sheet with a cut-out to fit around the O-ring. The sheet metal liner sits on top of the shim. A series of flat-head screws pass through the liner and the shim and thread into the outer plate to squeeze the liner against the O-ring. This forms a tight seal to hold the water inside the wall. The outer plate has a threaded hole to accommodate a 1/4" Swagelok® tube fitting which connects to the water supply line. The water pressure within the sidewalls is adjusted using custom built pressure-volume controllers described in Section 2.4.2.1. The average pressure in the sidewalls corresponds to the intermediate principal stress for plane strain shearing and is monitored by pressure transducers attached to the water supply lines.

### 2.2.5. Minor Principal Axis

The APSR cell maintains a constant stress in the direction of the minor principal axis during shearing of the soil. A thin rubber bladder (airbag) applies the minor principal stress (confining pressure) to the specimen (Figure 2.3). The details of the airbag construction are shown in Figure 2.9. A thin (0.76 mm) natural rubber membrane is wrapped around the 1/4" thick aluminum plate. A groove in the inner plate holds the O-ring which is used to form an air-tight seal. A series of 3/16" size



screws pass through the outer plate and thread into the inner plate to squeeze the rubber membrane against the O-ring. Compressed air enters the airbag through a nozzle attached to the inner back plate. When inflated, the airbag completely fills the void between the front of the cell and the specimen without any loss in pressure. The airbag allows a maximum of about 30 mm outward movement of the specimen which corresponds to the lateral strain,  $\epsilon_{xx} = 6.7\%$ . The air pressure is controlled by an electro-pneumatic regulator described in detail in Section 2.4.2.3. The lateral deformations of the specimen are monitored using three displacement transducers (LVDTs). The LVDT cores enter the air bag through air-tight holes in the back plates and rest against tiny metal discs attached to inside of the rubber membrane. A pressure transducer mounted near the front wall measures the air pressure. The ends of the air bag overlap the loading platforms near corners as shown in Figure 2.5 resulting in some interaction between minor and major principal stresses. However, the magnitude of this interaction is small and it is easily accounted for by the calibration process described in Section 2.3.2.

## 2.2.6. Reinforcement System

The reinforcing inclusion in the APSR cell is clamped externally to a load cell. Since the plane through which the inclusion enters the cell is the plane of symmetry (Figure 2.3), it is assumed that the load cell measures the tensile stress at the center of an inclusion with an effective length twice the actual inclusion length. For this to be true, however, the position of the reinforcement should be controlled such that there is no displacement at the entry point marked X in Figure 2.2. This is accomplished by continuously monitoring the position of the entry point using a special sensor and adjusting the tensile force in the reinforcement to compensate for any tendency of this

point to move. This section describes the reinforcement grips, monitoring, and control mechanisms which constitute the most important components of the APSR cell.

The cell has been designed to accommodate a wide variety of reinforcing materials with different thicknesses, stiffness, and geometry. The reinforcement enters the APSR cell through a slot in the rear wall as shown in Figure 2.10. Two wedge shaped pieces form a tight fitting slot around the inclusion to prevent soil particles from squeezing out of the cell. By adjusting these wedges, it is possible to accommodate reinforcement widths up to 130 mm.

Since the APSR cell employs active feedback control to maintain reinforcement position, some slippage within the reinforcement grips is acceptable. This simplifies the grip design considerably. The grips consist of two 12.7 mm thick aluminum plates measuring 152 mm by 79 mm (Figure 2.11). They fasten to the free end of the inclusion with three 3/8" bolts. The spacing of these bolts is changed to accommodate grids of different geometry. Two 3.2 mm thick rubber pads placed between the grips and the inclusion increase the interface friction.

The inclusion position is varied by a light-weight linear ball screw actuator manufactured by Duff Norton Inc. (Figures 2.10, 2.11). The ball screw actuator converts rotary motion of a small DC servo motor into linear motion. Section 2.4.2 describes the linear actuator, DC motor, and associated control hardware in detail. A C-clamp connects the reinforcement grips to the actuator arm through a load cell. The actuator assembly is mounted on a rigid semicircular arch which provides the necessary reaction. The arch fits against the rear wall of the cell where it is held in place by four angles (Figure 2.10). Each angle bolt into the arch with three 3/8" diameter bolts.

The measurement of the inclusion position at the exit point presented a special challenge due to the following reasons.

- a. For a relatively stiff reinforcing material such as steel, the accuracy with which the exit point position has to be maintain is very high. A position error greater than a few micrometer (microns) can alter the reinforcement load significantly (refer to Appendix A for more information).
- b. Ideally, the exit point should coincide with the inside of the APSR cell back wall. However, it is impossible for any measurement device to access this point directly due to space restrictions.
- c. The measurement system should be sensitive only to longitudinal movement of the reinforcement. In other words, it should not respond to any lateral (out of plane) deflections.
- d. The measurement of position requires a stable reference point with respect to which any possible movement of the target point may be discerned. Since the structural box of the APSR cell is subjected to large stresses (and hence strains) during the test, the reference point should be located sufficiently away from the main body of the cell.

The reinforcement referencing and positioning system has gone through several revisions as described in Appendix A. The final design uses a matched infrared (IR) source-detector pair comprising a Ga-As IR emitting diode source and a photo-transistor detector. Figure 2.12 illustrates the application of IR sensor to monitor the reinforcement displacement at the exit point of the APSR cell. The rectangular infrared light beam (1.52 mm x 1.27 mm) passes through a tiny window (hole) in the reinforcement which is located at some convenient distance from the APSR exit point. The IR beam bisects the sharp edge of an 'indicator strip' which is only bonded to the reinforcement at the exit point. All movements of the indicator edge across the

window, therefore, correspond to displacements occurring at the exit point. The amount of light reaching the IR detector registers as a change in voltage. Figure 2.13 shows the calibration of the IR detector as a function of the position of a sharp edge. The data show that a displacement of 0.2 mm corresponds to a 3.5 V change in the output voltage. Taking a conservative value of 0.15 mV (about 16 bit) as resolution of the voltage measuring device, the system has a sensitivity of 0.0085  $\mu\text{m}$ !

Figure 2.14 shows the mechanical design of the IR reinforcement monitoring system. The IR source and detector are both enclosed in separate opaque plastic housings with IR transmissive inserts which form the optical windows. This provides excellent protection against ambient light and airborne contamination. The source-detector pair is mounted, about 5mm apart, to the top of a light-weight aluminum fork. The fork is attached to a miniature sliding platform. A precision vernier screw allows the adjustment of the platform position. The complete assembly is attached to the frame which supports the APSR cell. This frame provides a fixed reference point which is unaffected by deformations of the APSR cell. The assembly provides four degrees of freedom (three translational, one rotational) required for the initial alignment of the sensor-detector with respect to the indicator edge. The initial alignment is carried out to adjust the sensor output to its mid-span value (about 2.5 V) to ensure maximum useful range and sensitivity. Although the details of mounting indicator strip vary depending on the type of material (e.g. steel sheet vs. polymeric grid), the basic mechanism of infrared sensing works well for all materials used so far in the APSR cell (see Section 2.4.4.4).

## 2.3. Instrumentation

The APSR cell is extensively instrumented with a variety of transducers for measuring boundary stresses and displacements. It is also possible to add instrumentation to monitor behavior of points along the inclusion. A total of 18 electronic transducers measure various parameters during the course of a standard test. Additional instrumentation is used whenever diagnostic tests are performed for solving specific problems. Figure 2.15 shows the locations of the transducers and Table 2.1 lists their calibration factors, ranges, and resolution. The data collected from these transducers are used for both test control and interpretation. The transducers receive their power from two Hewlett Packard 6205C dual DC power supplies. The output of all transducers is converted to digital form by two analog-to-digital converter (ADC) cards and continuously recorded by the test computer. In addition to external measurements, X-Ray techniques can be used to measure strains occurring within the soil and inclusion. However, the present study did not use this capability and the reader is referred to Larson (1992) for details.

### 2.3.1. Boundary Displacements

The direct-current-in-direct-current-out (DC-DC), linear voltage displacement transducers (LVDTs) are used to measure the axial displacement imposed on the specimen by the platforms and the lateral deformation of the specimen against the airbag. The DC-DC LVDT (also referred as DCDT) is manufactured by Trans-Tek, Inc. and consists of two parts: a 3/4" diameter cylindrical body (barrel), and a metal core which slides freely through the barrel. The barrel contains a differential transformer, an oscillator, and a demodulator. The oscillator converts the DC input voltage to a high frequency AC signal. This AC current excites the primary winding of the differential transformer which induces a voltage in the secondary windings. The

amount of voltage generated in the secondary depends on the position of the core with respect to the barrel assembly. The demodulator converts the output back to a DC signal which is linearly proportional to core displacement within a specified range. A LVDT is an infinite resolution device meaning its resolution is limited only by the resolving capability of the voltage measuring system (i.e., the ADC). All of the LVDTs used in the APSR cell have linear displacement range  $\pm 13$  or  $\pm 25$  mm. The LVDTs accept any input voltage between 6 and 30 V. However, because most of the other transducers used in this research require 5 to 6 V input, a standard 6V was used for all transducers. At this input voltage, the LVDTs have an output range of  $\pm 3.0$  V for the  $\pm 13$ mm capacity and  $\pm 4.3$  V for the  $\pm 25$ mm capacity transducers. For 0.1 mV (about 17 bits) effective resolution, this translates into measurement sensitivity of about 0.43 and 0.58  $\mu\text{m}$  respectively.

Four LVDTs (D1, D2, D3, D4; Figure 2.15) measure the displacements of the two loading platforms. The barrel of each transducer is held in a fitting that threads into the outer wall of the APSR cell. The LVDT core threads into 12.7 mm diameter posts, which in turn thread into the platforms to form a rigid connection (Figure 2.5). The control algorithm uses the readings from each pair of adjacent transducers to match displacements of the two jacks on each platform and hence prevent platform rotation (refer to Section 2.4.4.1). Since the loading platforms are rigid, the average strain in the major principal direction ( $\epsilon_{yy}$ ) is readily computed from the platform LVDT readings<sup>1</sup>. The platform LVDTs have 0.58  $\mu\text{m}$  measurement sensitivity and, therefore, are capable of measuring axial strain ( $\epsilon_{yy}$ ) to a resolution of about 0.0002 %.

---

<sup>1</sup> Non-uniformity of the strain field within the soil specimen can be interpreted from radiographic measurements using procedures developed by Larson (1992).

Two electromagnetic proximity sensors are used to monitor the movement of the inner plane strain walls (PS1, PS2; Figure 2.15). The sensors are manufactured by Electro Corporation, and marketed under the name Electro-Mike Displacement Transducer (EMDT). The transducer comprise two parts: a converter module and a sensor tip. The converter module sends power to the sensor, which project a 200 kHz electric field in front of its tip. This field generates eddy currents in any metal target which the field intercepts. The eddy current removes some of the energy from the field. The amount of energy removed from the electric field is inversely proportional to the distance between the tip and the metal target. Although 2.5 mm linear range of a proximity sensor is much smaller than that of a typical LVDT, it is an excellent sensor for applications where maintaining a stationary position is the prime objective. The proximity sensor offers the following advantages over a LVDT: a) about five times higher sensitivity, b) ability to make remote (non-contact) measurement, and c) a less noisy signal. The EMDT accepts 12 V input voltage and produces a 10 V output voltage swing for about 2.5 mm displacement. Again, taking 0.1 mV measurement resolution this corresponds to 0.025  $\mu\text{m}$  sensitivity. The specimen is 152 mm wide in the plane strain direction, so this provides capability of measuring the out-of-plane strain (as represented by displacement of the target point) to a resolution of 0.00003 %.

The lateral displacements of the specimen are measured by three LVDTs (D5, D6, D7; Figure 2.15) which pass through the airbag to enter the cell. These LVDTs are also held in position by fittings that thread into the outside of the cell. Their cores rest against tiny metal discs glued to the inside of the airbag rubber membrane. Thin rubber bands are used to provide the axial thrust necessary to maintain a firm contact between the cores and the specimen. The LVDTs have about 0.43  $\mu\text{m}$  measurement sensitivity, so the setup is capable of estimating average lateral strains to a resolution of about 0.0001 %.

## 2.3.2. Boundary Stresses

The three principal boundary stresses applied to the APSR soil specimen are measured by seven pressure transducers (P1 through P7; Figure 2.15). The pressure transducers are manufactured by Data Instrument Corporation. They come in different capacities but share the same basic design. The transducer consists of a sealed chamber with a stainless steel diaphragm that deforms under external pressure. Two semiconductor strain gages attached to the diaphragm register the resulting tensile or compressive strains. A bridge circuit produces a small output voltage which is proportional to the applied pressure.

Four pressure transducers (P1, P2, P3, P4; Figure 2.15; Table 2.1) attached to the pressure-volume controllers (see Section 2.4.2.1) measure the water pressure in the platform pistons. Since the rate of water flow is negligible, measuring the pressure at the source reflects accurately the pressure in the pistons. The major principal stress is computed from piston pressures by applying three corrections discussed below.

- a. The area correction is necessary because the combined area of two pistons is about  $25664 \text{ mm}^2$  while the area of the platform is  $68400 \text{ mm}^2$ . This provides an area ratio of approximately 0.375.
- b. The O-ring inside the piston-cylinder assembly produces a small amount of friction which dissipates a part of the pressure. The loss of pressure equals the piston pressure required to drive the platforms when there is no soil specimen in the cell.
- c. As mentioned in Section 2.2.3, the airbag ends overlap part of the loading platforms near corners. Thus, the platforms have to overcome a small lateral thrust produced by the airbag. The magnitude of this thrust is



estimated by inflating the airbag and registering the resulting increase in the piston pressures.

The above corrections are incorporated into the following formula which is used to convert piston pressures into equivalent platform pressure, P:

$$P = 0.19(PP_1 + PP_2) - 5.5 \quad (3.1)$$

where  $PP_1$  and  $PP_2$  are piston pressures in kPa. The major principal stress  $\sigma_{yy}$ , taken as the average of the two platform pressures, is measured to the nearest 0.25 kPa.

The intermediate principal stress,  $\sigma_{zz}$ , is measured by two pressure transducers (P5, P6; Figure 2.15; Table 2.1) attached to the pressure-volume controllers that feed the plane strain sidewalls. A low capacity pressure transducer measures the minor principal stress,  $\sigma_{xx}$ , to a resolution of 0.11 kPa.

### 2.3.3. Inclusion Load and Position

A load cell connected to the linear actuator arm measures the force carried by the inclusion. The APSR testing program uses two interchangeable load cells, JP-500 and JP-2000, with 2.2 kN and 8.8 kN capacity respectively to cover the range of inclusion material types and lengths. The load cells, manufactured by Data Instrument Inc., are of shear beam design which minimizes the error due to eccentric loading. The load cells, like the pressure transducers described earlier, rely on a strain gage circuit to produce output which is linearly proportional to applied load. They have a maximum deflection of 0.05 mm and an output of approximately 150 mV at maximum load. Taking 0.1 mV as the resolution of A/D converter, the load cells measure reinforcement load to nearest 1.47 N and 5.87 N respectively. In addition to the external load measurement, strain gages were used in some of the tests to measure the internal

reinforcement strains. Larson (1992) describes the attempts to measure distribution of load along the length of steel sheet inclusion using foil type strain gages. However, further testing showed that internal strain gages did not perform adequately in the APSR experiments (see Chapter 4) and their use was discontinued in the later stages of this APSR test program.

The position of the inclusion exit point with respect to a reference frame is measured by an infrared (IR) sensor (Figure 2.14). The sensor assembly comprises an IR diode source and a matched photo-transistor detector shown in Figure 2.16. The photo-transistor is powered by 6.0 V DC input and swings the output ( $V_{out}$ ) from about 5.8 V (completely dark) to 0.1 V (fully saturated). The IR diode requires 1.5 V DC input and is powered from a separate power supply (Hewlett Packard model 6234A). The series resistors in Figure 2.16 limit the current through the diode and transistor to acceptable levels. Although the response of the IR sensor as a function of reinforcement position is non-linear (Figure 2.13), it is highly repeatable. The feedback loop that controls the inclusion position keeps the sensor reading very close to its initial value. Within this small range, the calibration curve can be approximated as straight line with a constant calibration factor (slope). The infrared sensor provides an extremely stable (noise free) output with sub-micron sensitivity.

#### 2.3.4. Data Acquisition and Reduction

The APSR test computer (see Section 2.4.2.6) was used to control the test as well as acquire, process, and display test data. This integrated system offers excellent flexibility and ease of use, but puts greater burden on the computer program that coordinates these activities. Section 2.4.3 and Appendix D give details of a menu based windowing software called FlexCAT which provides real-time data acquisition capability through ADC boards.

All of the transducers in the APSR cell receive their power from two identical junction boxes, each with 16 channel capacity. The junction boxes are made from aluminum sheet metal and fitted with a set of 5 pin male Amphenol® connectors (Figure 2.17). A shielded cable with four insulated conductors (Balden 8723) connects each transducer to the junction box. The power is delivered to a transducer through the lower two pins (pins D and E) of the connector; pin D connects to the low end and pin E to the high end of the input voltage. The transducer signal returns to the junction boxes through the upper two pins (pins A and B). The cable shield is connected to the central pin (pin H) which, in turn, is connected to the power supply ground. The power pins of all Amphenol® connectors are connected to the connector labeled "Power Input". Two computer cables connected to the signal pins of the Amphenols feed the output of transducers to the AIC-16 analog-to-digital (ADC) boards through a 36 pin D-sub connector. The transducer output is also wired to one of the two rotary switches which connects the selected channel to a voltmeter through the connector marked "Line to Voltmeter" in Figure 2.17. Two digital voltmeters (Keithley Model 177) can be used to take manual readings and to verify proper functioning of the ADC boards.

The computer reads 18 transducer channels at 90 second intervals and stores raw voltage readings on the hard disk in an ASCII format file. This file can be imported into LOTUS® or any other common spreadsheet program for further processing. The user specifies the input file name, number of channels, reading frequency, and maximum number of readings through a dialog box. It is possible to suspend the data acquisition activity and resume it any time during a test. At the end of a test, the computer program can also produce another file which contains processed data. This file can be directly imported into any graphing software for plotting various test parameters.

## 2.4. Computer Control System

The APSR cell design makes extensive use of digital feedback technique for test control and automation. There are eight feedback control loops that maintain and update boundary conditions during a test. Feedback control is used to regulate: a) the confining air pressure in the cell, b) the movements of four platform jacks, and c) the positioning of the plane strain sidewalls and reinforcement. The active control of plane strain side walls and reinforcement position makes it possible to achieve extremely high virtual stiffness in these systems without increasing sizes of mechanical components. A responsive and stable feedback control system is, therefore, the key requirement for reliable measurements in the APSR cell.

### 2.4.1. Background

Modern control theory and practice is a vast field spanning across a number of disciplines including linear systems theory, electrical and electronic engineering, computer science, and mechanical engineering. This section contains a few basic concepts and terminology necessary for the discussion of the APSR control system. A more detailed discussion is provided in Appendix B. Control is the process of causing a system variable to conform to some desired *reference value*. The system to be controlled can be a physical object (such as a piece of equipment) commonly referred to as *plant*, or it can be a *process* (such as chemical, biological, economic, etc.). Most control systems comprise three basic elements as shown in Figure 2.18a.

1. The central component is the plant or process, one of whose variable is to be controlled.

2. The *actuator* ( or servo ) is a device which can influence the controlled variable in a predictable manner.
3. The *controller* is the device (either analog, or digital, or both) responsible for generating control action based on the available information about the system and its environment.

As shown in Figure 2.18, control systems can be broadly divided into two types. A *feed-forward* (open-loop) control system generates its control signal solely based on pre-existing knowledge of plant dynamics and the value of the reference input. Although the feed-forward control is a simple one-step process, it is sensitive to external disturbances and internal variations in system parameters because the controller cannot adapt to a new situation. A *feedback* (closed-loop) control system is one in which the variable being controlled is measured by a sensor and the information is fed back to the controller. A *comparator* (or summing junction) compares the actual value of output variable with the reference (desired) value to generate an *error signal*. The error signal is further processed by the controller to generate appropriate *control action*. Since any deviation of output variable from its desired value is used to produce control action, feedback systems are essentially self-correcting, hence more robust. An important aspect of feedback control design is the relationship between the error signal and control action. The *control algorithm* is set of mathematical operations performed on the error signal to arrive at appropriate control action. Three common algorithms (see Appendix B) are: a) proportional feedback, b) derivative (rate) feedback, c) integral feedback. In practice, it is common to use more than one kind of feedback, for example proportional-derivative (PD) control or proportional-integral (PI) control. The required mathematical operations on error signal can be performed by either analog or digital means. A digital control system uses a computer (or microprocessor) to implement comparator and controller functions as shown in Figure 2.19. Apart from

the microprocessor, Figure 2.19 contains two additional blocks namely analog-to-digital converter (ADC) and digital-to-analog converter (DAC). These converters form an interface between the outside world which is analog (continuous) and the computer which is a digital (discrete) machine. The clock in Figure 2.19 synchronizes the input/output activity and keeps track of elapsed time. In general, a digital system offers higher noise immunity, flexibility, and reliability compared to an equivalent analog system. The original APSR setup had three analog proportional control loops for controlling plane strain sidewall and reinforcement positions (Larson, 1992) which were converted to digital form as part of the present work.

Figure 2.19 illustrates one of the four feedback loops that control the position of the APSR platform pistons. In this context, the platform piston whose position is to be controlled constitutes the plant. The actuator is a combination of DC servo motor and pressure-volume controller (see Section 2.4.2.1 for details). Feedback is obtained from the LVDT that measures the platform position. The user enters a desired piston displacement rate into the computer. At regular intervals (3 seconds in this case), the computer reads the LVDT and compares actual platform position with the desired position computed from the target displacement rate. It goes through a set of PID calculations (details given in Section 2.4.4) and adjusts the output of DAC to minimize the error. A central clock synchronizes the input/output activities and keeps track of elapsed time. The feedback loop which maintains the confining pressure works in a similar fashion but uses an electro-pneumatic regulator as a servo and a pressure transducer as feedback sensor. The next section describes these and other control hardware components, while Section 2.4.3 describes the real-time software that coordinates the control activity.

## 2.4.2. APSR control Hardware

APSR control system is built around electronic, hydraulic, and pneumatic components which have been used extensively in the MIT geotechnical laboratory for other research projects. Most of them are hardware building blocks of Flexible Automated Technologies for Computer Assisted Testing (FATCAT) system (Sheahan and Germaine, 1992). A brief description of the important components is presented in this section. Appendix C provides a more detailed description of the newly developed analog-to-digital and digital-to-analog converter cards.

### 2.4.2.1. Pressure-Volume Controller

Many geotechnical tests involve generation and/or control of pressure and displacement (or volume). A hydraulic piston-cylinder assembly is the most reliable mechanism for developing pressures and axial loads. The water pressures required to drive the platform pistons and to control the sidewalls in the APSR cell are applied by custom-designed pressure-volume controllers designed originally by Germaine and Andersen (Andersen, 1991). Figure 2.20 shows the schematic view of the pressure-volume controller. Each controller consists of a water chamber (cylinder) and a piston made of brass. The piston is driven by a ball screw actuator into the cylinder, causing an increase in the water pressure. The ball screw actuator converts the rotary motion of a small DC servo motor into linear thrust. Given an input torque of 2.37 N-m, these ball screw actuators can supply up to 4 kN of thrust with a stroke of about 150 mm (Larson, 1992). Table 2.2 provides full technical specifications for the ball screw actuator. A similar actuator is used to control the reinforcement position. The water chamber consists of a hollow cylinder with two end plates. The end plates have a circular groove that fits over the cylinder with an O-ring seated at the bottom of the groove. A set of six stainless steel tension rods connect the top and bottom end plates

and resist the axial force produced by the chamber pressure. The piston passes through a hole in the top end plate with a clearance of about 0.03 mm. This hole has a double O-ring seal to prevent the water in the chamber from leaking out around the piston. The bottom end plate has a small threaded hole to accommodate a Swagelok® tube fitting. The linear actuator is mounted on an aluminum plate with two 3/8" bolts. The reaction force for the actuator comes from four stainless steel tension rod which tie this plate to the bottom end plate. The entire assembly is mounted atop a 127 mm wide aluminum channel.

#### 2.4.2.2. DC Motor and Servo Amplifier

Fractional horse power DC servo motors are used to drive the ball screw actuators. The advantages of DC servo motors over stepper motors include: a) smoother operation with minimum noise and vibrations, b) higher power efficiency, and c) simpler control interface which can be used with both analog and digital control system. The APSR cell uses Electrocraft Motomatic series E-352 motors manufactured by Robbins Myers Corporation. Table 2.3 provides the technical specifications. Each motor is fitted with a 100:1 reduction gear box to provide hundred fold increase in maximum output torque. The motor consists of two units mounted on the same shaft; a permanent magnet brush type DC motor, and a DC tachometer (generator). The tachometer generates voltage output signal proportional to the speed of the motor. The computer control of the DC motor is accomplished by sending a command voltage to a solid state electronic controller (servo amplifier). The servo amplifier maintains a constant motor speed proportional to the command voltage through the high speed analog feedback loop shown in Figure 2.21. The signal from the tachometer is fed to a comparator where it is compared with the speed command voltage. The servo amplifier provides more (or less) voltage to the motor winding in order to increase (or decrease)



speed and hence maintain a balance between the speed command and tachometer voltage. Set speed is thus maintained, regardless of changes in load torque or line voltage. It is important to realize that this feedback loop operates independent of the digital feedback loop (Section 2.4.1) which is built on top of this inner analog loop. The velocity command voltage can vary between +15 to -15 volts which corresponds to full speed clockwise and counterclockwise respectively. A custom built control panel allows the motors to be operated in either manual mode or computer controlled mode. In manual mode, a motor can be advanced or retracted at high speed by sending  $\pm 10$  V signal to the servo amplifier. This mode is useful for de-airing the pressure-volume controllers and making initial adjustments during the setup phase of an APSR test.

#### 2.4.2.3. Electro-Pneumatic Regulator

An electro-pneumatic regulator (also called E/P Transducer) is a device which scales the amount of air pressure on its output side according to the command voltage signal it receives from a control source. The APSR cell uses electro-pneumatic regulator model T5221-9, manufactured by Fairchild Corporation to control confining air pressure. Table 2.4 provides the technical specifications. The T5221 regulator consists of two distinct units; a low pressure electro-magnetic controller and an adjustable ratio pneumatic relay. The electric signal (command voltage) applied to the regulator energizes a small solenoid within the electro-magnetic controller which controls the position of a plunger. The clearance between the plunger and a nozzle determines the amount of air flow through the nozzle and hence the output pressure. Since the electro-magnetic controller cannot handle either large pressure or large flow, its output is fed to a pneumatic ratio relay. The ratio relay provides high volume of flow with an output pressure which is the product of the signal pressure multiplied by

the relay ratio. The ratio of output pressure to signal pressure is continuously adjustable through a knob.

#### 2.4.2.4. AIC-16 Analog-to-Digital Card

The analog-to-digital converter (ADC), as noted in Section 2.4.1, serves as a translator between the analog (continuous) signals produced by the measurement devices (transducers) and the computer which can discern only two voltage levels (corresponding to 1 and 0). The APSR cell uses a custom designed ADC card built around Analog Devices' AD1170 chip (refer to Appendix C for details). The AIC-16 card fits into one of the several expansion slots (bus) provided on the back of the motherboard of an IBM compatible computer. The card provides a total of 16 differential (double ended) analog input channels. The transducer signal lines are connected to the card using a standard 36 pin D-sub connector. AD1170 is a high resolution, programmable, integrating type, A/D converter chip. It can convert to a maximum resolution of 22 bits ( $1 \text{ in } 2^{22}$ ) over  $\pm 5$  volt range. An on-chip microprocessor can perform automatic offset and gain corrections. The values of bit resolution, conversion (integration) time, correction factors etc. can be changed by sending software commands from the computer and stored in an on-chip non-volatile (permanent) memory. The APSR setup generally uses 20 bit resolution and 166.7 milliseconds integration time which provides a maximum reading rate of about 5.8 reading per second. Theoretically, 20 bit resolution translates into an ability to measure voltage changes as small as 0.01 mV over a range of 10 V. However, the system level (overall) resolution in a measurement device is always less than the component (chip) level resolution due to the electrical noise from ground and other sources. The APSR setup has effective resolution of the order of 0.1mV (about 17 bit).

#### 2.4.2.5. AOC-8 Digital-to-Analog Card

Most actuator devices such as DC motors, pneumatic valves, etc. produce an output which is proportional to an analog (continuous) control signal. Computer control of such devices requires a digital-to-analog converter (DAC) to translate digital computer commands into equivalent analog voltage signals. The APSR cell uses a custom designed D/A converter card built around Analog Devices' AD767 chip. The AOC-8 card provides 8 analog channels whose output can be independently set to one of the four ranges: 0-5V, 0-10V,  $\pm 2.5V$ , and  $\pm 5V$ . AD767 is a 12 bit resolution D/A converter chip complete with on-chip data latch and an output amplifier (refer to Appendix C for details). For the 10 V output range, 12 bit resolution provides a minimum discrete step size of about 2.5 mV. The output amplifier is capable of providing  $\pm 5$  mA continuous current. Therefore, any actuator device connected to the DAC should present at least  $1000\Omega$  input impedance. Table 2.5 gives the channel allocation and output range for the APSR servo setup.

#### 2.4.2.6. Digital Computer

The digital control and data acquisition functions are performed by an IBM compatible personal computer (PC). This computer has an Intel 486, 66 MHz DX/2 micro-processor, 4 MB of random access memory (RAM), a Maxtor 250 MB hard disk, and a 14" Super VGA color monitor. Three EISA compatible expansion card slots on the motherboard accept analog-to-digital (ADC) and digital-to-analog (DAC) converter cards described earlier. Two of these slots are occupied by the AD1170, ADC cards providing a total of 32 differential analog input channels for reading transducers. The remaining slot is taken up by the AD767, DAC card which provides 8 analog output channels.

### 2.4.3. APSR Control Software

As mentioned earlier, the PC controls various aspects of the APSR test and also acquires, processes, and displays data while the test is running. Managing all these tasks, smoothly and efficiently, requires a sophisticated real-time computer program. A computer program is classified as 'real-time' when its usefulness depends not only on the 'correctness' of an internal computation (algorithm), but also on the time at the instance of the computation or the time taken to execute the computation. Since a program that performs digital feedback control has to process data as it arrives and send appropriate response to the outside world at the *correct time*, it falls into real-time category. In addition to managing time, the following requirements were identified for the APSR control software.

- a. It should integrate feedback control, data acquisition, data processing, and graphical presentation of data in one seamless environment. The ability to visualize incoming data during execution of a test would allow the operator to make appropriate corrections before it is too late.
- b. It should be relatively easy to modify the test configuration without the need for reprogramming. This is a very important consideration since frequent changes in hardware setup (i.e., the number of converter cards and sensors, sensor locations and calibration factors, feedback gains etc.) are very common in a research environment.
- c. It should present a standard, easy-to-use interface to the user. Most modern software use some form of mouse-driven Graphical User Interface (GUI) with multiple windows, pop up menus, push buttons, and other control gadgets.
- d. It should be possible to reuse a large portion of the program while adopting the code to run different types of tests. The reusability requirement is

particularly important for a complex software which represents a significant investment of time.

All of these ideas have been incorporated into a software package called FlexCAT (Flexible Integrated Software Framework for Computer Aided Testing). The FlexCAT provides a software framework for developing laboratory test control and data acquisition application programs. The FlexCAT consists of more than 8000 lines of C language code which runs under PC-DOS® environment. It incorporates some of the essential features of modern object-oriented software design (Rumbaugh, 1991). An object-oriented design attempts to capture real world objects into corresponding software 'objects' which combine knowledge (data) and behavior (operations) into single entities. The FlexCAT program represents physical and logical objects from the control domain (data converter cards, sensors, servos, feedback loops, etc.) as monolithic data structures and allows meaningful operations (calibrate, test, run, stop, etc.) to be performed on them. The APSR control software is embedded within the FlexCAT and uses facilities provided by the latter to accomplish its tasks. The next two sections describe the overall structure of the FlexCAT and the main user interface of APSR program. Appendix D provides a more detailed discussion in the form of a user's guide to the FlexCAT and also includes a complete code listing.

#### 2.4.3.1. Structure of FlexCAT

The FlexCAT environment is comprised of three parts as shown in Figure 2.22: a) the real-time control environment, b) the execution environment, and c) the graphical user interface facility. The main purpose of the real-time control environment is to manage time and to maintain a database of important test parameters on the hard disc. The time management functions rely on a commercially available library called PCHRT™ supplied by Ryle Design. The execution environment largely consists of C

language constructs and run-time library routines which are used to control the program flow and perform computations. The graphical user interface facility is provided through another commercial library called LabWindows™, marketed by National Instruments Inc. It consists of a graphical user interface editor which is used to create interface elements called *controls* (such as pop up menus, push buttons, panels etc.), and a collection of precompiled library routines which perform operations on these controls. The use of commercial libraries provides access to thoroughly tested code and saves considerable programming time.

As discussed in Section 2.4.1, digital control of a dynamic process involves periodic communication with input/output devices and performing calculations at regular time intervals. Availability of a real-time clock is, therefore, a must for any control and data acquisition program. There are two basic software schemes for executing a set of periodic tasks; polling loop and interrupt mechanism.

The first scheme is the simpler of the two and is employed by most simple, real-time programs. The program stays within an infinite (closed) loop. If the total time taken to perform the useful tasks is  $t$  and the interval at which these tasks are to be performed is  $T$ , then it is necessary that  $t \ll T$ . After performing useful tasks, the control is transferred to a polling routine where the program checks (polls) the clock continuously to see if the time elapsed since it last visited the polling routine is equal to  $T$ . The program then breaks out of the polling loop and starts the process all over again. This method of synchronizing with the external clock has the following disadvantages: a) time spent in the polling loop is effectively lost since the computer cannot respond to any user commands during that time, b) program flow becomes exceedingly complex if there are several periodic tasks each having a different cycle time

T, and c) if for any reason,  $t$  exceeds  $T$  for a particular cycle, the resulting error is cumulative.

The interrupt mechanism is a hardware scheme which allows an external device to send a message (interrupt) to a computer to get its attention. Once the interrupt is received, the computer suspends whatever it is doing and starts executing a special function called interrupt service routine (ISR). When the computer is finished executing the ISR, it resumes the previously interrupted task. Most modern computers have some sort of clock-driven interrupt scheme wherein the clock interrupts the computer at a regular time interval (about 53.7 milliseconds in an IBM PC ). It is possible to use this interrupt to synchronize with the clock in the form of a foreground-background scheme.

The FlexCAT flow diagram shown in Figure 2.23, illustrates the clock driven background-foreground scheme. The program spends most of its time in a foreground infinite (closed) loop performing low priority activities such as data analysis and display. At any point in the foreground loop, a clock interrupt can send the program control to background to execute time critical tasks including the control and data acquisition. In order to increase the time resolution, FlexCAT reprograms the PC's clock chip in order to execute a clock interrupt every 1 millisecond. It then installs a special scheduler/dispatcher code in the clock interrupt service routine (ISR) which is executed one thousand times in a second. The scheduler maintains a list of up to 8 background tasks and dispatches them at appropriate time for execution. In the APSR setup, the background tasks include reading each analog-to-digital card every 170 milliseconds; performing control calculations and updating the digital-to-analog card every 3 seconds; and saving data on the hard disc every 90 seconds. As the time spent

executing background tasks is only a small fraction of the total time, computer always appears to be in the foreground mode, ready to respond to the user commands.

#### 2.4.3.2. APSR Interface

Figure 2.24 is the screen output of the main APSR user interface. A menu bar at the top gives access to various facilities provided and managed by the FlexCAT module. Through the menu commands the user can:

- a. enter and edit test parameters including sensor names and calibration factors, feedback loop constants, servo voltage limits etc.
- b. calibrate and test individual components such as ADCs, DACs, sensors, servos, and feedback loops.
- c. set up and run data acquisition tasks and view the test data in the form of graphs.

The rest of the screen is specific to the APSR test and is logically organized into five non-overlapping panels. Three peripheral panels correspond to each principal stress directions while the central panel corresponds to the inclusion position control. These panels have input/output fields for entering and displaying test parameters, selection lists for choosing options, and push buttons for activating control loops. The operator can use either mouse or keyboard to navigate through the menu and manipulate the screen elements (e.g. set target values, test mode etc.). A complete user's guide appears in Appendix D.

#### 2.4.4. Control Algorithm and Performance

This section presents a brief description of the digital control algorithm and the overall performance of the APSR computer control system. A rigorous approach to control system design is to obtain a mathematical model of plant dynamics and use it to



derive a control algorithm that satisfies stability and response criteria. While this approach is mathematically sound, it is also very difficult especially for a complex system with many hidden variables such as the APSR cell. An alternate approach is to use a generic form of the PID algorithm and use a trial and error procedure to obtain the optimal values of feedback loop constants.

The APSR control system uses a recursive form of the PID algorithm in which the current value of output variable  $Y_n$  is computed from the previous output variable  $Y_{n-1}$  and a correction term  $\Delta Y$  such that:

$$Y_n = Y_{n-1} + \Delta Y \quad (2.2)$$

Instead of computing the absolute value (position) of the output variable at each iteration, this so called "velocity algorithm" computes only the correction term  $\Delta Y$  as:

$$\Delta Y = K_P(E_n - E_{n-1}) + K_I E_n + K_D(E_n - 2E_{n-1} + E_{n-2}) \quad (2.3)$$

where error signal,  $E = \text{Target Value} - \text{Actual Value}$ , and subscripts  $n$ ,  $n-1$ , and  $n-2$  denote values at time  $T$ ,  $T-1$ , and  $T-2$  respectively. The constants  $K_P$ ,  $K_D$ , and  $K_I$  represent the contributions of proportional, integral, and derivative actions respectively. The integral and derivative actions can be deactivated simply by equating  $K_I$  and  $K_D$  to zero. The reader is referred to Appendix B for the derivation of equation 2.3.

#### 2.4.4.1. Platform Pistons

Four independent feedback control loops drive platform pistons at constant displacement rate (velocity) during a standard APSR test. In this control loop, feedback is in the form of signal from a LVDT (e.g. D3, D4; Figure 2.15) while the controlled parameter  $V_P$  is the platform velocity. Since the controlled parameter is the time derivative of the measured variable, the controller is continuously chasing a

moving target. This type of control requires a full PID controller for which the constants,  $K_P$ ,  $K_I$ , and  $K_D$  are all obtained by trial and error (Table 2.6).<sup>2</sup> It is possible to get a jump start on the control process by guessing the initial value of the output variable and using it as  $Y_{n-1}$  in equation 2.2 for the first iteration ( $n = 0$ ). Figure 2.25a shows the open loop (feed-forward) response of the servo controller that controls the platform pistons. The initial value of output voltage is estimated from slope of this plot as:

$$Y_0 = 2.0 * V_P \quad (2.4)$$

where,  $V_P$  is the desired platform velocity in mm/min. Figure 2.26 shows the performance of platform piston control system. For the target displacement rate of 0.01 mm/min., the system is able to control all four pistons within  $2\mu\text{m}$  of their target positions. The normal displacement rate during a test is 0.05 mm/min., while the fastest possible rate is approximately 2.0 mm/min.

#### 2.4.4.2. Active Plane Strain

Since the purpose of the active plane strain control is to keep the sidewalls in a fixed position, simple proportional control is sufficient (i.e.,  $K_I = K_D = 0$ ). When the controller is activated for the first time, it takes the top and bottom proximity sensor readings and designates them as the new targets for respective sidewalls. Any subsequent deviation from this reading generates an error signal. Table 2.6 gives sensor and servo channel numbers and values of feedback parameters for plane strain

---

<sup>2</sup> The values of the PID constants depend upon the system of units used in expressing the control (input/output) parameters. The values listed in Table 2.6 are for the SI system.

control loops. Figure 2.27 illustrates the performance of the sidewall control system for two typical tests on reinforced sand. The out-of-plane strain is kept within 0.001% which corresponds to less than 1  $\mu\text{m}$  displacement of individual sidewall. The measurement of the intermediate principal stress is reflected in the parameter  $b = (\sigma_2 - \sigma_3)/(\sigma_1 - \sigma_3)$ . Figure 2.27b shows consistent and repeatable results from two tests (APSR 54N, 55N) in which  $b$  parameter is initially set to,  $b = 1$  ( $\sigma_2 = \sigma_1$ ). During shearing,  $b$  decreases rapidly to a minimum of 0.25 and approaches 0.4 as the soil specimen approaches failure.

#### 2.4.4.3. Air Pressure

The confining stress,  $\sigma_{xx}$  is maintained by a proportional-integral (PI) feedback loop which controls an electro-pneumatic pressure regulator. Again, the control action converges very rapidly if the initial output value,  $Y_{n-1}$ , can be estimated beforehand. Figure 2.25b shows the open-loop response of the Fairchild electro-pneumatic regulator. For a target pressure  $P_T$  in kPa, an approximate value of initial output value can be obtained from this calibration curve as:

$$Y_0 = 1.5 + P_T / 24.0 \quad (2.5)$$

The steady state response of the controller is shown in Figure 2.28. It can be seen that the controller is able to maintain the steady state value of confining pressure within 0.05 kPa (0.16 % of the target value).

#### 2.4.4.4. Reinforcement Position

The reinforcement controller, like the active plane strain controller, strives to maintain a constant position. It is also based on simple proportional control algorithm with integral and derivative constants set to zero (i.e.,  $K_I = K_D = 0$ ; Table 2.6). Figure

2.29a shows the speed of the reinforcement actuator arm as a function of servo motor command voltage. The response time  $T_R$  of the controller can be estimated by combining this feed-forward (open loop) response with the feedback parameter  $K_P$  as:

$$T_R = \frac{1}{K_P * S} \quad (2.6)$$

where,  $S$  is the slope of the plot in Figure 2.29a. The assumed value of  $K_P = 200$  (Table 2.6) gives characteristic responses time of the controller close to 1 second which is fast enough for the application. Figure 2.29b shows the performance of the reinforcement control system for three different inclusion materials. The exit point displacement as measured by the infrared sensor is kept well within  $1 \mu\text{m}$ .

Sensor No.	Sensor Name	Location	Channel No.*	Unit	Calibration Factor † (Unit/V/V <sub>in</sub> )	Range (Unit)	Resolution (Unit)
0	D1	Piston 1	0	mm	33.01829	50	5.8x10 <sup>-4</sup>
1	D2	Piston 2	1	mm	-32.67636	50	5.8x10 <sup>-4</sup>
2	D3	Piston 3	2	mm	-33.04792	50	5.8x10 <sup>-4</sup>
3	D4	Piston 4	3	mm	-33.60630	50	5.8x10 <sup>-4</sup>
4	P1	Piston 1	4	kPa	34500.64	0-700	0.47
5	P2	Piston 2	5	kPa	69597.59	0-1400	0.93
6	P3	Piston 3	6	kPa	69950.70	0-1400	0.93
7	P4	Piston 4	7	kPa	34442.66	0-700	0.47
8	PS1	Top Wall	8	mm	3.03996	10	2.5x10 <sup>-5</sup>
9	PS2	Bottom Wall	9	mm	2.97012	10	2.5x10 <sup>-5</sup>
10	P5	Top Wall	11	kPa	34442.65	0-700	0.47
11	P6	Bottom Wall	12	kPa	34366.80	0-700	0.47
12	D5	Airbag (Left)	13	mm	25.0035	26	4.3x10 <sup>-4</sup>
13	D6	Airbag (Cent.)	16	mm	24.8599	26	4.3x10 <sup>-4</sup>
14	D7	Airbag (Right)	17	mm	24.7899	26	4.3x10 <sup>-4</sup>
15	P7	Airbag	15	kPa	-8051.112	0-170	0.11
16	IRS	Reinforcement	10	mm	-0.194404	0.1	3.2x10 <sup>-6</sup>
17	LC	Reinforcement	14	kN	67.35444	2.2	1.47

\* Channel numbers greater than 15 are located on the second analog-digital converter card.

† The calibration factors are based on 6.0 V input voltage (V<sub>in</sub>) for all transducers except proximity sensors, PS1 and PS2 which require 12 V input.

Table 2.1: The APSR Cell Transducers.

Manufacturer	Duff-Norton Inc., Charlotte, NC
Model	M28630-8, Inverted Ball Screw Type
Rated Load	1000 LBS
Worm Gear Ratio	5:1
Raise for 1 turn of Worm	0.025 INCH
Maximum Stroke	6 INCH
Torque at Full Load	21 LB-INCH
Actuator Efficiency	20 %

Table 2.2: Specifications for the APSR Cell Linear Actuators.

Manufacturer	Robbins and Myers, Eden Prairie, MN.
Model	E 352-501-404
Gear Ratio	100:1
Max. Continuous Speed	50 RPM
Maximum Torque	14 LB-INCH
Maximum Terminal Voltage	28 V
Maximum Current	1 A continuous / 5 A peak
Torque Constant, Kt	4.7 OZ-IN/A
Voltage Constant, Ke	3.5 V/KRPM
Motor Inertia	0.00033 OZ-IN-SEC <sup>2</sup>
Tach. Voltage Constant	3.5 V/kRPM

Table 2.3: Specifications for the DC Servo Motors.

Manufacturer	Fairchild Inc., Winston, SC
Model	T5221-9
Supply Pressure	250 PSI Maximum
Output Pressure Range	
Minimum	0.35 - 1.75 PSI
Maximum	0 - 150 PSI
Maximum Output Capacity	40 CFM
Maximum Air Consumption	0.36 CFM
Input Voltage Range	0 - 9 V DC
Input Impedance	2740 OHMS
Linearity	$\pm 0.25$ % Full Scale

Table 2.4: Specifications for the Electro-Pneumatic Controller.

Servo No.	Servo Name	DAC Channel Number	Range (Volt)
0	Piston 1 Motor	0	$\pm 5$
1	Piston 2 Motor	1	$\pm 5$
2	Piston 3 Motor	2	$\pm 5$
3	Piston 4 Motor	3	$\pm 5$
4	Top Sidewall Motor	7	$\pm 5$
5	Bottom Sidewall Motor	6	$\pm 5$
6	Air Pressure Controller	4	0 - 10
7	Reinforcement Motor	5	$\pm 5$

Table 2.5: Digital-to-Analog Card Channel Allocation and Range.

Loop No.	Name	Sensor No.	Servo No.	$K_P^\dagger$	$K_D$	$K_I$
0	Piston 1	0	0	0.6	0.16	0.01
1	Piston 2	1	1	0.6	0.16	0.01
2	Piston 3	2	2	0.6	0.16	0.01
3	Piston 4	3	3	0.6	0.16	0.01
4	Top Wall	8	4	360	0	0
5	Bottom Wall	9	5	360	0	0
6	Air Pressure	15	6	0.05	0	0.01
7	Reinforcement	16	7	200	0	0

<sup>†</sup> The values of the PID constants depend upon the system of units used in expressing the control (input/output) parameters. The values listed here are for the SI system.

Table 2.6: The APSR Feedback Loop Parameters.



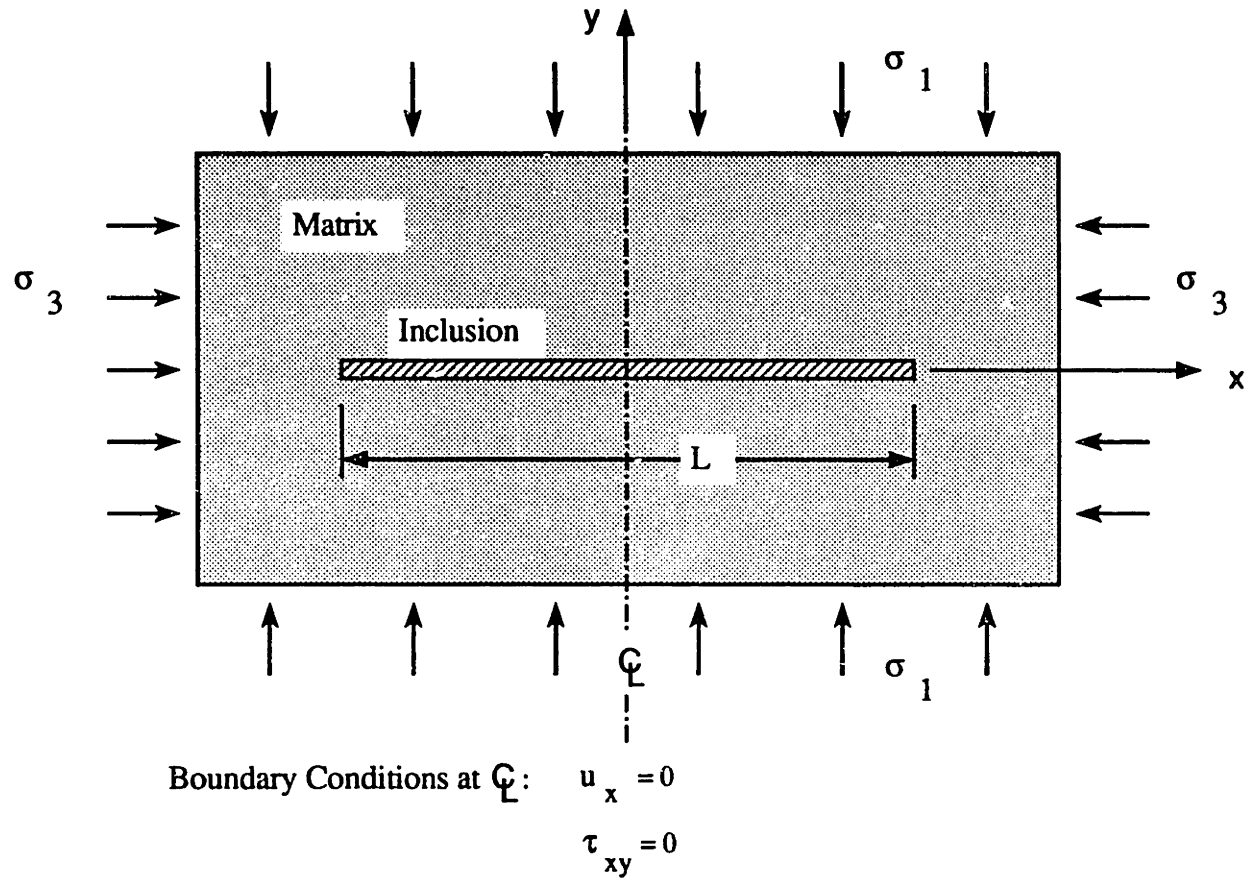


Figure 2.1: Geometry of the APSR Ideal Plane Strain Reinforced Soil Element.

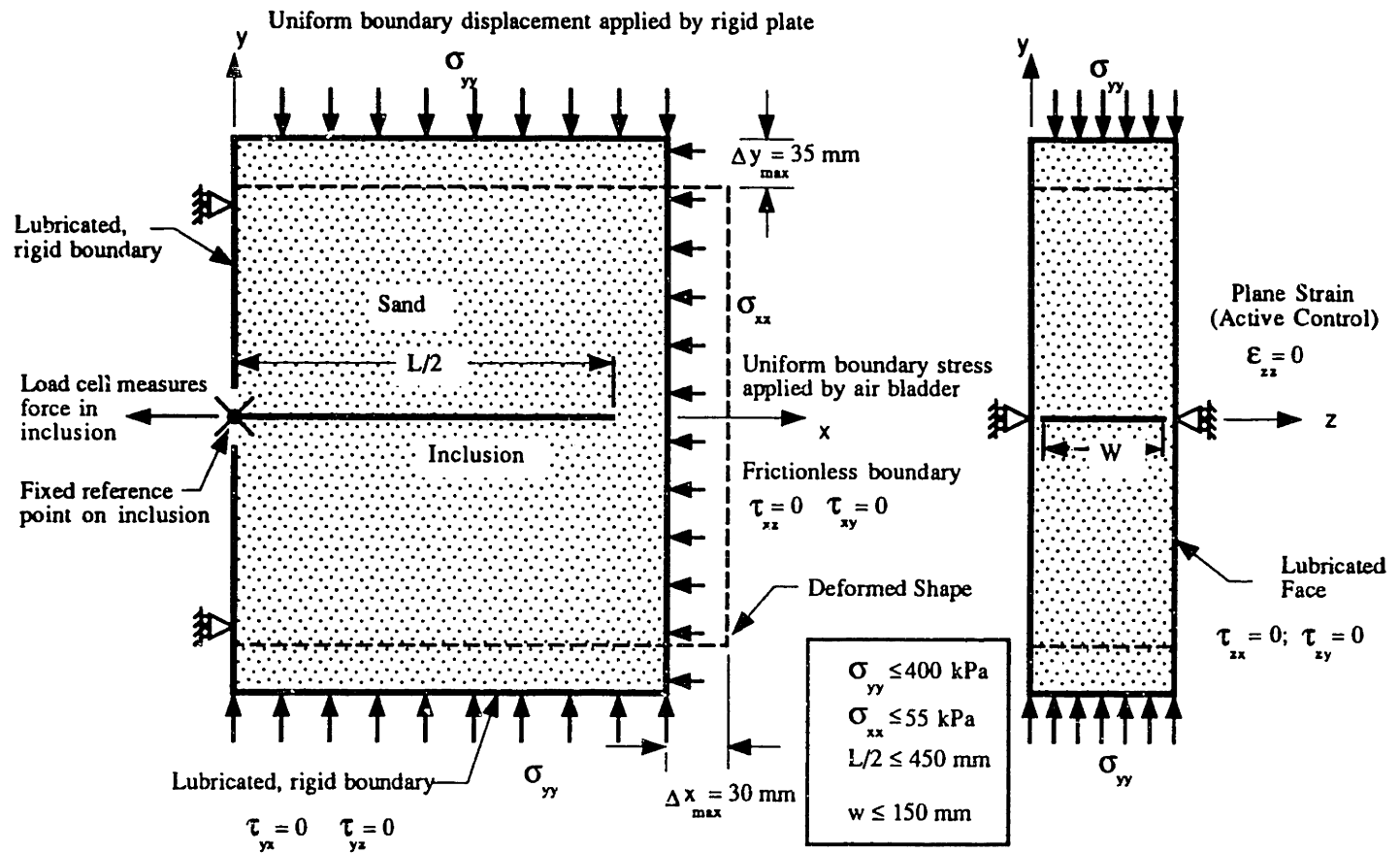


Figure 2.2: Conceptual Design of the APSR Cell (after Whittle et al., 1991).

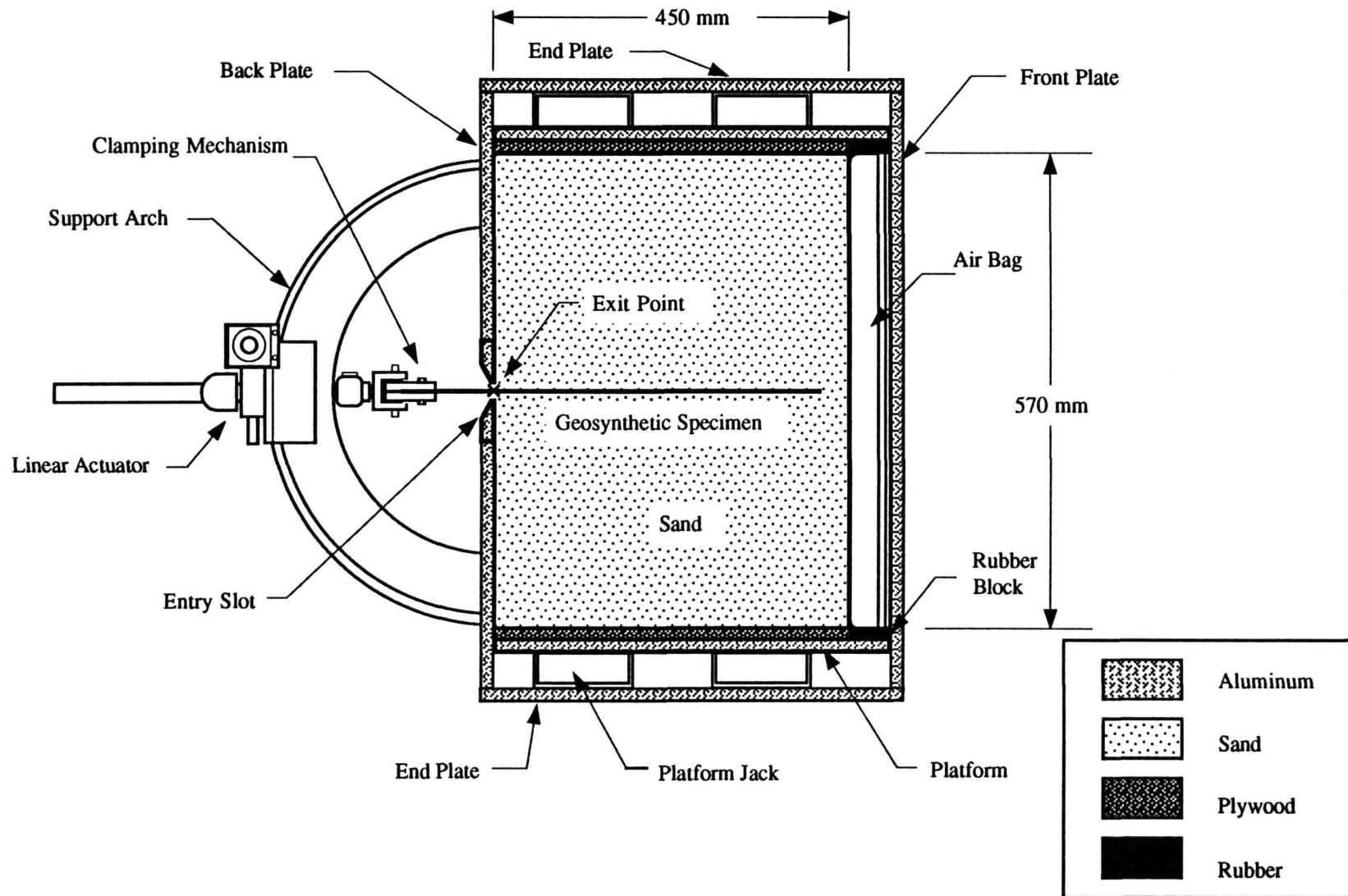
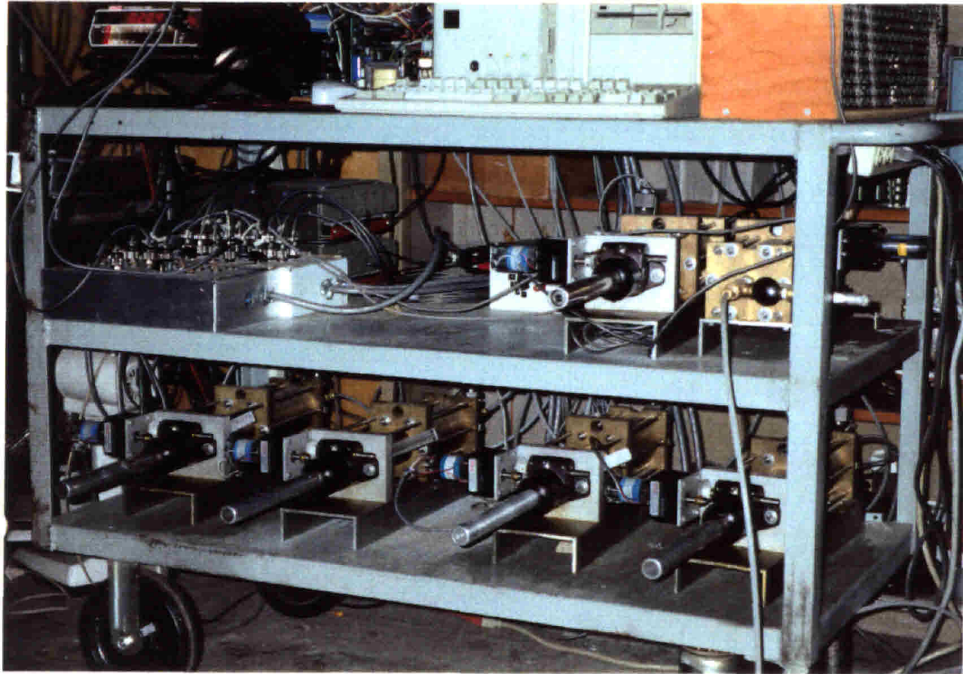
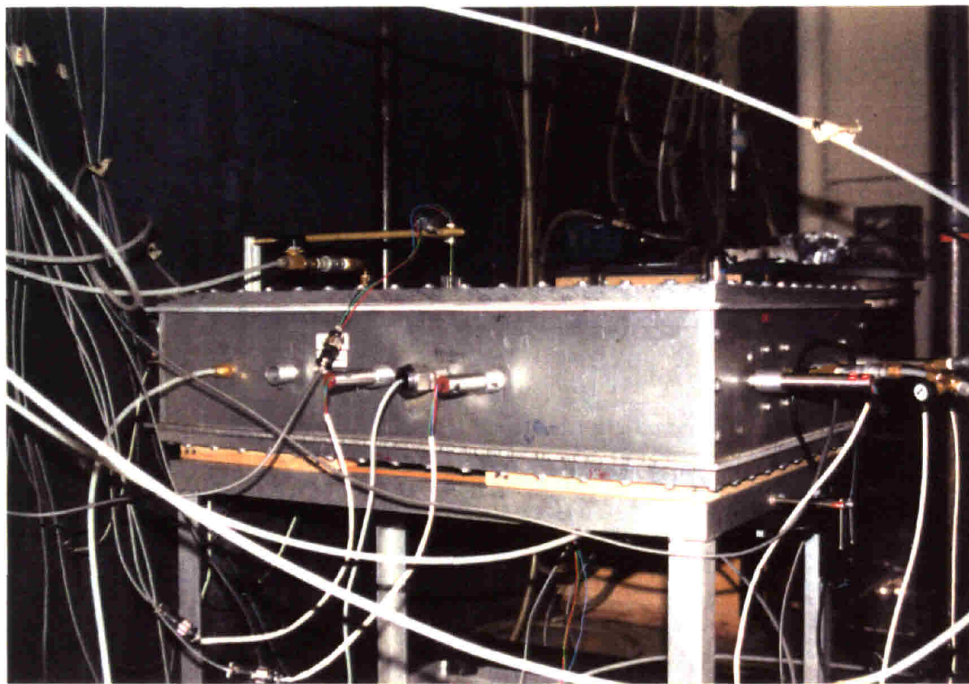


Figure 2.3: Cross Section Through the APSR Cell.





(a) The APSR Control Hardware.



(b) The APSR Cell.

Figure 2.4: Photographs of the APSR Cell and Control Hardware.



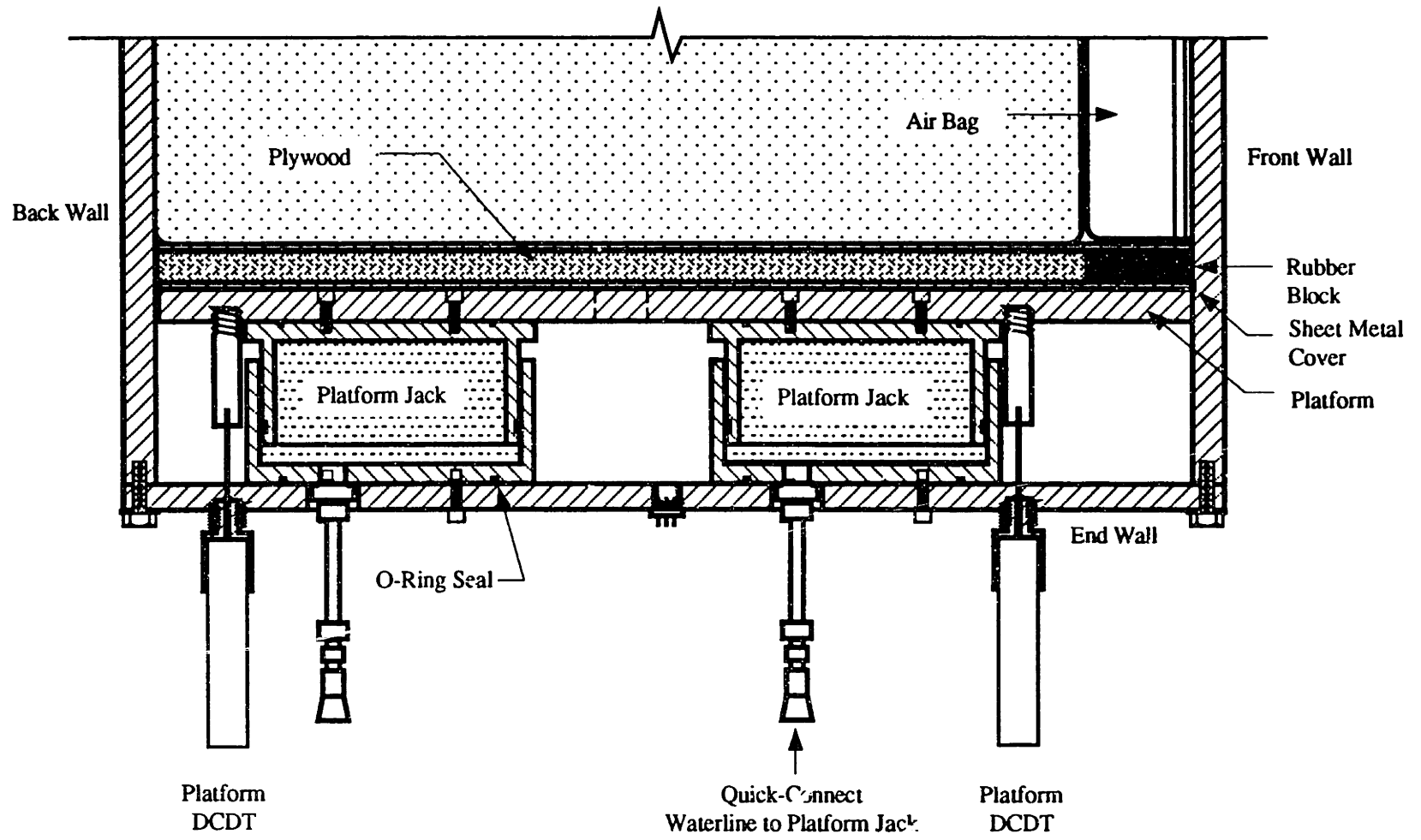


Figure 2.5: The APSR Cell Platform Assembly.

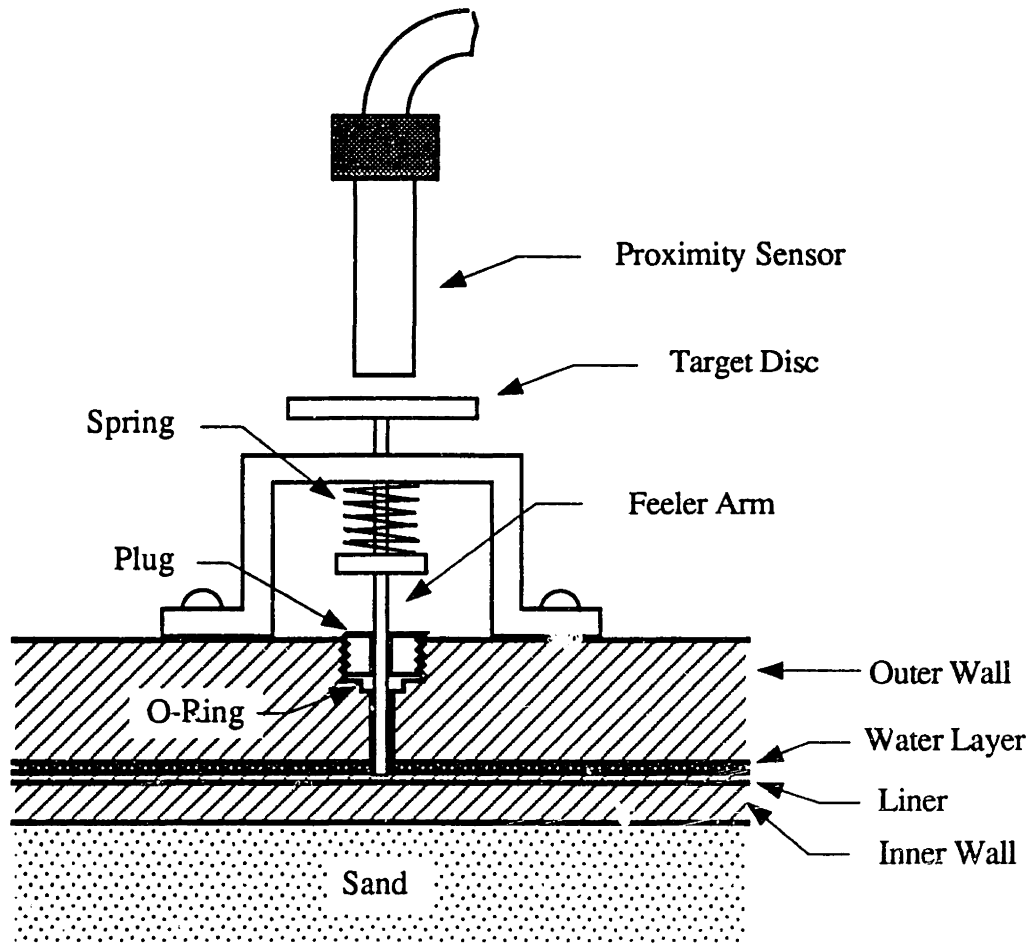


Figure 2.6: Detailed View of the Sidewall Positioning System.



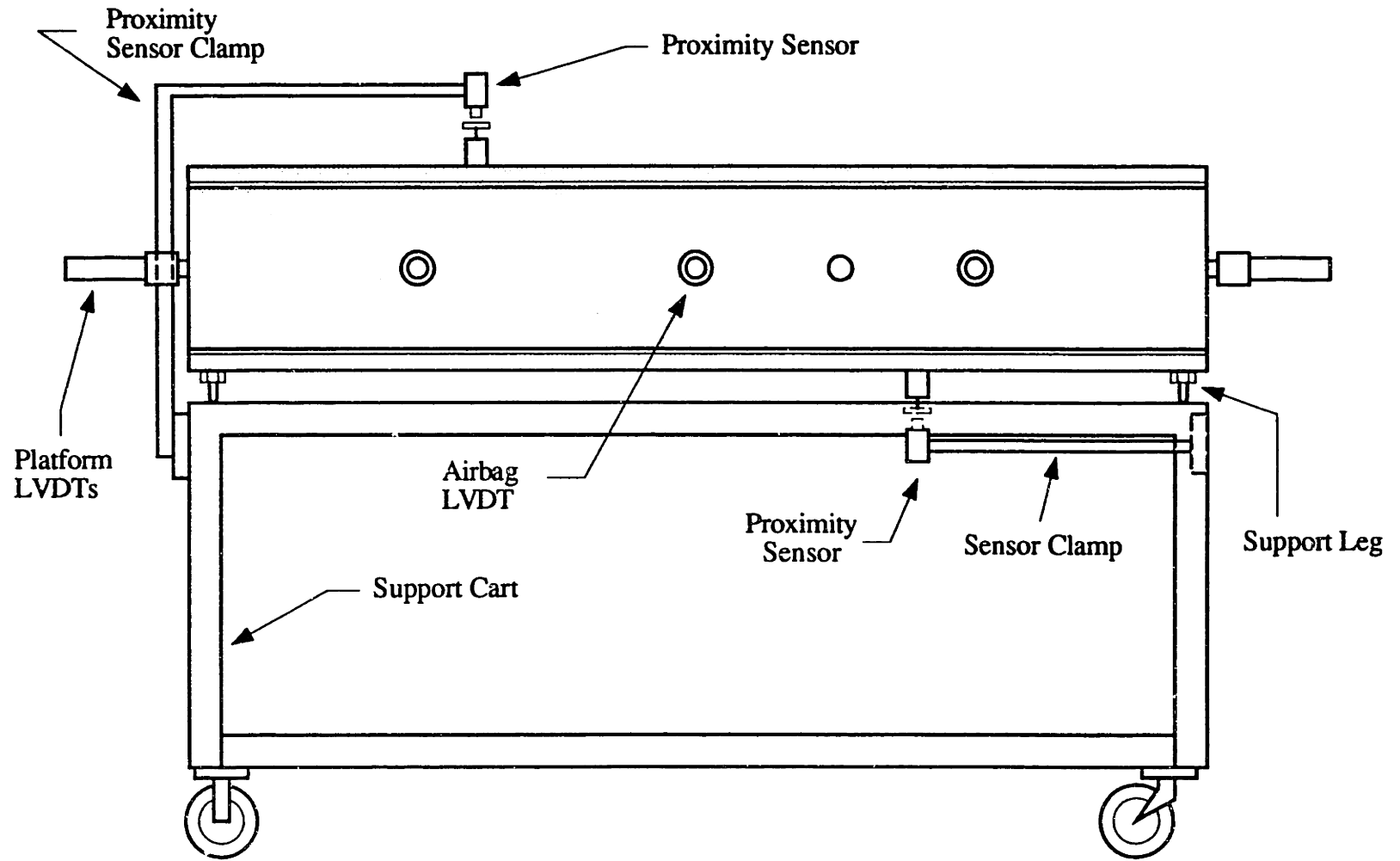


Figure 2.7: The APSR Support Cart and Sidewall Referencing System.

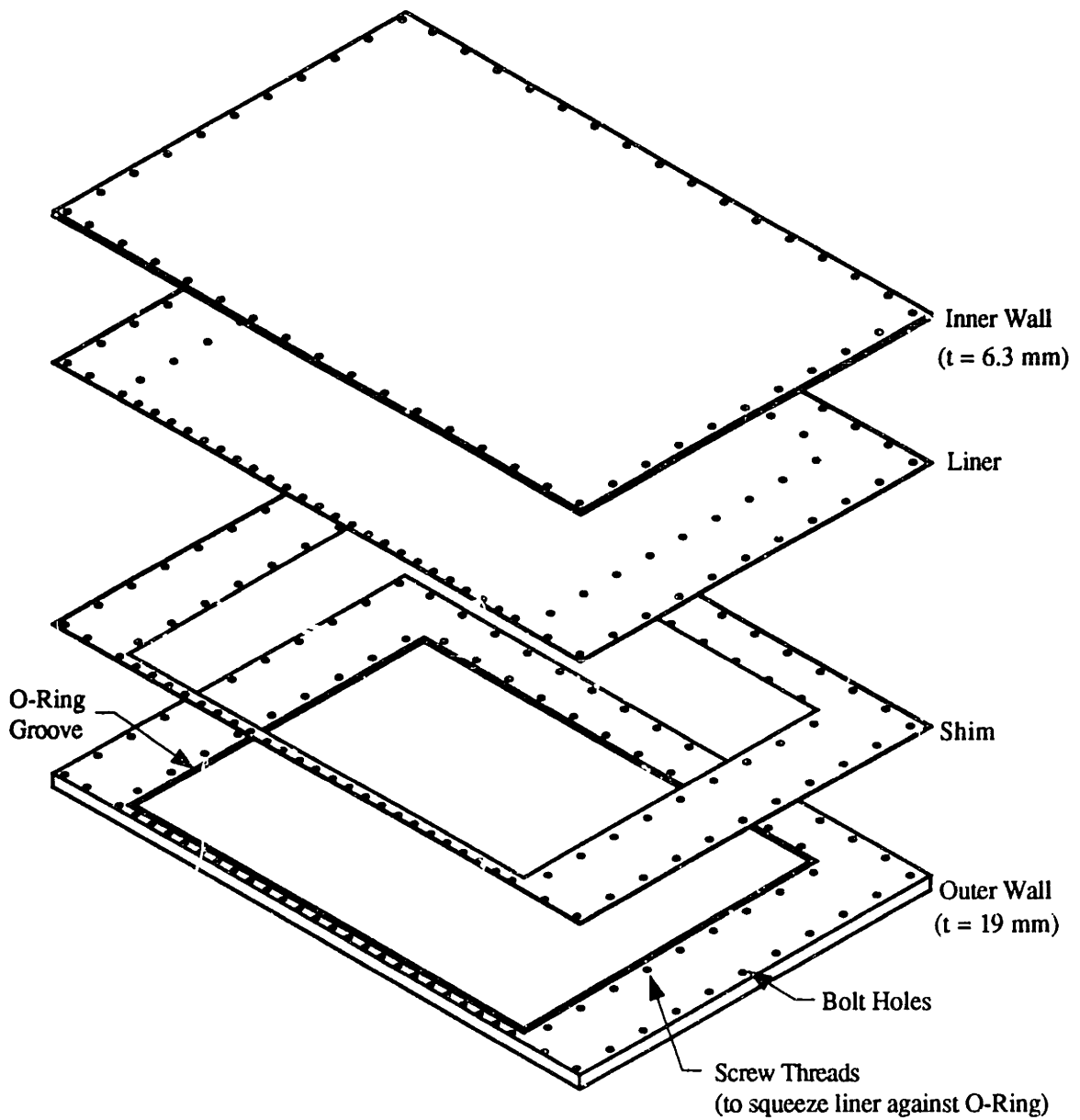


Figure 2.8: Exploded View of Plane Strain Sidewalls (after Larson, 1992).

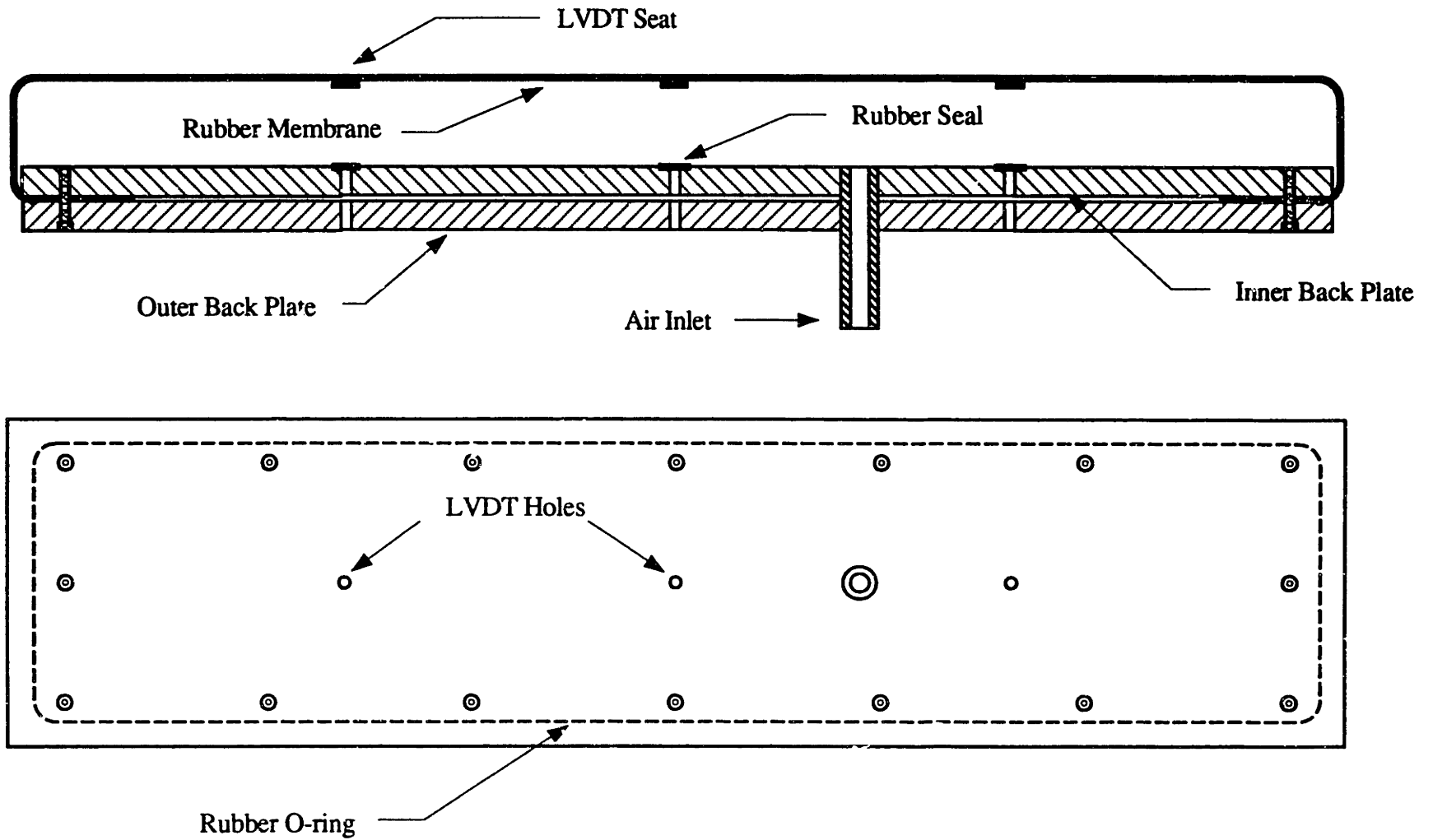


Figure 2.9: The APSR Cell Airbag Assembly.

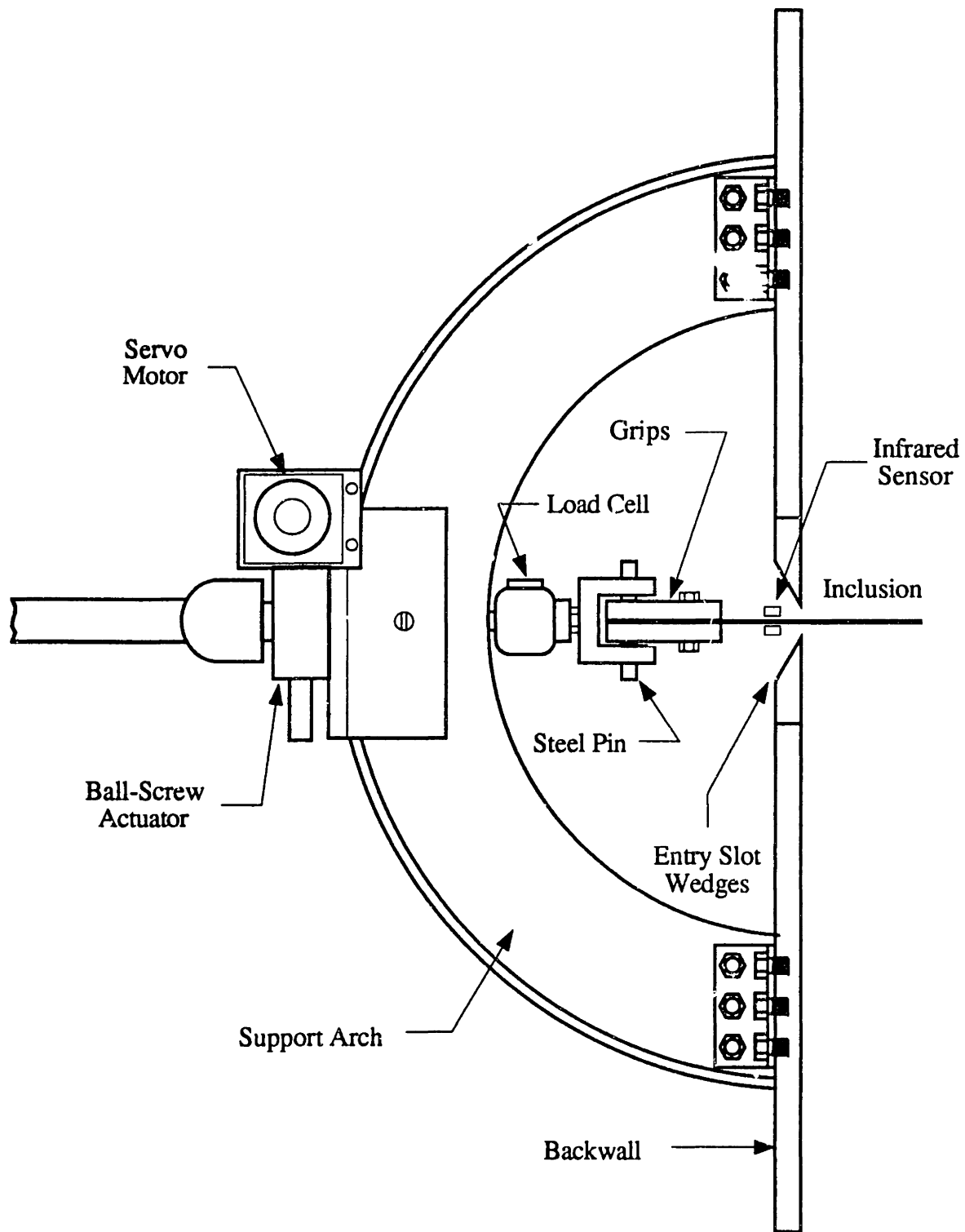


Figure 2.10: Reinforcement Entry Slot and Support Arch.

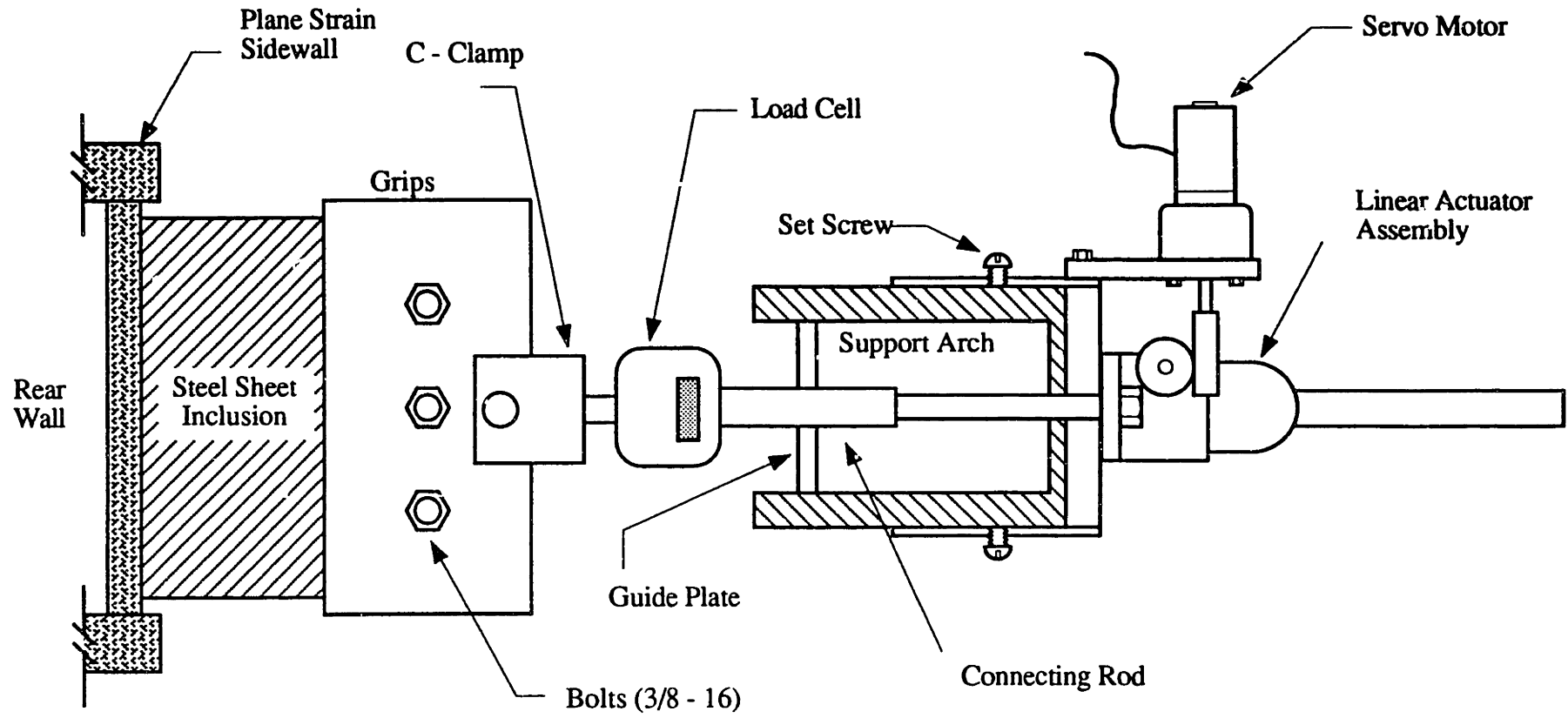


Figure 2.11: Reinforcement Control Mechanism.

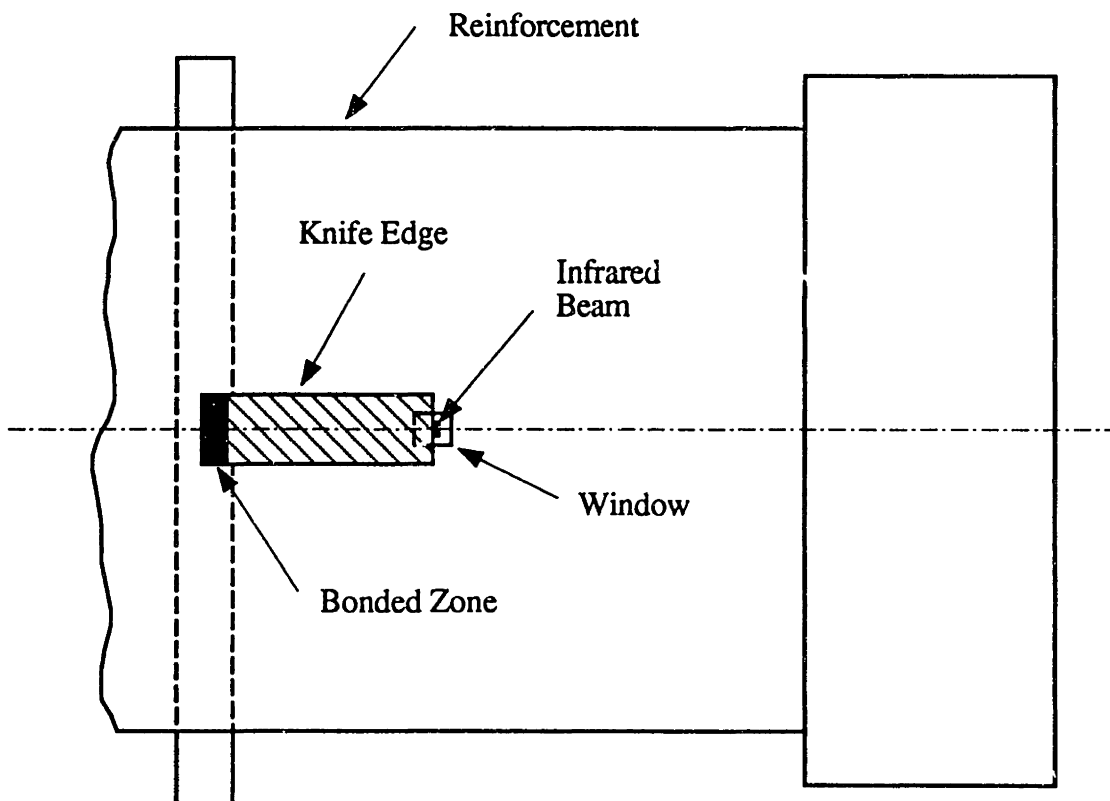
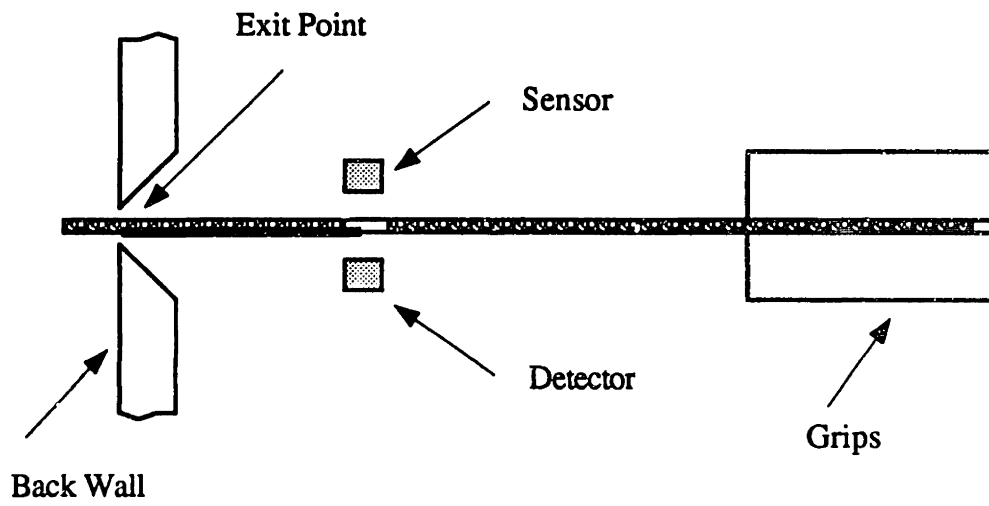


Figure 2.12: Principle of Infrared Position Detection.

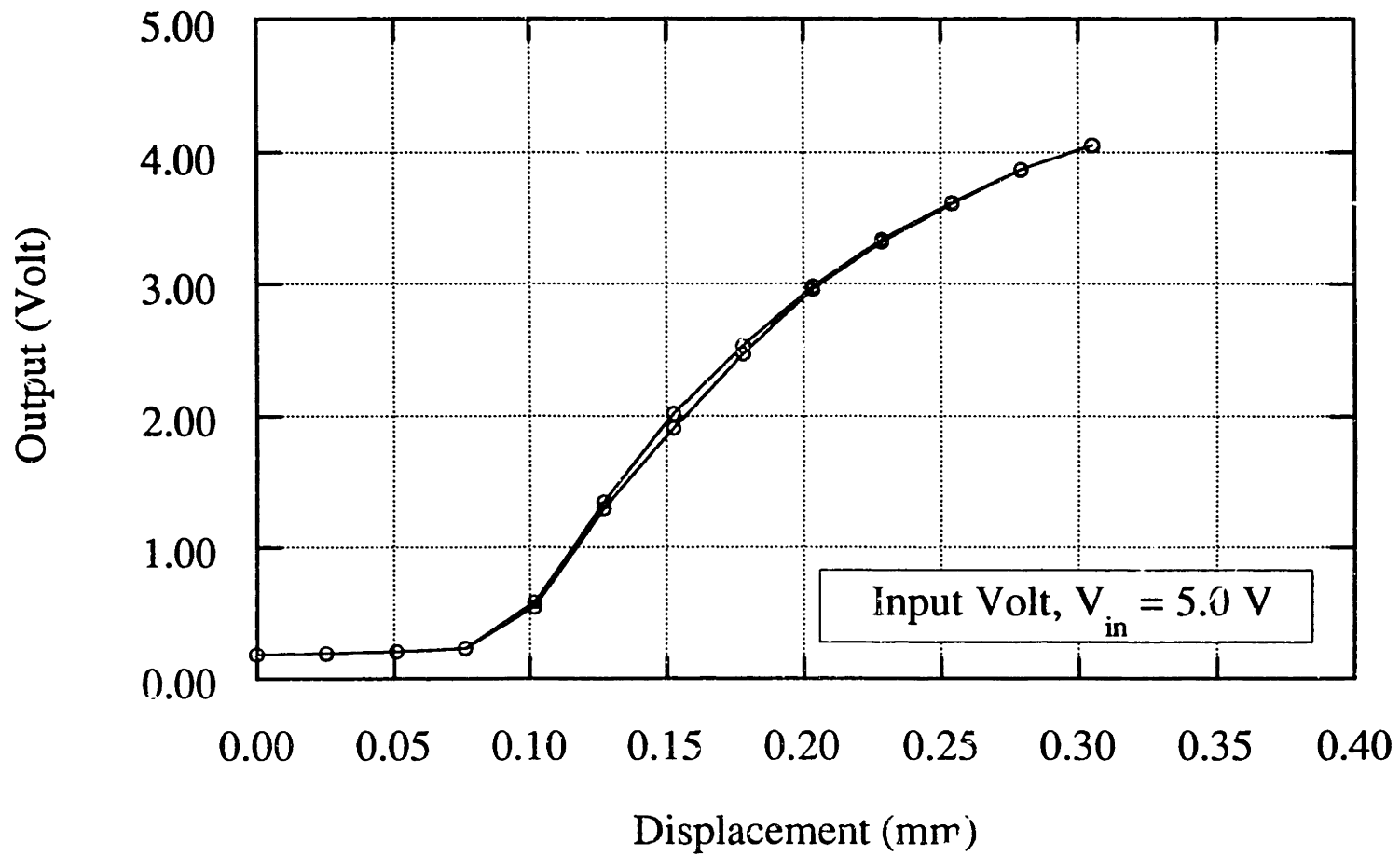


Figure 2.13: Calibration Curve for Infrared Sensor.

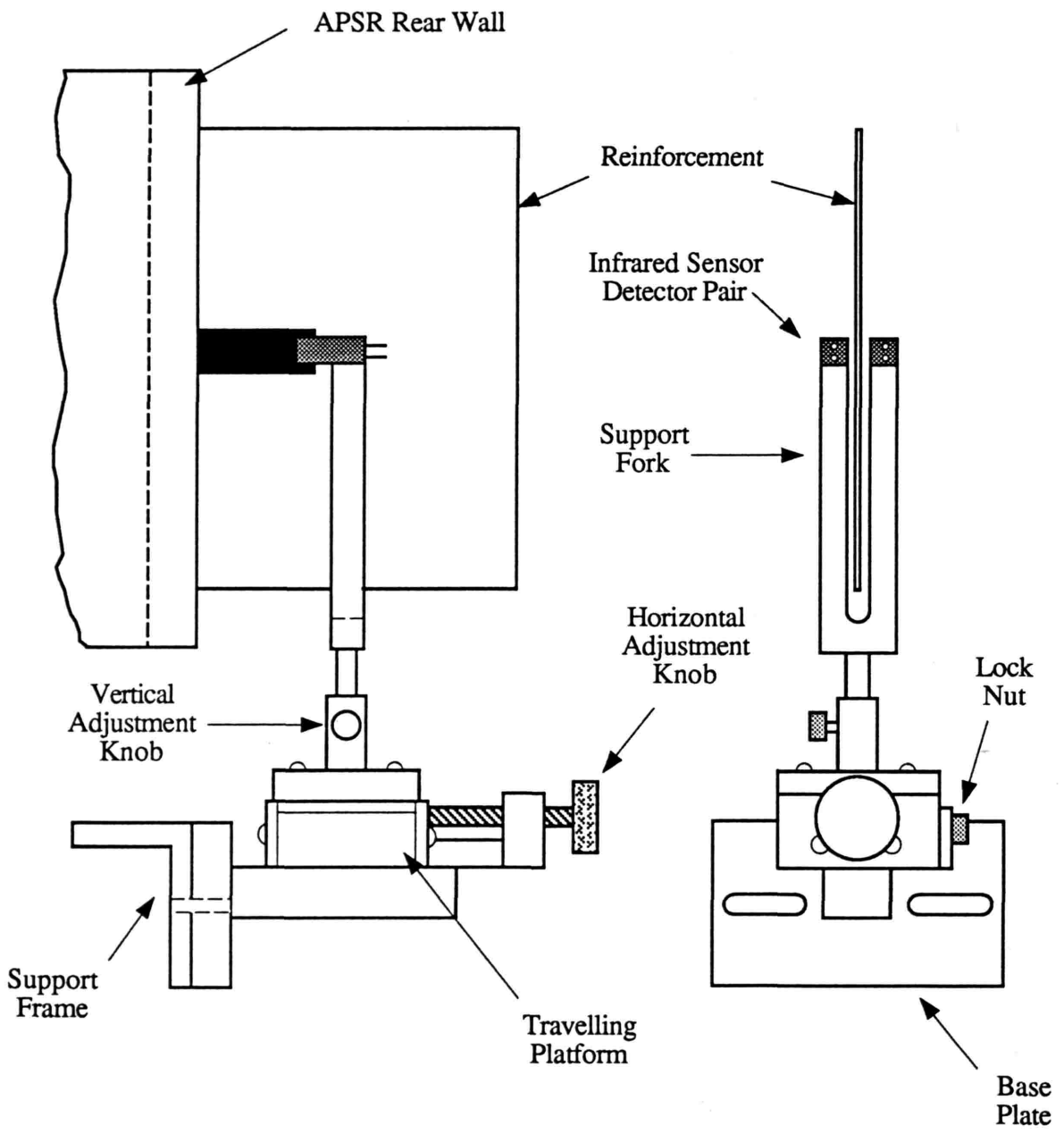


Figure 2.14: Infrared Positioning System Assembly.



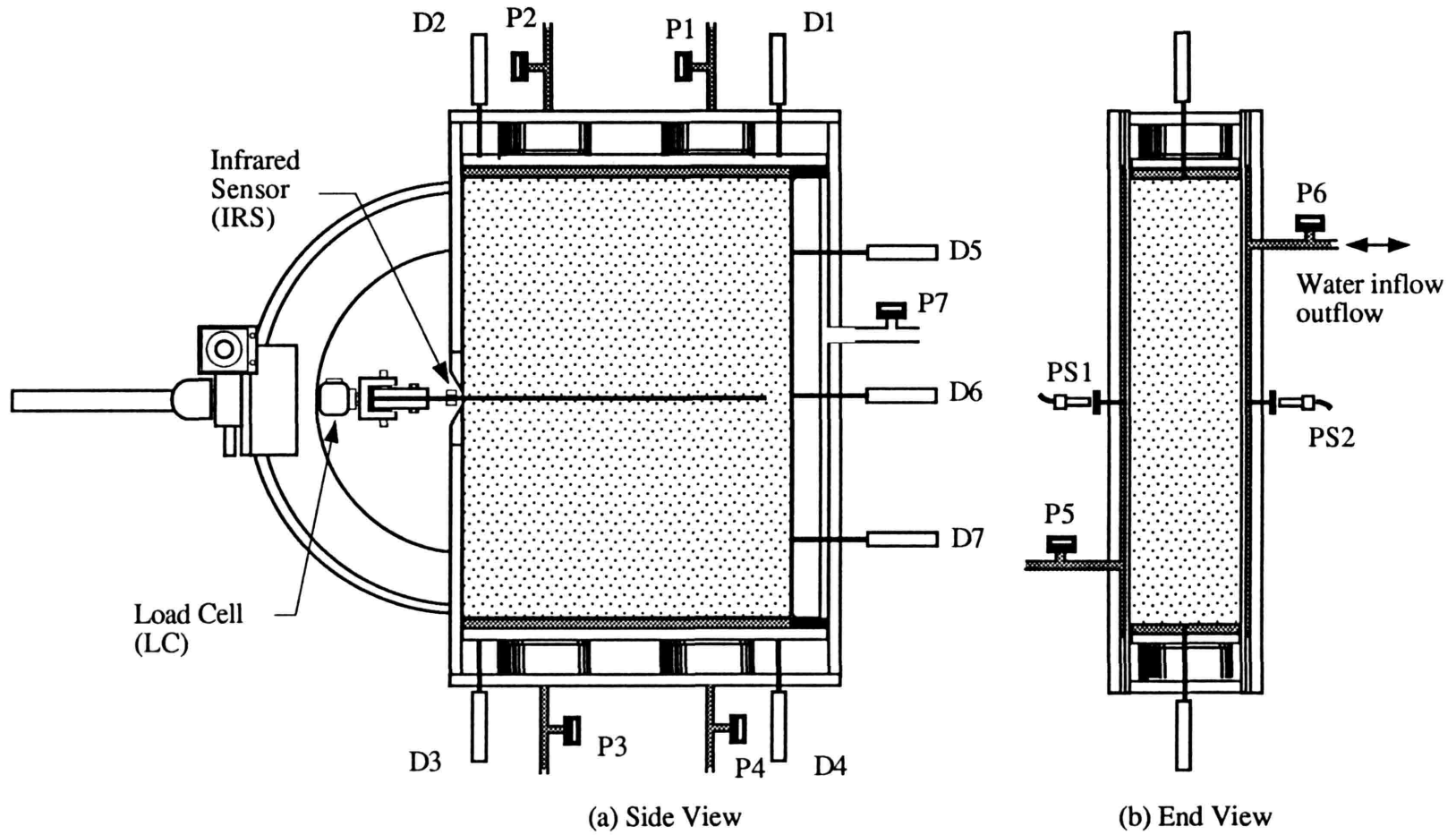


Figure 2.15: The APSR Cell Instrumentation.

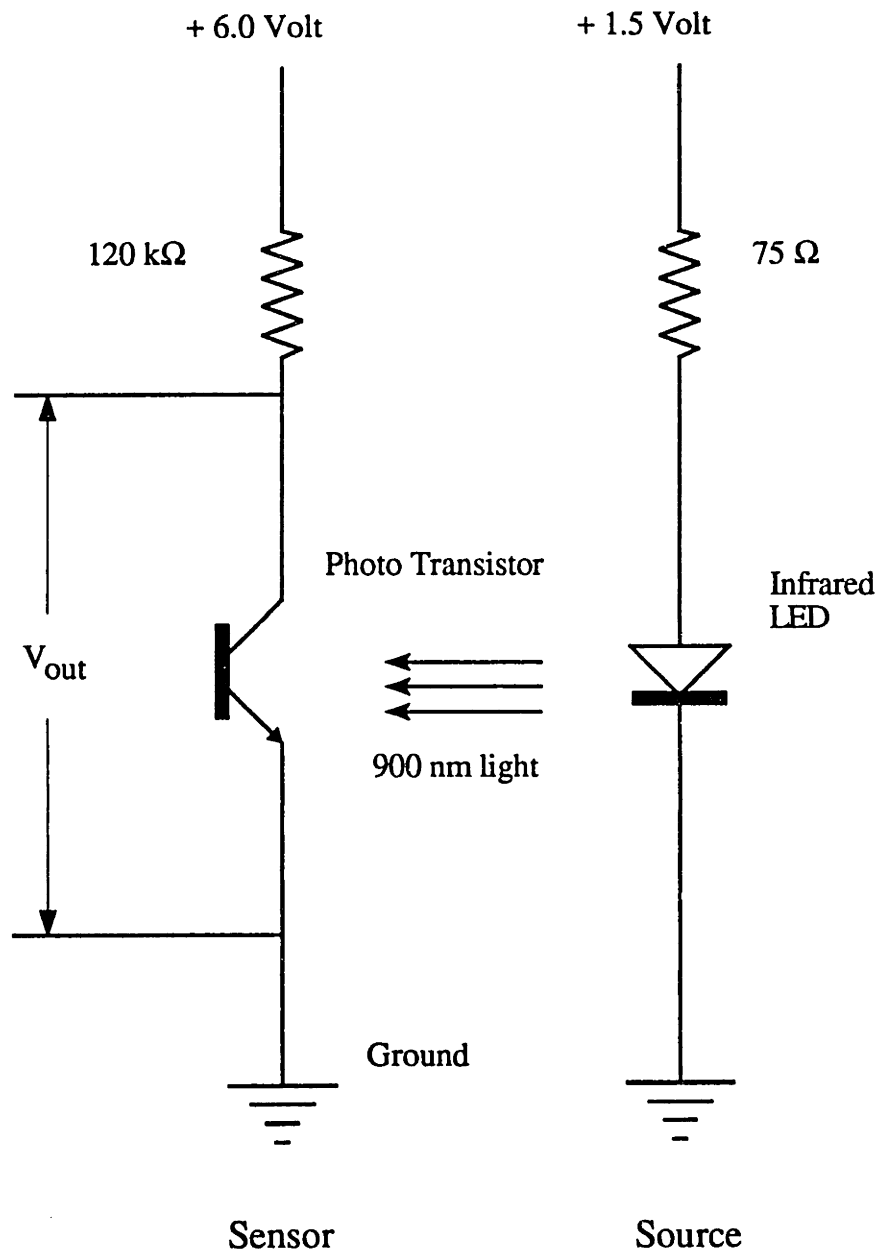


Figure 2.16: Circuit Diagram for Infrared Sensor Assembly.

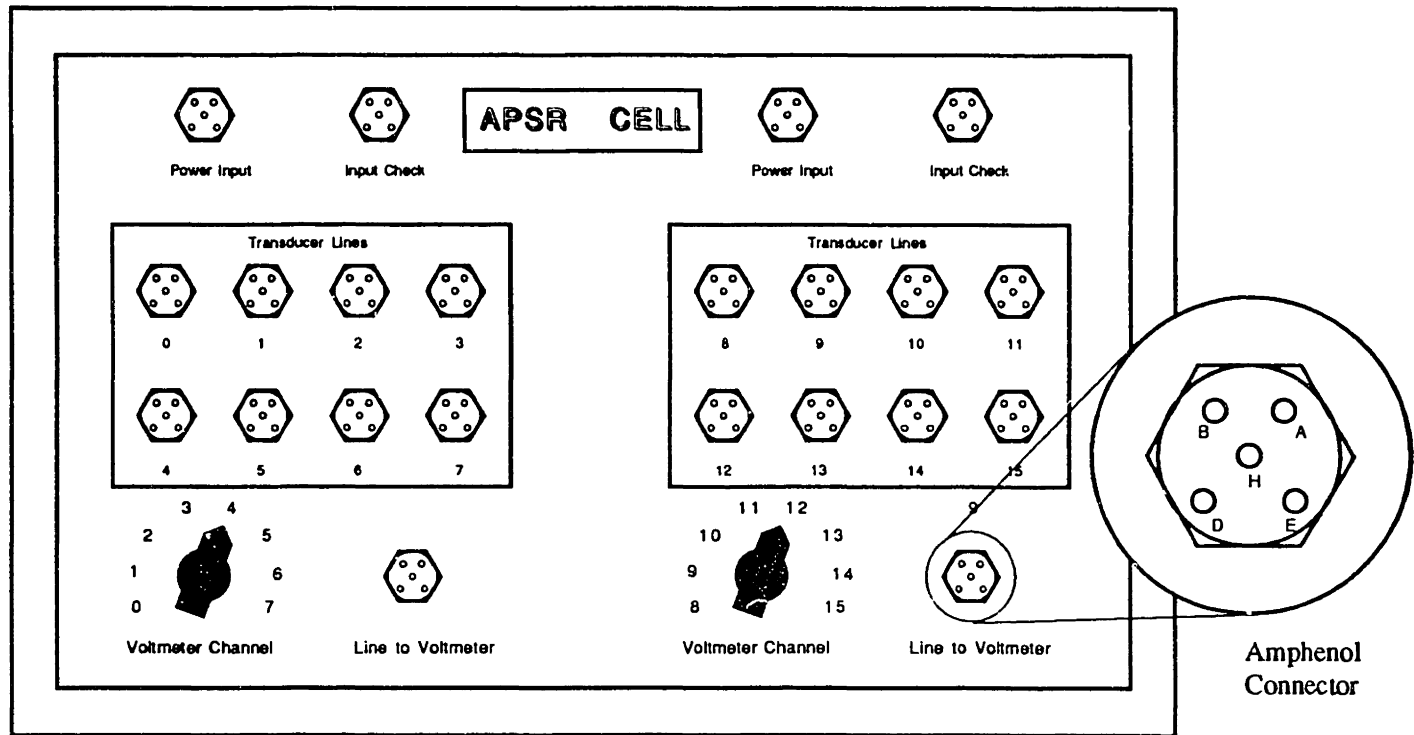
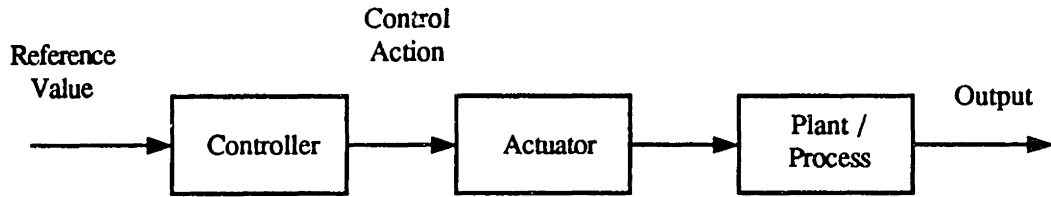
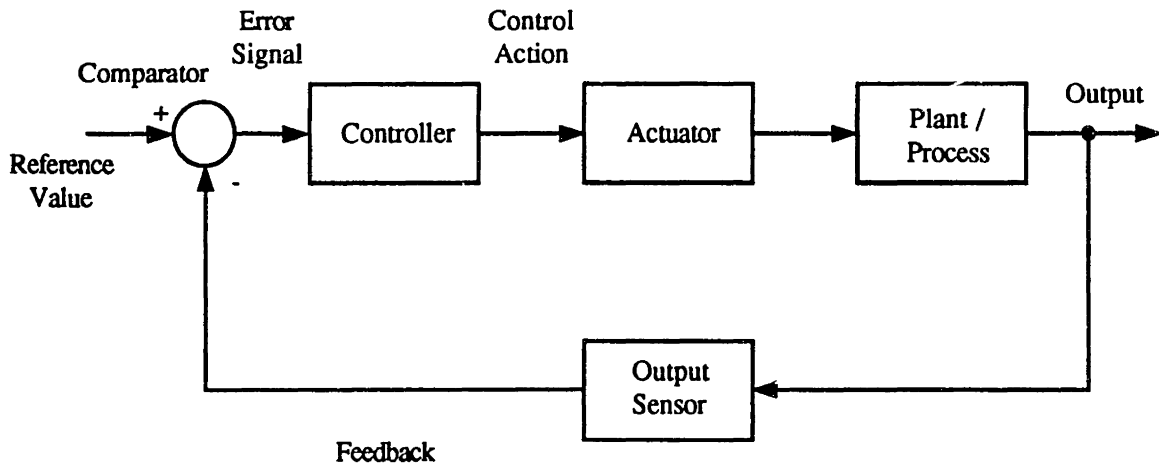


Figure 2.17: The APSR Junction Box.



(a) Feed-forward (Open Loop) Control.



(b) Feedback (Close Loop) Control.

Figure 2.18: Types of Control Systems.

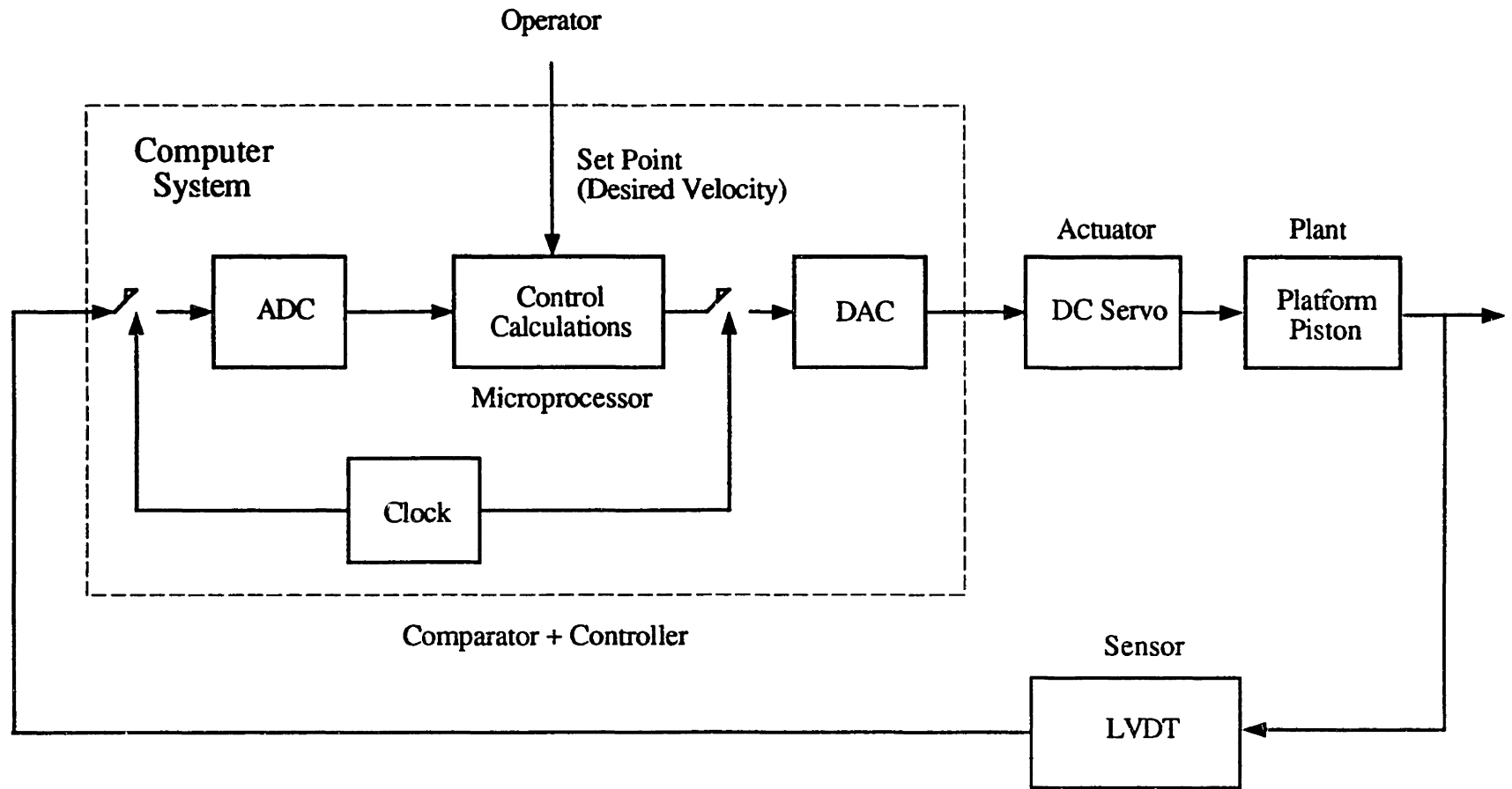


Figure 2.19: Platform Feedback Control Loop.

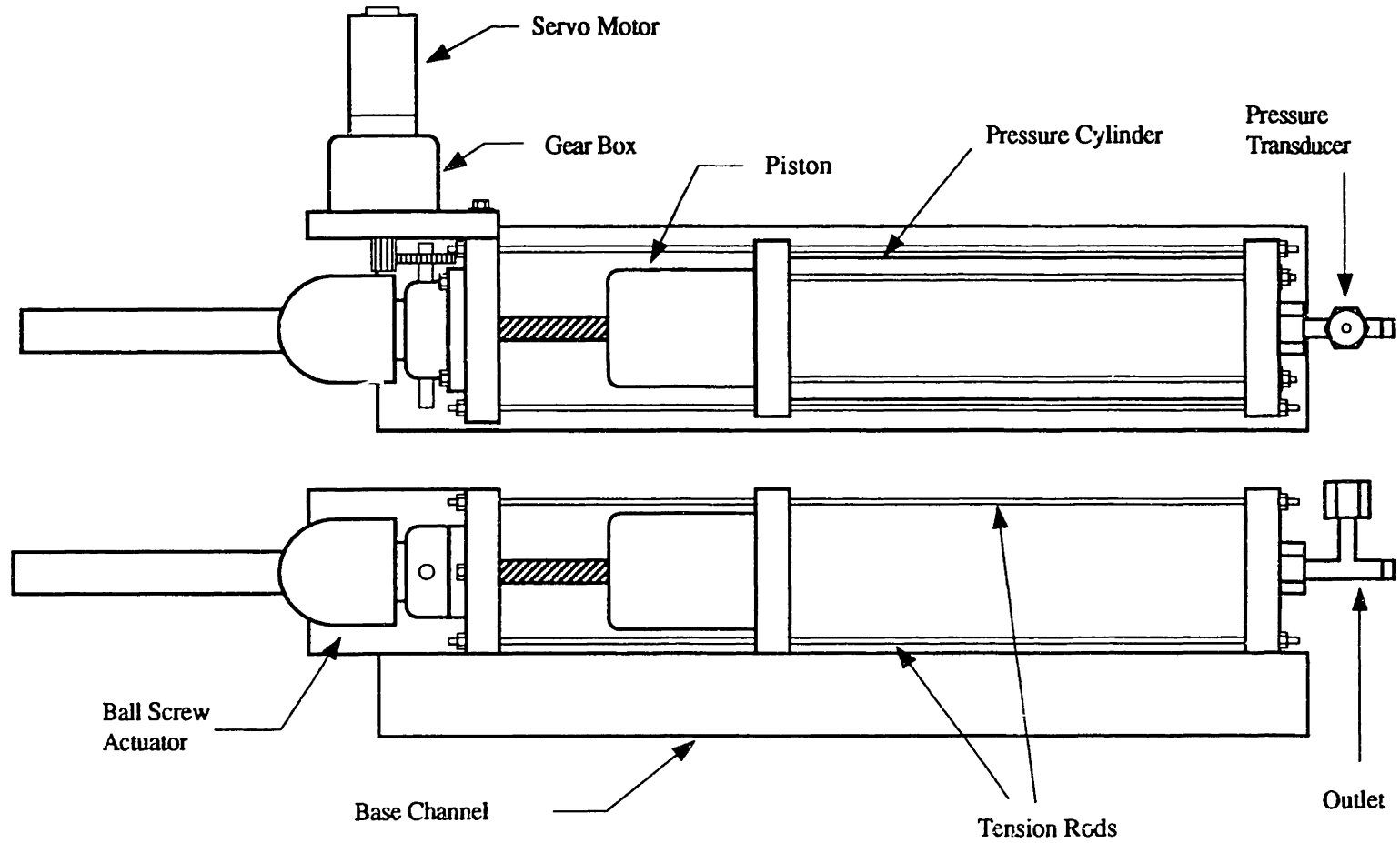


Figure 2.20: Schematic of the APSR Pressure-Volume Controller.

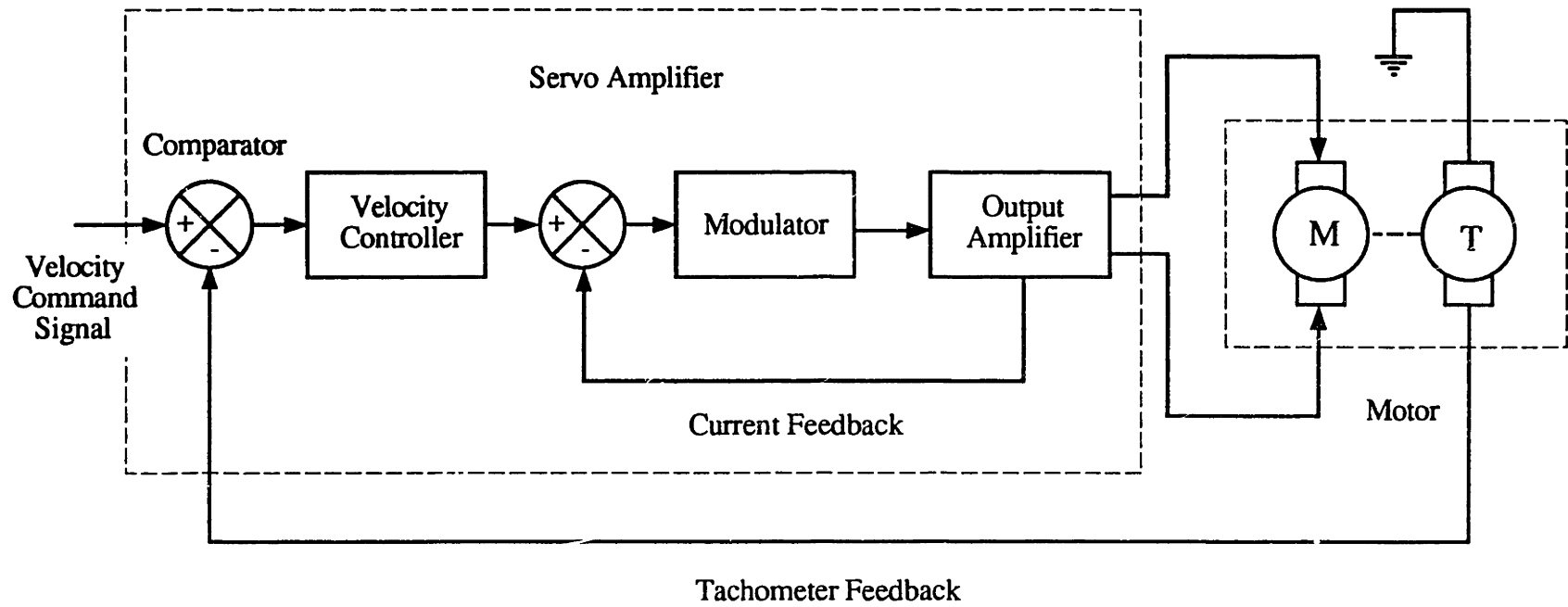


Figure 2.21: Simplified Block Diagram of the DC Servo Motor Controller.

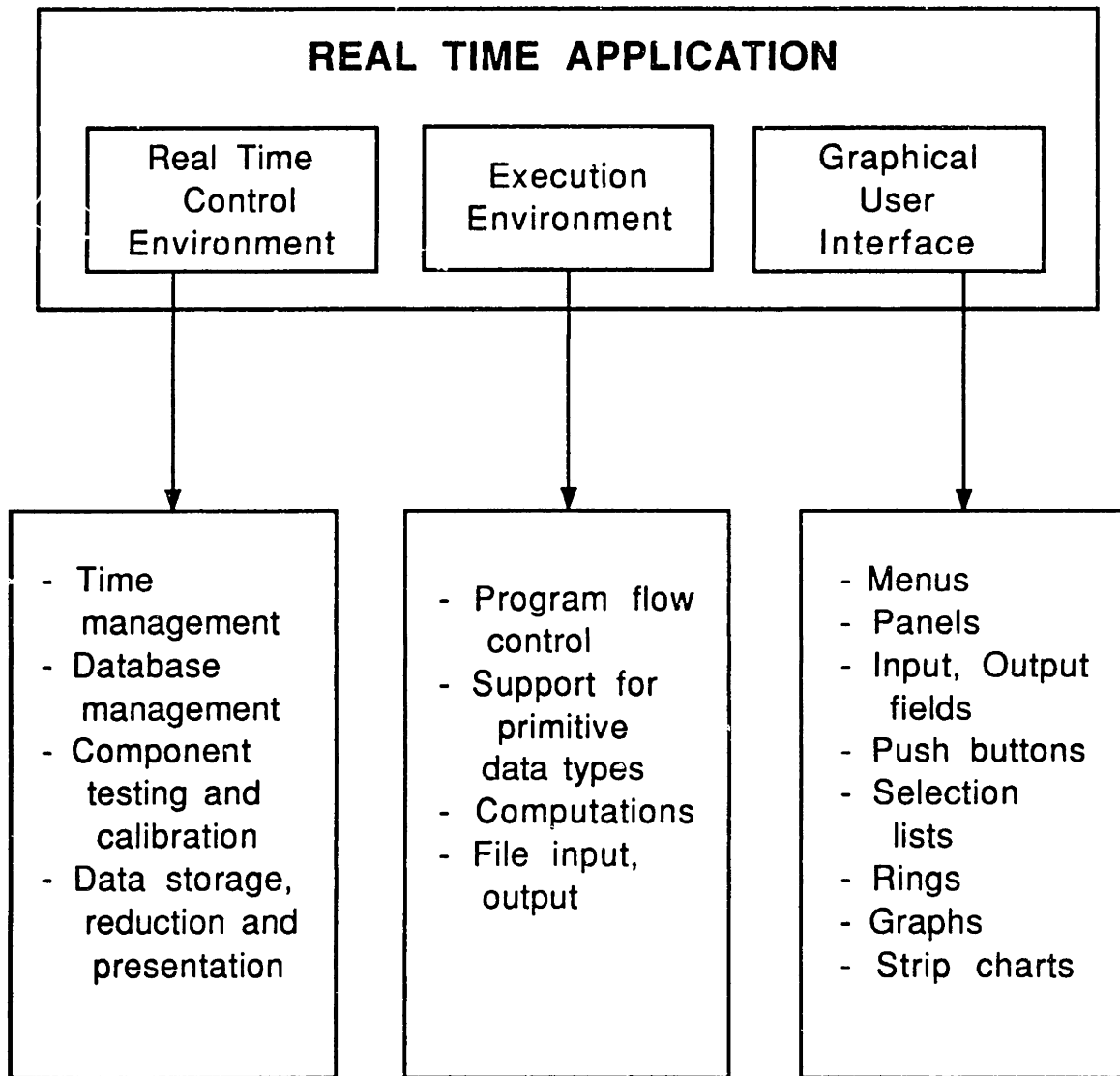


Figure 2.22: Structure of FlexCAT Program.



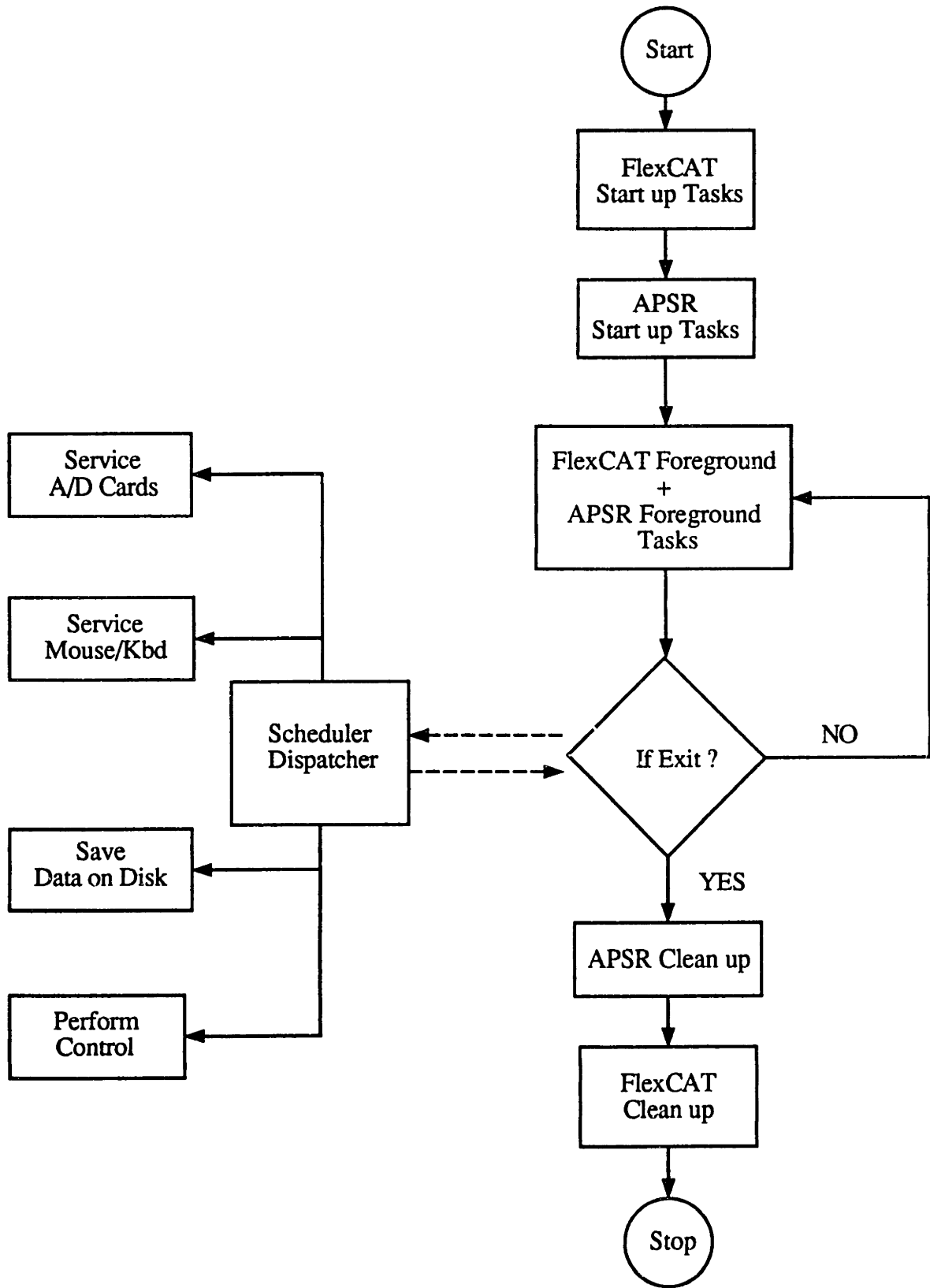


Figure 2.23: FlexCAT Flow Control Diagram.



System Setup Calibrate Test DataAcq Functions Graph

### Active Plane Strain Soil Reinforcement

DCDT 1	DCDT 2	DCDT 3	DCDT 4	Axial Strain %	Test Mode
0.000	0.000	0.000	0.000	0.0000	
Axial Stress kPa		Strain Rate mm/min.		Limits	
Plat. 1	0.000	Target	Actual	Disp.	Stress
Plat. 2	0.000	0.0000	0.0000	0.00	0.00

Sidewall Pressure kPa

Top: 0.000

Bottom: 0.000

Plane Strain %: 0.0000

b Value: 0.000

Run

Reinf. Position mm: 0.0000

Load kN/m: 0.000

Run

Air Pressure kPa

Target: 0.00

Actual: 0.00

Lateral Strain %: 0.0000

R Value: 0.000

Run

Graph Selection

Axial Strain-time

Figure 2.24: The APSR User Interface.



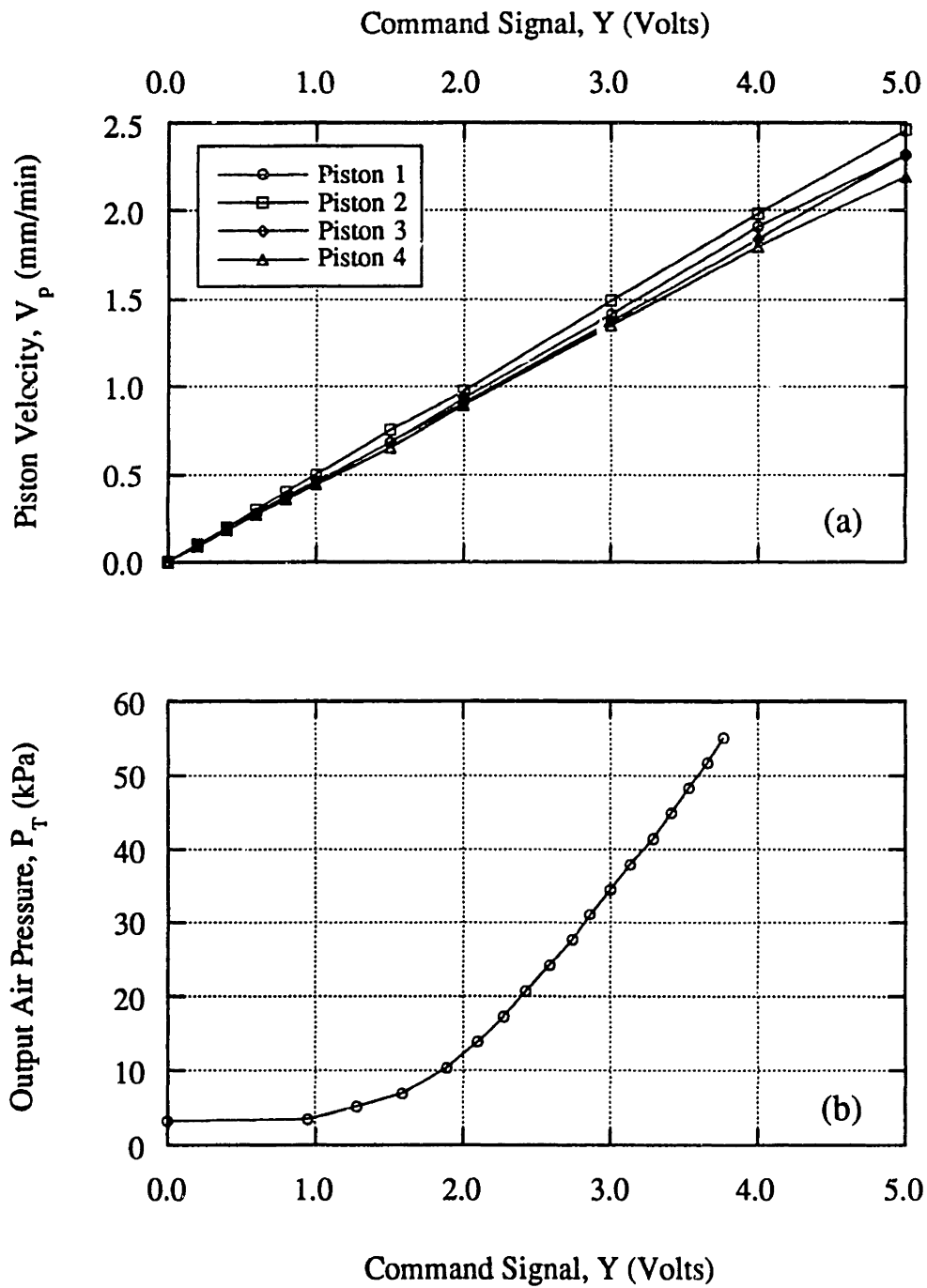


Figure 2.25: Calibration Curves for Platform Pistons and Air Pressure Regulator.

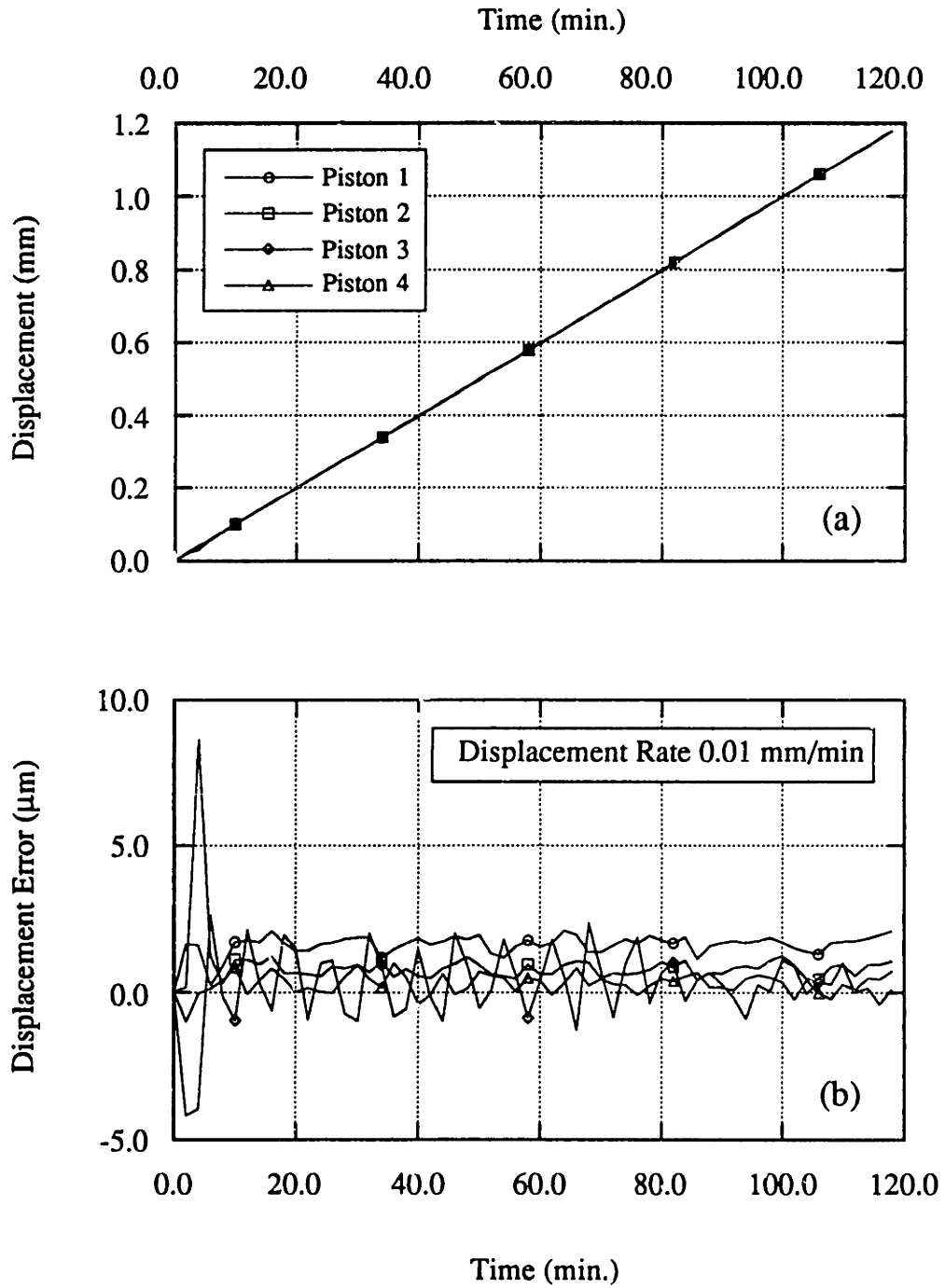


Figure 2.26: Performance of Platform Piston Control System.

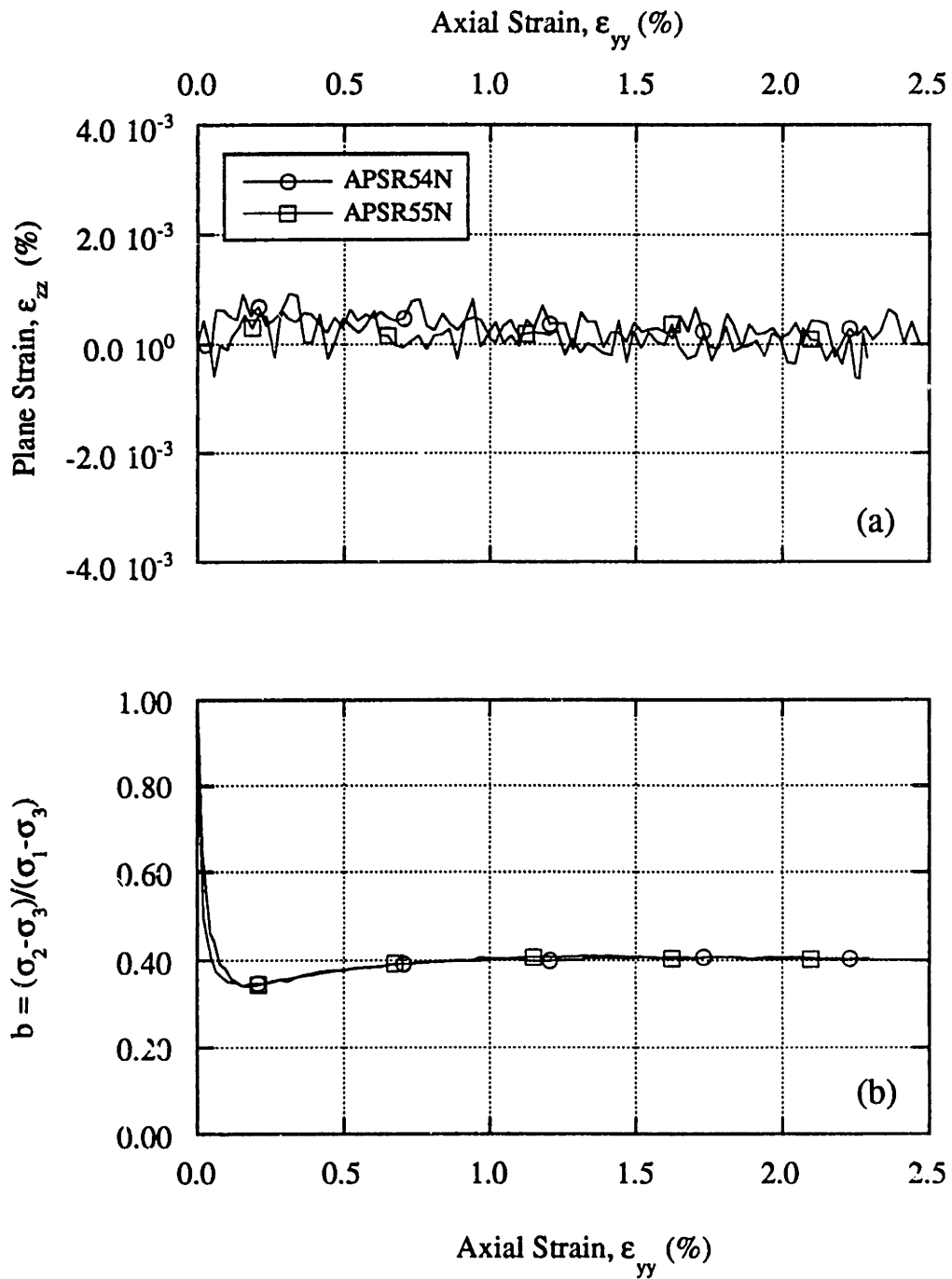


Figure 2.27: Performance of Plane Strain Sidewalls.

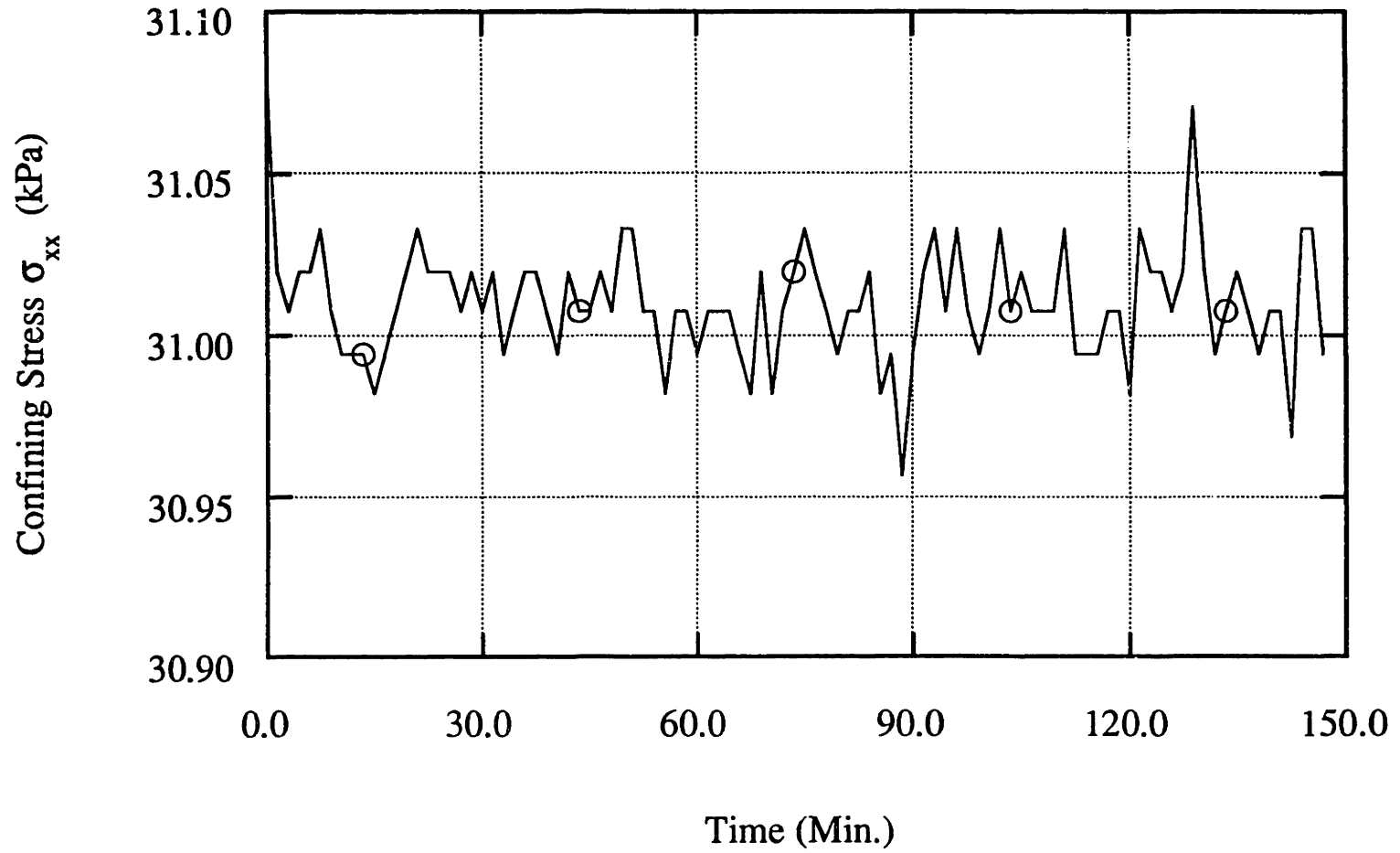
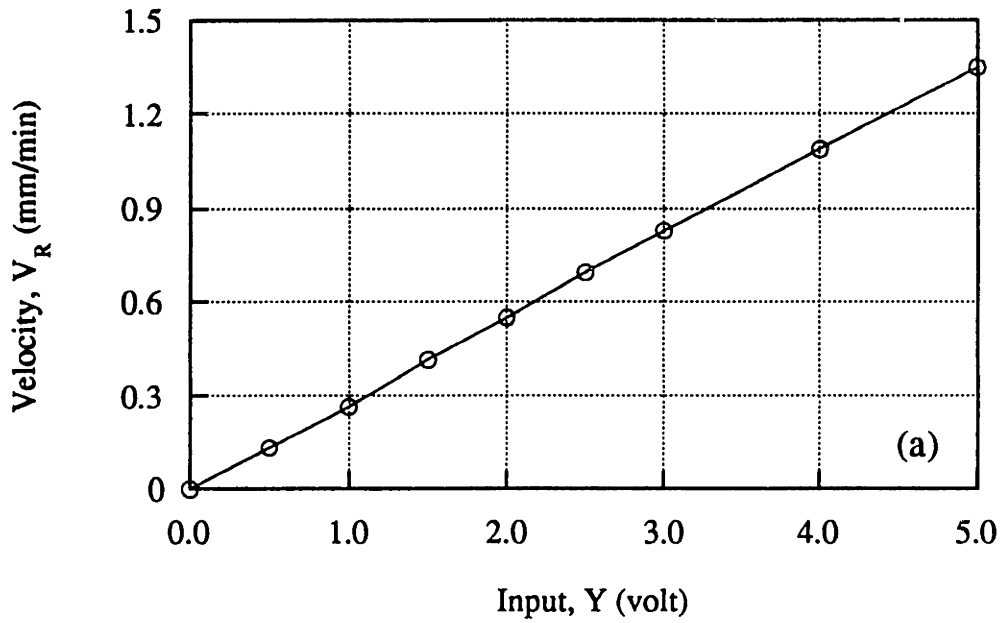
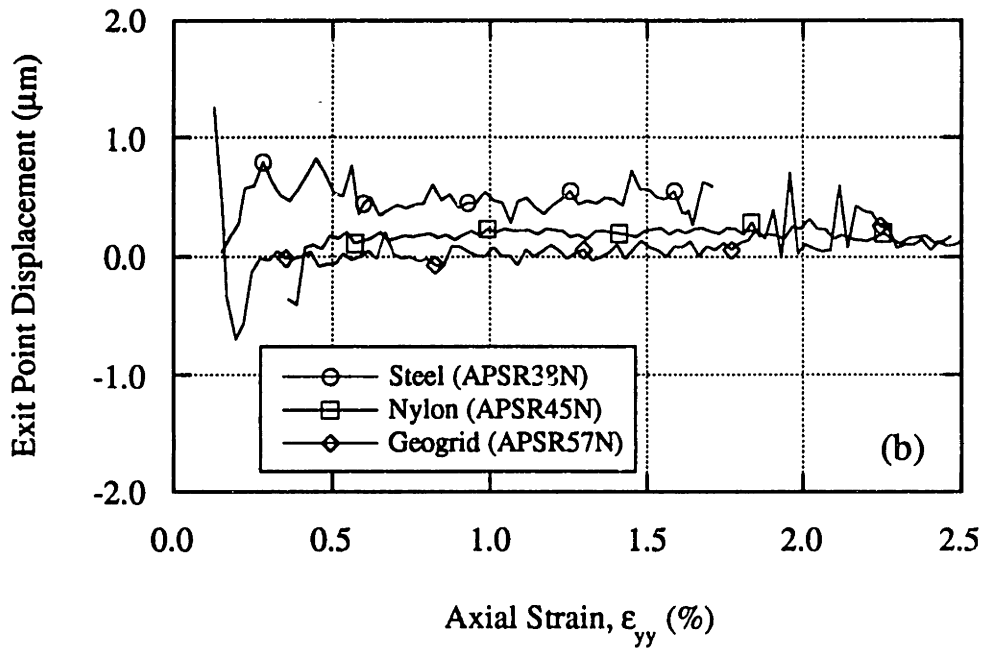


Figure 2.28: Performance of Air Pressure Control System.





(a) Calibration of Reinforcement Control System.



(b) Performance of Reinforcement Control System.

Figure 2.29: Calibration and Performance of Reinforcement Control System.



# Chapter 3: Test Materials and Procedures

## 3.1. Introduction

This chapter describes the materials and testing procedures used in the APSR research. The chapter is divided into three sections. The next section presents engineering properties of the matrix (soil) and reinforcing materials used in the present test program. Section 3.3 outlines the important steps involved in preparing a test specimen and conducting the APSR test. The final section (Section 3.4) presents results showing the shear behavior of unreinforced Ticino sand in the APSR cell. Where appropriate, a comparison has been made with the previously reported work (Larson, 1992).

## 3.2. Test Materials

The APSR cell has been designed with the aim of performing tests on a wide variety of matrix and reinforcing materials. Although it is possible to use either granular or cohesive soils in the cell, the present work employs only one variety of dry, medium sand. The material tested in the APSR cell is Ticino sand, a typical, multi-colored, clean river sand imported from Italy. Ticino sand was chosen because its engineering properties are typical of many natural sands and have been extensively documented in the literature (Baldi et al., 1985; Franco, 1989). At the present time, the

most commonly used reinforcements are made from steel or polymers (Koerner, 1993; Van Santvoort, 1994). Accordingly, three different types of reinforcing inclusions have been used in this study: a) mild steel sheet, b) Nylon 6/6 sheet, and c) woven polyester geogrid sold under the trade name Fortrac® (Huesker, 1994).

### 3.2.1. Ticino Sand

Ticino sand is a uniform (poorly graded) sand with medium grained (less than 0.3% fines), sub-rounded particles with specific gravity,  $G_s = 2.67$ . It has a uniformity coefficient,  $C_u = 1.5$ , and mean particle size,  $D_{50} = 0.5$  mm as obtained from the sieve analysis results shown in Figure 3.1. Physical properties of Ticino sand are summarized in Table 3.1 adopted from Larson (1992).

Larson (1992) evaluated the specimen uniformity by measuring formation densities locally at several locations in the APSR cell. He concluded that the air pluviation method (see Section 3.3.1.4) produces medium dense sand specimen with average density,  $\gamma = 1.63$  g/cm<sup>3</sup> and standard deviation of 0.01 g/cm<sup>3</sup>. This corresponds to a relative density,  $D_r = 78\%$ , according to the maximum and minimum densities reported in Table 3.1. Table 3.2 shows the measurements of global (overall) specimen densities for nine dense APSR specimens. The densities range from 1.61 to 1.63 with mean density  $\gamma = 1.62$  g/cm<sup>3</sup> ( $D_r = 74.87\%$ ) and a standard deviation of 0.01 g/cm<sup>3</sup>. The last three measurements in Table 3.2 were part of the present study and are well within the range defined by the first six measurements reported by Larson (1992). All tests in the present test program were performed on dense sand specimens. The plane strain shear behavior of Ticino sand in the APSR cell is discussed in Section 3.4.

### 3.2.2. Reinforcing Materials

The APSR cell has been designed to accommodate a wide range of potential reinforcing materials including polymer strips, geogrids, and woven and non-woven geotextiles. The interpretation of geotextile and geogrid behavior from a laboratory test is often complicated by their non-linear, time-dependent stress-strain properties, as well as their complex geometry and surface texture. Since the first goal of the experimental program (Chapter 4) was to refine the APSR cell design and to evaluate the proposed shear-lag analysis (Abramanto, 1993), two types of linearly elastic, planar (sheet) materials were used. Steel and Nylon 6/6 reinforcements were selected to represent the upper and lower limits of axial stiffness of the materials used in practice. Nylon 6/6 has an axial stiffness which is approximately two orders-of-magnitude lower than that of steel (Table 3.4). The second part of the test program (Chapter 5) used two grades of Fortrac® geogrids manufactured by Huesker Corporation. Fortrac geogrid was selected in the APSR cell primarily because: a) it has a relatively small size aperture (23 mm) making it possible to accommodate sufficient number of grid 'cells' in the transverse direction needed to approximate the behavior of an infinite sheet, (McGown et al., 1985), b) the polyester (PET) fibers used in making Fortrac grids undergo virtually no creep (time-dependent) deformations, and c) the grids are available in a wide variety of geometry, axial stiffness, and strength.

#### 3.2.2.1. Steel Sheet

Steel sheet was chosen because: a) it exhibits linear elastic behavior over the stress range of interest in APSR tests, b) it is relatively easy to bond strain gages to steel in order to measure local strains in the inclusion, and c) a relatively simple grip can be used to hold the reinforcement (Larson, 1992). The inclusions were cut from a 152 mm wide roll of 0.127 mm thick shim stock which had a thickness tolerance of 10%.

The APSR specimens consist of two sheets bonded together using a strong but flexible adhesive. Some of the steel reinforcement specimens were instrumented with foil type strain gages mounted on the inside face of one of the sheets. The sandwich construction was used because: a) it protects the strain gages from abrasion by the surrounding soil and b) mounting the sheets back-to-back counters the curvature of the shim stock and produces a flat reinforcement. Although, the use of strain gages was discontinued in the later part of the test program, the two ply design was maintained. Table 3.4 lists the dimensions and mechanical properties of the steel sheet reinforcement. The coefficient of friction for smooth steel sheet and dry sand is approximately 0.4 which corresponds to an interface friction angle  $\delta = 21.8^\circ$  (Abramanto, 1993).

#### 3.2.2.2. Nylon 6/6

Nylon 6/6 belongs to a class of polymers known as aliphatic polyamides. The chemical structure of Nylon 6/6 (Figure 3.2a) contains polar -CONH- groups spaced at regular intervals giving rise to strong interchain attraction. High intermolecular attraction accounts for the high melting point ( $T_m > 200^\circ \text{C}$ ) and excellent creep resistance of Nylon as shown in Figure 3.2b. The glass transition temperature ( $T_g$ ) of Nylon 6/6 ranges between  $50^\circ$  to  $90^\circ \text{C}$  (Chanda and Roy, 1992). Therefore, at room temperatures, Nylon 6/6 is a glassy, semicrystalline polymer. The elastic moduli of most common polymers in their bulk (unoriented) state fall within a narrow range as shown in Table 3.3 (Branrup and Immergut, 1975; Hall, 1981; Chanda and Roy, 1992). Nylon 6/6 was selected because over the strain and time ranges of interest in the APSR cell, it exhibits: a) nearly linear elastic behavior and b) very little creep (or stress relaxation). Figure 3.3 presents typical results from uniaxial tension tests performed at

three different strain rates.<sup>1</sup> It can be seen that a 100 fold change in strain rate has negligible effect which indicates that stress-strain behavior is time independent within the time frame of these tests. A linear regression analysis of the initial loading curve for the range of axial strain expected in the APSR test ( $\epsilon_a < 2\%$ ) yields secant elastic modulus,  $E_f = 1.92$  GPa.

The APSR reinforcement specimens were cut from a 152 mm wide roll of 0.255 mm thick Nylon strip with a thickness tolerance of about 7%. The inclusion consisted of two sheets bonded together using a flexible rubber adhesive. Strain gages were used to measure internal strain on a few Nylon reinforcement specimens although, in general, the strain gage results were found to be unsatisfactory. Table 3.4 lists the dimensions and mechanical properties of the Nylon 6/6 reinforcement. Abramson (1993) established coefficient of interface friction between Nylon 6/6 and dry Ticino sand in the ranges between 0.15 and 0.4 based on data published by O'Rourke, et al. (1990) for similar materials. For a smooth, relatively undamaged Nylon sheet, the lower value is more representative.

### 3.2.2.3. Fortrac Geogrid

Geogrids are net shaped synthetic fabrics that are used extensively in the construction of reinforced earth walls and embankments. Fortrac<sup>®</sup> geogrids are manufactured from high-tenacity polyethylene terephthalate (PET, also known as Polyester) yarns woven in an interlocking pattern (Figure 3.9). A special weaving process ensures that the fibers in the longitudinal direction are kept more or less straight

---

<sup>1</sup> The tests were conducted on small strip specimens 101.6 mm long, 19.05 mm wide, and 0.254 mm thick, cut from the same stock used for making the APSR inclusions.

(crimp free), thus producing a grid whose axial stiffness is comparable to that of the raw yarn (Huesker, 1994). The grid is coated with a thin layer of black polyvinyl chloride (PVC) for ultraviolet protection. Fortrac geogrids are available in a variety of grades with different stiffness and strength. Figure 3.4a presents axial stress-strain curves supplied by the manufacturer for five different grades (Huesker, 1994).<sup>2</sup> The first two numbers in the grid designation (i.e., Fortrac 20/13) indicate the ultimate strengths in kN/m, measured in the longitudinal and lateral directions respectively. Fortrac grids exhibit an S-shaped stress-strain response which is characteristic of many polymer multifilament structures (Van Santvoort, 1994). Exceptionally high creep resistance of PET fibers used in Fortrac grids is evident from Figure 3.5 which shows that even when loaded to 50% of its ultimate strength, Fortrac 80/30-30 geogrid undergoes virtually no creep within the duration of the test. Fortrac 80/30-20 and Fortrac 110/30-20 grades were used in the present test program because their small strain ( $\epsilon_a < 1\%$ ) stiffness is comparable to that of Nylon sheet reinforcement described earlier. The small strain behavior of Fortrac 110/30-20 grid is shown in Figure 3.4b which is based on data supplied by the manufacture. Since special on-specimen strain measuring devices and wide, frictional grips needed for performing representative uniaxial tensile tests on geogrids were not readily available, the small strain secant stiffness of Fortrac grids was estimated from the data shown in Figure 3.4b. Results of APSR tests using Nylon 6/6 reinforcements (Chapter 4) indicate that the maximum tensile strain in a relatively extensible inclusion is of the order of 0.5 to 1.0%. Figure 3.4b shows that at axial strain,  $\epsilon_a \approx 0.8\%$ , the secant stiffness of Fortrac 110/30-20 geogrids,  $J \approx 2000$  kN/m. Table 3.5 summarizes the physical and mechanical

---

<sup>2</sup> Data in Figure 3.4 have been obtained from ASTM wide width tensile test (ASTM D-4595).



properties of Fortrac 80/30-20 and Fortrac 110/30-20 geogrids. The fraction of the total grid cross sectional area which is solid is termed the 'solids ratio',  $\alpha_s$ . It should be noted this ratio is slightly higher for the small grid specimens used in the APSR cell than for a field scale application where very large grids are used.<sup>3</sup> This has implications on the interpretation of measured load transfer in the cell as discussed in Chapter 5.

### 3.3. Sample Preparation and Test Procedure

#### 3.3.1. Sample Preparation

For the purpose of this description, the sample preparation process has been divided into four discrete steps. In reality, some of the activities within these steps are performed concurrently. The first step involves the preparation of a thin, air-tight, rubber membrane (rubber bag) to enclose the specimen. The second step involves preparation of reinforcement and its attachment to the rubber membrane. Next, the APSR cell is cleaned and greased and the membrane-reinforcement assembly is placed in the cell. Finally, the sand is deposited by pluviation (raining) method and the rubber membrane is sealed. The entire process typically takes about two days from start to finish.

---

<sup>3</sup> The ASPR grid specimen has six longitudinal members and five cells as shown in Figure 3.9.

### 3.3.1.1. Rubber Membrane

The rubber membrane contains the sand during deposition and supports the vacuum necessary to hold the specimen before application of boundary stresses. The membrane is custom-made from a standard roll of 0.38 mm thick natural rubber sheet sold by the Green Rubber Company. Each membrane is made by gluing together three separate pieces shown in Figure 3.6a. The layout pattern shown in Figure 3.6b provides the maximum yield for a given length of the roll. A thin, even coat of a latex adhesive (P7865 contact adhesive from American Finish and Chemical Co.) is applied to the shaded areas in Figure 3.6a using a spatula and allowed to dry. The cured adhesive is then covered with plane paper strips<sup>4</sup> to prevent accidental contacts. In order to assemble the membrane, the pieces are laid on a flat surface and paper strips are removed only from the surfaces which need to be bonded together. When the membrane is fully assembled, a narrow bead of RTV (a silicon rubber adhesive made by General Electric) is applied to the inside of all four corners to create air-tight joints. The paper strips along the side and edge flaps (all 1.5 inch wide shaded surfaces in Figure 3.6a) are left in place until after the sand pluviation process (see Section 3.3.1.4.).

### 3.3.1.2. Reinforcement

Careful preparation of the reinforcement is the key to successful load measurements in the APSR cell. Larson (1992) describes the full details of the procedure used for preparing the steel reinforcement including how to mount strain gages and perform calibration tests. Since most of the planar inclusions in the present

---

<sup>4</sup> Cash register tapes are very suitable for this purpose.

investigation did not use strain gages, full scale in-isolation calibration tests were generally not performed. Instead, material properties were either obtained from the tests on small strips or based on the average values reported in the literature. Figure 3.7a shows a schematic of the longest planar reinforcement used in these experiments (360 mm long). The steps required to prepare and install the planar inclusions can be summarized as follows:

1. The inside (concave) surface of both sheets is cleaned with acetone to remove surface contamination. In the case of steel, fine sand paper is used to remove any patches of rust.
2. The steel and Nylon 6/6 inclusions are made by bonding two sheets of material using a flexible rubber vinyl adhesive (Number 80 from 3M Inc.) which does not crack or alter the stiffness of the reinforcement. The contact adhesive is sprayed on both sheets and allowed to dry. The adhesive bonds instantly when the two surfaces are brought together and pressed against each other.
3. The free end of the reinforcement is sandwiched between two wooden blocks and the hole pattern (Figure 3.7b) for the grips is drilled using a drill press. The center line and exit boundary are marked using a suitable marker.
4. For the steel inclusion, a circular window is made for the infrared light beam by drilling a 3.0 mm diameter hole about 20.0 mm away from the exit boundary (Figure 3.7b). Similarly, a small rectangular IR detector window is cut in the Nylon specimen using a sharp Octo knife.
5. Exposed (completely dark) X-Ray film is used as indicator strip (Figure 3.7b). One end of the strip is attached to the reinforcement using a small dab of Permabond 910 industrial Cyanoacrylate adhesive. Care should be

taken to make sure that the free end of the strip is perpendicular to the centerline of the reinforcement.

6. A rectangular piece of transparent Scotch tape holds the indicator strip close to the reinforcement. In order to allow the indicator strip to slide freely beneath the Scotch tape, a second piece of tape covers the exposed surface which is in contact with the indicator strip.

Once the reinforcement is ready, it is attached to the specimen rubber membrane. A slit of required length is cut along one side of the rubber membrane such that the final position of the inclusion is equidistant from the plane strain faces of the APSR cell (Figure 3.8a). The air-tight connection between the reinforcement and specimen membrane is achieved in two steps.

1. A rubber collar, made of 0.25 mm thick rubber sheet, is fitted around the inclusion as shown in Figure 3.8c. The seam between the collar and inclusion is sealed using a bead of RTV spread evenly across the joint. After about 30 minutes (by which time the bead is partially cured), the process is repeated on the other side. The final assembly is made easier if the angle between the rubber collar and the reinforcement is greater than  $45^\circ$  as shown in Figure 3.8d. The partially cured joint is sufficiently flexible that the angle can be changed by gently stretching the rubber collar.
2. Once the RTV is completely cured (after about 24 hours), the collar is bonded to the specimen membrane using latex adhesive. This is done by fitting the membrane over a slotted wooden frame. A thin coat of latex adhesive is applied to the rubber collar and to the corresponding area around the inclusion opening in the specimen membrane. The two surfaces are then bonded on contact after the adhesive dries.

Fortrac geogrid samples were cut from large rolls supplied by the manufacturer. A steel flap cut from 0.05 mm thick shim stock acts as the indicator strip as shown in Figure 3.9. The flap is wrapped around a transverse rib element and fixed to it using Permabond glue. The far edge of the steel flap, which is used by the infra-red sensor as target, should be perpendicular to the longitudinal axis of the reinforcement. Since the grids have large openings, it is very difficult to create the air-tight connection between the inclusion and the specimen membrane at the exit boundary. Figure 3.9 shows the arrangement used to connect the grid to the rubber membrane. Two 17 mm wide strips are cut from 0.254 mm thick Nylon 6/6 sheet. The strips are attached to each side of a geogrid rib using silicone rubber adhesive RTV. The space between the Nylon strips is completely filled with RTV to block any air passages. However, care is taken not to apply any RTV between the indicator flap and the Nylon strips. The Nylon strips now provide smooth continuous surfaces to which the rubber collar can be attached using the procedure described for planar inclusions.

### 3.3.1.3. Cell Preparations

The inside surfaces of the APSR cell must be cleaned to remove any dirt and sand before setting up a new test. Rubber liner sheets are removed and old grease is scrapped off using a razor blade. All surfaces of the cell directly in contact with the rubber membrane containing the sand specimen are lubricated with 50-50 mixture of Dow Corning high vacuum grease and Compound 7. The Compound 7, also manufactured by Dow Corning, is a release agent which reduces the viscosity and shear strength of the vacuum grease. A measured amount of grease mixture (about 0.01 gm/cm<sup>2</sup>) is applied to all contact areas and spread evenly using fingers. The old grease may be reused if it is free of sand particles and other impurities. Rubber sheets are

placed on greased surfaces and covered with another layer of grease. A paint roller is used to squeeze out any air bubbles trapped beneath the sheets.

The specimen mold is inserted into the cell and attached to the rim of the cell with seven 1/4" bolts. The membrane is spread evenly inside the mold to remove wrinkles and to squeeze out any entrapped air. A light layer of grease inside the mold causes the membrane to stick to the surface of the mold, which facilitates setting up the membrane. The top edges of the membrane are firmly attached to the mold using masking tape to prevent sand particles from falling between the mold and membrane. The membrane is pierced with a sharp nail through the vacuum port and the vacuum tube is inserted through the membrane. The rubber membrane closes itself around the vacuum tube so that the specimen is able to hold the vacuum reliably. A filter screen (within the vacuum tube fitting) prevents sand particles from being sucked into the tube when the vacuum is applied. For steel and Nylon reinforcements, the spacing between the reinforcement and the cell floor is maintained by a small piece of rubber block. The particles bouncing off the rim of the cell may drop into the cell at uncontrolled velocity resulting in non-uniform densities near the walls of the cell. A sheet metal deflector shield is, therefore, placed on the top of the specimen mold to prevent sand from depositing on the rim (Figure 3.10a). The inside dimensions of the deflector shield match the final specimen length and width. A reinforcement holder shown in Figure 3.10b is used to maintain longitudinal alignment of flexible (Nylon and grid) inclusions during sand pluviation. It consists of two aluminum sheets which are taped together on three sides. The reinforcement is gripped between the two sheets which exert enough lateral force to keep the former in proper alignment.

#### 3.3.1.4. Sand Deposition

Specimens in the APSR cell are prepared by means of multiple sieve dry pluviation (raining), a method in which sand is dropped from a controlled height through a set of sieves configured to achieve a desired density. For poorly graded sands, pluviation creates homogeneous specimens with controlled relative densities, and simulates a soil fabric close to natural, sedimentary deposits (Larson, 1992; Rad and Tumay, 1987; Oda et al., 1978). It is possible to achieve excellent control of formation density (Table 3.2) using a properly designed pluviation apparatus. Figure 3.11 shows the raining apparatus used for preparing dense APSR specimens with average relative density  $D_r = 75\%$ . It consists of a sand hopper attached to the top of a 1.4 m high chimney. The hopper has a perforated floor with 18.3 mm diameter holes arranged in a 65 mm rectangular pitch (about 6.2% total opening area). A sliding trap door has a similar hole pattern but the holes in the trap door are initially staggered with respect to the holes in the hopper floor. By sliding the trap door, the holes can be aligned, allowing the sand to fall freely. Two wire mesh screens (diffusers) with 6.3 mm openings fitted inside the chimney scatter the sand particles to create a uniform deposit. The two sieve patterns are rotated  $45^\circ$  with respect to each other and arranged at a distance of 200 mm apart. The lowest sieve is about 430 mm above the floor of the APSR cell. Loose specimens with average relative density,  $D_r = 31\%$  can be prepared by removing the diffuser screens from the chimney.

The total weight of the sand to be rained is about 20% greater than the final weight of the APSR specimen to allow for uneven top surface and spillage. The sand is normally deposited in two layers using the procedure given below.

1. Approximately half of the sand is poured into the hopper and spread uniformly. The chimney is lifted using an overhead crane and placed on top of the specimen mold.
2. The trap door is opened causing the sand to pour through the perforated hopper floor. When pluviation is complete, the chimney is carefully lifted off the specimen mold, and the reinforcement restrainer and deflector shield are removed. At this stage, if radiography is to be used, tungsten markers are laid out on the sand surface in a grid pattern. The sand spilled around the mold rim is vacuumed out and the deflector shield is placed back.
3. The second half of the specimen is rained using the same procedure except this time, the reinforcement restrainer is not used. The top surface of the rained specimen is about 2-3 cm higher than the final specimen height.
4. The excess sand is removed using a screeder to produce a smooth top surface. The screeder comprises a 3.2 mm thick aluminum plate measuring 445 x 125 mm, mounted on an aluminum angle with two screws. The angle slides along the top of the specimen mold such that the bottom edge of the plate scrapes the sand off at a controlled height. The sand pushed from the top of the specimen is removed using a scoop.
5. All adhesive covered surfaces are exposed by carefully removing the paper tapes. The lateral joint (running along the width of the specimen) is made first by folding the two side flaps over the top of the specimen. Next, dabs of latex adhesive are applied to the areas near corners to reduce chances of leakage and edge (longitudinal) flaps are sealed to the side flaps.
6. Vacuum is applied to the specimen using a centrifugal vacuum pump. The specimen membrane rarely seals near the corners at the first attempt and RTV must be applied to corner seams. Small leaks in the top seams are



easily located by the hissing sound they produce and can be readily sealed with RTV. A vacuum of about 60 kPa is considered satisfactory, although it is possible to remove the specimen mold when the vacuum is as low as 25 kPa.

### 3.3.2. Test Procedure

In the 'standard' APSR test, the soil specimen is sheared in the horizontal plane by increasing the major principal stress,  $\sigma_1$ , at a fixed confining stress,  $\sigma_3$ , while tensile force is applied to the reinforcement to prevent displacement at the exit point relative to a fixed external frame of reference. The test is performed in two steps: 1) the soil specimen is initially consolidated to a prescribed stress state ( $\sigma_{1c}$ ,  $\sigma_{3c}$ ) and 2) the specimen is sheared in a compression mode by increasing the axial stress,  $\sigma_1$ . The following paragraphs give more details of the test procedure.

#### 3.3.2.1. Application of Initial Boundary Stresses

In order to avoid collapse, compressive boundary stresses must be applied to the APSR test specimen before removing the vacuum. In addition, a small shear stress (seating load) is imposed to ensure a good contact between the specimen and loading surfaces in the major principal stress direction (y direction, Figure 2.2). The procedure comprises the following steps:

1. A small tensile seating load (about 25 N typically) is applied to the reinforcement by manipulating the actuator arm which controls the reinforcement position (Figure 2.11). This ensures a good contact between the reinforcement grip and the C-clamp located at the end of the arm.
2. The air gap between the specimen and loading platforms is closed by displacing the platforms inwards from both ends at a rate of 0.5 mm/min.

Once the contact is established between the platforms and the specimen, the displacement rate is reduced to about 0.1 mm/min. The platforms are displaced until the major principal stress,  $\sigma_1$  is about 7 kPa.

3. Water is then injected into the top and bottom side walls until the intermediate principal stress,  $\sigma_2 \approx 7.0$  kPa, and the confining air pressure  $\sigma_3$  is increased to 7.0 kPa to provide initial hydrostatic confinement for the specimen.
4. Confining pressures in all three axes are increased in increments of 7.0 kPa by repeating the above process until  $\sigma_1 = \sigma_2 = \sigma_3 = 31.0$  kPa.<sup>5</sup> The inclusion load and exit point position are monitored during this process. Generally, increasing  $\sigma_3$  pushes the specimen against the rear wall of the cell causing the inclusion to move outwards resulting in a slight drop in the reinforcement seating load applied in step 1.
5. The major and intermediate principal stresses are further increased to 38.0 kPa so that the applied stress ratio,  $\sigma_1/\sigma_3 = R \approx 1.22$  and  $b = (\sigma_2 - \sigma_3)/(\sigma_1 - \sigma_3) \approx 1$ .
6. The vacuum is then released and the specimen is allowed to equilibrate for about 30 minutes. This process causes some change in the boundary stresses ( $\sigma_1, \sigma_2$ ) and also in the reinforcement load,  $F_R$ . After equilibrium, these parameters are restored to their previous values, the infrared sensor for the reinforcement is adjusted to its mid-span value (refer to Section 2.2.6) and the specimen is ready for shearing.

---

<sup>5</sup> The final increment is 10.0 kPa.

### 3.3.2.2. Plane Strain Shearing

The major principal stress is applied through the loading platforms which are displaced at a rate of 0.05 mm/min. At this speed, an average test lasts for about 100 minutes, and corresponds to an average stress rate of 2.8 kPa/min. The top and bottom sidewalls are held in position by feedback control in order to achieve plane strain conditions throughout the shear phase. During the first phase of shearing, the reinforcing inclusion is allowed to move freely and the displacements are monitored at the exit point. These data show that the inclusion is forced into compression<sup>6</sup> until the applied stress ratio remains below a characteristic value,  $R \leq R^*$  where  $R^* \approx 4.5$  (see Section 4.4). The inclusion compression cannot be controlled in the APSR cell, and is an undesirable behavior for a thin, flexible, tensile reinforcement. Active control of the reinforcement is only established when the displacements monitored by the IR detector indicate the onset of tensile load transfer. The shear procedure in the APSR cell consists of the following steps.

1. A new file is created for reading all transducer channels at 90 seconds interval and the data acquisition task is started.
2. The plane strain control feedback loop is turned on.
3. Values of displacement rate, stress limit, and displacement limit are specified and the platform control feedback loop is started.
4. The reinforcement position is monitored continuously until displacements at the exit point indicate an inward movement, at which point the feedback control loop is activated.

---

<sup>6</sup> i.e., displacements measured at the exit point show the inclusion being pushed out of the APSR cell.

5. The test is carried out until either the specimen fails or the applied stress ratio,  $R = \sigma_1/\sigma_3 = 9$ .

### 3.4. Shear Behavior of Unreinforced Ticino Sand

As part of the initial proof testing program, Larson (1992) performed a series of tests on unreinforced Ticino sand and compared its shear behavior in the APSR cell with the data obtained from other laboratory shear devices. He also made comparisons between soil strains interpreted from boundary deformations and those obtained from radiographic measurements made locally within the soil specimen. In the current research program, a number of important changes have been made in the APSR cell configuration and test procedure in order to improve reliability and efficiency (refer to Appendix A for details). This section describes the effects these changes have on the externally measured shear behavior of unreinforced dense Ticino sand and interprets equivalent elastic material parameters (shear modulus,  $G_m$  and Poisson's ratio,  $\nu_m$ ) obtained from these data.

Figure 3.12 presents the shear behavior of three tests on dense Ticino sand ( $D_r = 75\%$ ) reported by Larson (1992). The data show consistent behavior up to an axial strain,  $\epsilon_{yy} = 1\%$  ( $R \approx 6$ ). In all three tests, the peak shear strength,  $R = 8.2$  to  $9.1$  (corresponding to  $\phi' = 52^\circ$  to  $53^\circ$ ) is mobilized at axial strains  $\epsilon_f = 1.7$  to  $1.9\%$ . The initial shear stress-strain response is influenced by the procedures used to bring the platforms into contact with the specimen surface (refer to Larson, 1992, and Section 3.4.1). As a result, the zero strain condition ( $\epsilon_{yy} = 0$ ) is defined in an ad hoc fashion based on the observed inflection point in the measured stress-strain behavior. The

stress-strain behavior appears quite linear for  $\epsilon_{yy} \leq 1\%$ , with an average secant modulus,  $E_{50} \approx 18.7$  MPa. Table 3.6 summarizes the shear data for points corresponding to zero dilation rate (point A) and peak shear resistance (point B) for all three tests.

### 3.4.1. Effect of Air Pressure Confinement

In the original design, the entire void space within the APSR cell was pressurized such that the platforms applied only the deviatoric stress component ( $\sigma_1 - \sigma_3$ ) to the soil specimen (see Figure 2.3). Therefore, the initial contact stress between the specimen boundary and the actuating surfaces was very low at the start of the shearing (e.g., for  $\sigma_3 = 31$  kPa,  $\sigma_1 = 38$  kPa, the actual contact stress between the specimen and loading platforms only 7 kPa). As a result, a large amount of deformation was required to level out the surface irregularities of the specimen during the shear phase causing the inflection point shown in Figure 3.12. One of the major refinements of the cell design (first used in test APSR 17N) was to introduce a thin rubber bladder (airbag) to apply the confining lateral pressure and thus, decouple the minor principal stress,  $\sigma_3$ , from the other principal stresses ( $\sigma_1$  and  $\sigma_2$ ). Figure 3.13 compares the shear behavior of dense, unreinforced Ticino sand with and without the airbag. The initial stress-strain response obtained from the test with an airbag (APSR 19N) is significantly stiffer and shows a continuous modulus degradation. Although the peak stress ratio is identical in both the tests, the strain at failure,  $\epsilon_f$  is about 25% lower for the test with the airbag. The post peak strain softening is inhibited in test APSR 19N, presumably due to the additional stability provided by the airbag at the front face of the specimen. The specimen with airbag shows slightly higher lateral deformations and dilation (Figures 3.13b, c).

Figures 3.14 and 3.15 summarize the externally measured stress-strain behavior of unreinforced Ticino sand in the modified APSR cell (for tests APSR 18N, 19N). The excellent repeatability and stability of the measurements in the refined APSR design is evident from the test data. Referring to Figure 3.14, the measurements show the following:

- a) The soil exhibits a non-linear stress-strain behavior throughout shearing with an average secant modulus,  $E_{50} \approx 18.78$  MPa.
- b) The average peak shear strength,  $R \approx 9.65$  (corresponding to friction angle,  $\phi' \approx 54.3^\circ$ ) is mobilized at axial strains  $\epsilon_f \approx 1.63\%$ .
- c) The specimen initially contracts during shearing. However, when the applied stress ratio,  $R \approx 7.1$  (point A in Figure 3.14), the rate of volumetric change ( $-d\epsilon_{v0}/d\epsilon_{yy}$ ) becomes zero and the sample dilates with continued shearing.

Table 3.7 gives a summary of the shear data obtained from the refined APSR cell for the points corresponding to zero dilation rate and peak shear resistance (points A and B respectively in Figure 3.14). A comparison with Table 3.6 shows that the values of most key shear parameters, such as the friction angle,  $\phi'$ , b parameter,  $\sqrt{J_2} / \sigma_{oct}$ , equivalent elastic modulus,  $E_{50}$ , and Poisson's ratio compare well with those reported previously for the prototype APSR apparatus (Larson, 1992). The high values of friction angle obtained in the APSR cell ( $\phi' = 52^\circ$  to  $54^\circ$ ) is due to the plane strain boundary condition and relatively low confining stress,  $\sigma_c = 31$  kPa, employed in the APSR test. Similarly, mostly compressive volumetric strains (Figure 3.14c) can be attributed to three factors: a) increased confinement in plane strain, b) the sand is medium dense ( $D_r = 75\%$ ), and c) the specimen is sheared in the isotropic plane. The values of the peak friction angle,  $\phi'$  and failure strain,  $\epsilon_a$ , observed in the APSR cell are consistent with the findings of plane strain tests on sands reported by other researchers

(Oda et al., 1978; Marachi et al., 1981; Deterling, 1984) as summarized by Larson (1992).

Figure 3.15 shows the performance of the active plane strain control system. The out-of-plane strain is kept within 0.001% which corresponds to less than 1  $\mu\text{m}$  displacement of the individual sidewall target points (Figures 2.6 and 2.7). The value of the parameter  $b = (\sigma_2 - \sigma_3) / (\sigma_1 - \sigma_3)$  is initially set close to  $b = 1$  ( $\sigma_2 = \sigma_1$ ) at the beginning of a test. During shearing,  $b$  decreases rapidly to a minimum of 0.25 at axial strain level,  $\epsilon_{yy} \approx 0.12\%$ , and reaches or exceeds 0.4 as the soil specimen approaches the maximum stress ratio  $R \approx 9$ . This same trend has been observed in almost every reinforced and unreinforced APSR test performed so far. Using a True Triaxial Apparatus (TTA) and Leighton Buzzard sand, Deterling (1984) showed that at a  $b = 0.4$ ,  $\epsilon_2 \rightarrow 0$  for both dense ( $D_r = 94\%$ ) and loose ( $D_r = 47\%$ ) specimens. Similar values of  $b = 0.35$  to 0.4 are reported by Wong (1985) for shearing (in the isotropic plane) of dense Leighton Buzzard sand ( $D_r = 90\%$ ,  $\sigma_c = 14$  kPa) using the Directional Shear Cell (DSC).

### 3.4.2. Effect of Rigid $\sigma_1$ Boundary

In the original APSR cell design, the major principal stress,  $\sigma_1$  was applied to the specimen via two flexible, water-filled, rubber bags (Larson, 1992). This design was intended to maintain uniform stress distribution along the major principal stress boundary. However, in practice, the flexible boundary leads to somewhat non-uniform soil deformations (also associated with friction along the top and bottom sidewalls) within the soil specimen. Internal strain measurements indicated that in the reinforced soil specimens, most incremental deformations at high values of stress ratio ( $R \geq 6$ ) occurred in a relatively small, unreinforced zone close to the specimen face. Beginning with test APSR 24N, the water bags were replaced with pieces of rigid plywood to

ensure relatively uniform displacement fields and to prevent premature failures within the soil. Figure 3.16 compares the shear behavior of unreinforced dense Ticino sand with flexible (water bag) and rigid (plywood) loading platforms. The stress-strain response is identical until  $R \geq 3$  since the specimen is still under influence of the vacuum pressure (see Section 3.4.3). However, for continued shearing, the specimen with a rigid boundary shows a significantly increased ductility with failure strain,  $\epsilon_f \approx 2.5\%$  while the failure shear stress is largely unaffected by the change in boundary condition. The lateral and volumetric strains are virtually identical for the two tests until  $\epsilon_{yy} = 1.5\%$ , at which point, failure is reached in the specimen with flexible boundary condition. The shear behavior of Ticino sand recorded in test APSR 24N have also been replicated in many subsequent reinforced tests with rigid  $\sigma_1$  boundary in which the reinforcement loads were low enough to have any significant impact on the global stress-strain response of the sand matrix (Section 4.4). Table 3.7 compares the shear data from APSR 24N (uniform displacement boundary condition) with results from APSR 18N and 19N (uniform traction boundary condition). The results show that apart from higher failure strain,  $\epsilon_f$ , the boundary conditions in the major principal stress direction have no significant effect on most key shear parameters.

### 3.4.3. Effect of Vacuum

Each APSR test specimen is subjected to an effective confining stress,  $\sigma_c = 60$  to 70 kPa due to the vacuum applied during the specimen preparation procedure (refer to Section 3.1.1). Referring to Figure 3.14a, the initial rapid increase in shear stress ratio,  $R$  followed by a mild break in the curve at  $R \approx 3$  is because of the pre-loading of the specimen. This effect was not discovered until later when two APSR specimens (with grid reinforcements) failed to reach the required vacuum during sample preparation. As illustrated in Figure 3.17, the specimen prepared using low ( $< 25$  kPa)



vacuum shows a stress-strain response markedly different from that of the specimen subjected to high (60 kPa) vacuum. This result implies that the standard APSR test procedures involve shearing of slightly overconsolidated Ticino sand with initial OCR  $\approx 2$  to 2.5.

### 3.4.4. Equivalent Elastic Soil Parameters

The shear-lag analysis (Abramento and Whittle, 1993) provides a framework for predicting and interpreting the inclusion loads in the APSR cell. The input parameters for shear-lag analysis (Chapter 4) include the linear elastic properties for describing the shear behavior of unreinforced sand ( $G_m, \nu_m$ ). The elastic approximations to the shear behavior of unreinforced sand in the APSR cell can be derived as follows. For plane strain compression of an elastic material, the slope of shear stress,  $q = (\sigma_1 - \sigma_3)/2$  versus axial strain,  $\epsilon_{yy}$  curve is related to the secant shear modulus,  $G_m$  as:

$$\frac{\Delta q}{\Delta \epsilon_{yy}} = \frac{G_m}{1 - \nu_m} \quad (3.1)$$

where,  $\nu_m$  is secant Poisson's ratio for plane strain compression.

The Poisson's ratio is computed from the slope of axial strain,  $\epsilon_{yy}$  versus lateral strain,  $\epsilon_{xx}$  plot as:

$$\frac{\Delta \epsilon_{xx}}{\Delta \epsilon_{yy}} = \frac{-\nu_m}{1 - \nu_m} \quad (3.2)$$

The earliest reliable measurements of unreinforced behavior of Ticino sand in the modified APSR cell were obtained in test APSR 24N (Figure 3.16). Figure 3.18 shows the variation of secant shear modulus,  $G_m$ , and Poisson's ratio,  $\nu_m$ , with axial strain,  $\epsilon_{yy}$ , for test APSR 24N. The initial high value of Shear modulus,  $G_m \approx 30$

MPa, decreases rapidly to  $G_m < 5$  MPa as the soil specimen is sheared. Figure 3.18b shows that the Poisson's ratio of the sand increases in a non-linear manner throughout shearing and approaches the upper limit for elastic materials,  $\nu_m = 0.5$ , at axial strain,  $\epsilon_{yy} \approx 2.2\%$ . Table 3.7 gives the values of equivalent elastic parameters for points corresponding to zero dilation rate (point A) and peak shear resistance (point B).

### 3.5. Conclusions

This chapter described the properties of soil and reinforcing materials tested in the APSR cell. It also described the testing procedures used in the APSR research and presented results from a set of APSR tests performed on unreinforced Ticino sand at relative density,  $D_r = 75\%$  and confining stress,  $\sigma_c = 31$  kPa.

A number of important changes have been made in the APSR cell design and testing procedures in order to improve reliability and efficiency of the test. The procedures established for specimen preparation, consolidation, and shear provide very consistent and repeatable results. The results show that although the boundary conditions imposed by the cell affect detailed shape of the stress-strain and volumetric response, values of most key shear parameters, such as the friction angle,  $\phi'$ , b parameter,  $\sqrt{J_2} / \sigma_{oct}$ , and Poisson's ratio,  $\nu_m$ , compare well with those reported by Larson (1992).

The results of the base-case APSR test on unreinforced Dense Ticino sand (Test APSR 24N) using the modified test apparatus show that:

1. The soil exhibits a non-linear shear stress-strain behavior throughout shearing with a secant shear modulus that varies from,  $G_m \approx 30$  MPa at the

beginning of the test to,  $G_m \approx 2.5$  MPa as the soil specimen approaches failure.

2. The peak shear strength,  $R = 9$  (corresponding to friction angle,  $\phi' \approx 53^\circ$ ) is mobilized at axial strain,  $\epsilon_f \approx 2.5\%$ .
3. The specimen contracts initially during shearing and the rate of volumetric change becomes zero when the applied stress ratio,  $R \approx 6$ . Thereafter, sample dilates with continued shearing but the net volumetric strains remain mostly compressive throughout the shearing.
4. The secant Poisson's ratio of unreinforced Ticino sand increases in a non-linear manner throughout shearing and approaches the upper limit for an elastic material,  $\nu_m = 0.5$ , at the applied stress ratio,  $R \approx 9$ .



Property	Symbol	Value	Source
USCS Designation		SP (< 0.3% fines)	Larson (1992)
Coefficient of Uniformity	$C_u$	1.5	Larson (1992)
Mean Particle Diameter	$D_{50}$	0.5 mm	Larson (1992)
Dry Densities:	$\gamma_{max}$ $\gamma_{min}$	1.70 g/cm <sup>3</sup> 1.38 g/cm <sup>3</sup>	Franco (1989)
Voids Ratios:	$e_{max}$ $e_{min}$	0.93 0.57	
Constant Volume Friction Angle*	$\phi'_{cv}$	32°	Franco (1989)

\* Measured in triaxial tests.

Table 3.1: Properties of Ticino Sand (after Larson, 1992).

Test Number	Specimen Weight (g)	Overall Specimen Density (g/cm <sup>3</sup> )	Relative Density $D_r$ , (%)
APSR 39†	62575	1.6127	72.74
APSR 41†	62852	1.6199	74.97
APSR 43†	62649	1.6147	73.33
APSR 45†	62683	1.6155	73.61
APSR 48†	63252	1.6302	78.19
APSR 49†	63327	1.6321	78.79
APSR 67	62943	1.6222	75.70
APSR 68	62734	1.6168	74.01
APSR 2N	62540	1.6118	72.45

† Test Performed by Larson, 1992.

Table 3.2: Relative Densities of Dense Ticino Sand Specimens.

Polymer Name	Abbreviation	Glass Transition Temp., $T_g$ ( $^{\circ}\text{C}$ )	Tensile Modulus (GPa)
Polyethylene Terephthalate (Polyester)	PET	60 to 70	1.2 to 1.8
Polypropylene	PP	-8 to -18	1.1 to 1.55
Polystyrene	PS	100	2.41
Polyvinyl chloride	PVC	82	2.42 to 3.45
Polycarbonate	PC	145	2.2 to 2.4
Nylon 6/6	PA66	50 to 90	1.8 to 2.83

Table 3.3: Glass Transition Temperatures and Elastic Moduli of Some Common Polymers (sources: Branrup and Immergut, 1975; Hall, 1981; Chanda and Roy, 1992).

Property	Steel	Nylon 6/6
Elastic Modulus, $E_f$ (kPa)	$2.07 \times 10^8$	$1.92 \times 10^6$
Poisson's Ratio, $\nu_f$	0.2	0.2
Thickness (mm)	$0.254 \pm 5\%$	$0.51 \pm 3.5\%$
Width (mm)	133.35	133.35
Axial Stiffness (kN/m)	52600	980
Interface Friction (with dry sand), $\tan\delta$	0.4 †	0.15-0.4 †

† After Abramento, 1993.

Table 3.4: Dimensions and Mechanical Properties of Steel and Nylon Sheet Reinforcements.

Property	Fortrac 80/30-20	Fortrac 110/30-20
Wide Width Ultimate Strength (kN/m)	80	110
Wide width Stiffness @ $\epsilon_a \approx 0.8\%$ (kN/m)	1400	2000
Aperture (mm)	22.8 x 22.8	22.8 x 22.8
Thickness (mm)		
at Rib	2.54	3.05
at Junction	3.30	3.81
APSR Specimen Width (mm)	120.6	120.6
Solids Ratio*, $\alpha_s$ for Infinitely Wide Grid (%)	20.30	20.30
Solids Ratio, $\alpha_s$ for APSR Specimen (%)	23.95	23.95

\* Defined as the ratio of the combined width of all longitudinal members to the total specimen width. Based on actual measurements.

Table 3.5: Dimensions and Mechanical Properties of Selected Fortrac Geogrids  
(Source Huesker, 1994).

Test	At zero dilation rate								
	q (kPa)	R	$\phi'$ °	b	$\frac{\sqrt{J_2}}{\sigma_{oct}}$	$\epsilon_a$ (%)	$\epsilon_{vol.}$ (%)	$G_m$ (MPa)	$\nu_m$
36	86	6.55	47.3	0.4*	0.71	1.1	0.34	4.30	0.44
38	96	7.23	49.2	0.4*	0.73	1.2	0.45	4.91	0.40
45	111	8.1	51.3	0.44	0.76	1.4	0.54	4.46	0.44
	At maximum shear stress								
36	112	8.2	51.5	0.47	0.77	1.7	0.28	3.45	0.48
38	125	9.1	53.3	0.44	0.78	1.7	0.36	3.95	0.46
45	121	8.8	52.7	0.44	0.77	1.9	0.50	3.43	0.46

Note: \* Values not available. Assumed for the purpose of other calculations.

Table 3.6: Shear Properties of Dense Ticino Sand in the Original APSR Cell  
(after Larson, 1992).

Test	At zero dilation rate								
	q (kPa)	R	$\phi'$ °	b	$\frac{\sqrt{J_2}}{\sigma_{oct}}$	$\epsilon_a$ (%)	$\epsilon_{vol.}$ (%)	$G_m$ (MPa)	$\nu_m$
18N	90.5	6.97	48.5	0.40	0.72	0.85	0.35	6.76	0.37
19N	93.5	7.29	49.3	0.40	0.73	0.92	0.31	6.15	0.39
24N	75.4	5.96	45.5	0.43	0.69	0.96	0.31	4.65	0.40
	At maximum shear stress								
18N	130.5	9.60	54.2	0.42	0.78	1.59	0.16	4.30	0.47
19N	130.1	9.76	54.5	0.42	0.78	1.68	0.11	4.07	0.48
24N	123.8	9.0	53.2	0.40	0.76	2.5	-0.10	2.40	0.50

Table 3.7: Shear Properties of Dense Ticino Sand in the Modified APSR Cell.



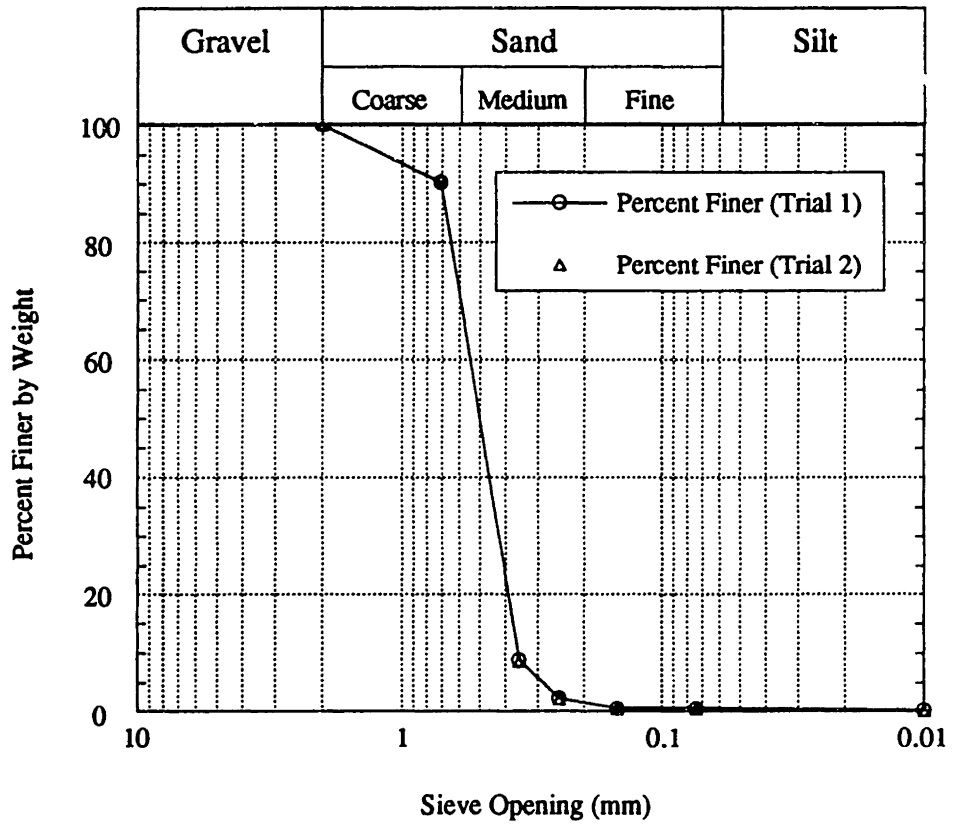
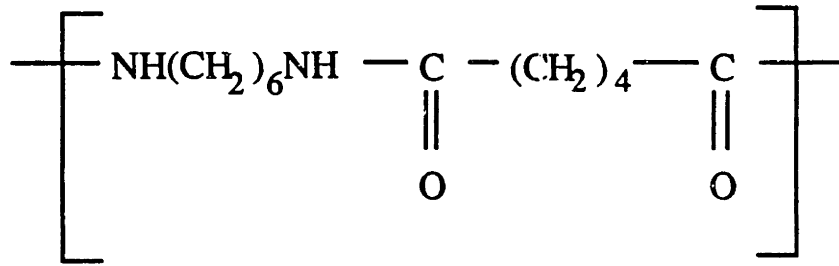
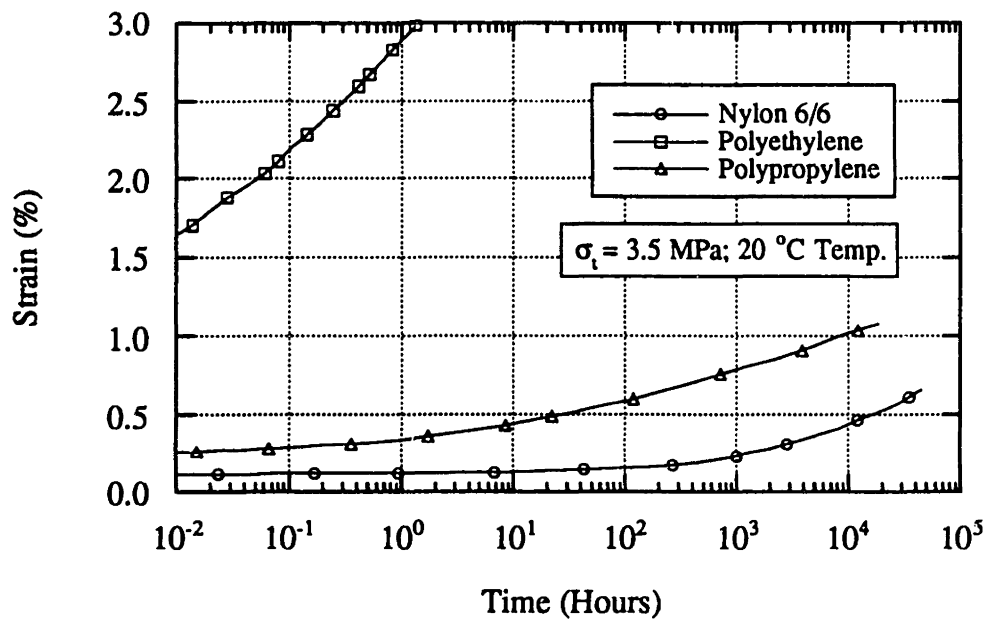


Figure 3.1: Grain Size Distribution for Ticino Sand (after Larson, 1992).



(a) Molecular Structure of Nylon 6/6 Monomer.



(b) Tensile Creep Behavior of Common Plastics  
(after Ogorkiewicz, 1970).

Figure 3.2: Molecular Structure of Nylon 6/6 and Creep Behavior of Plastics.

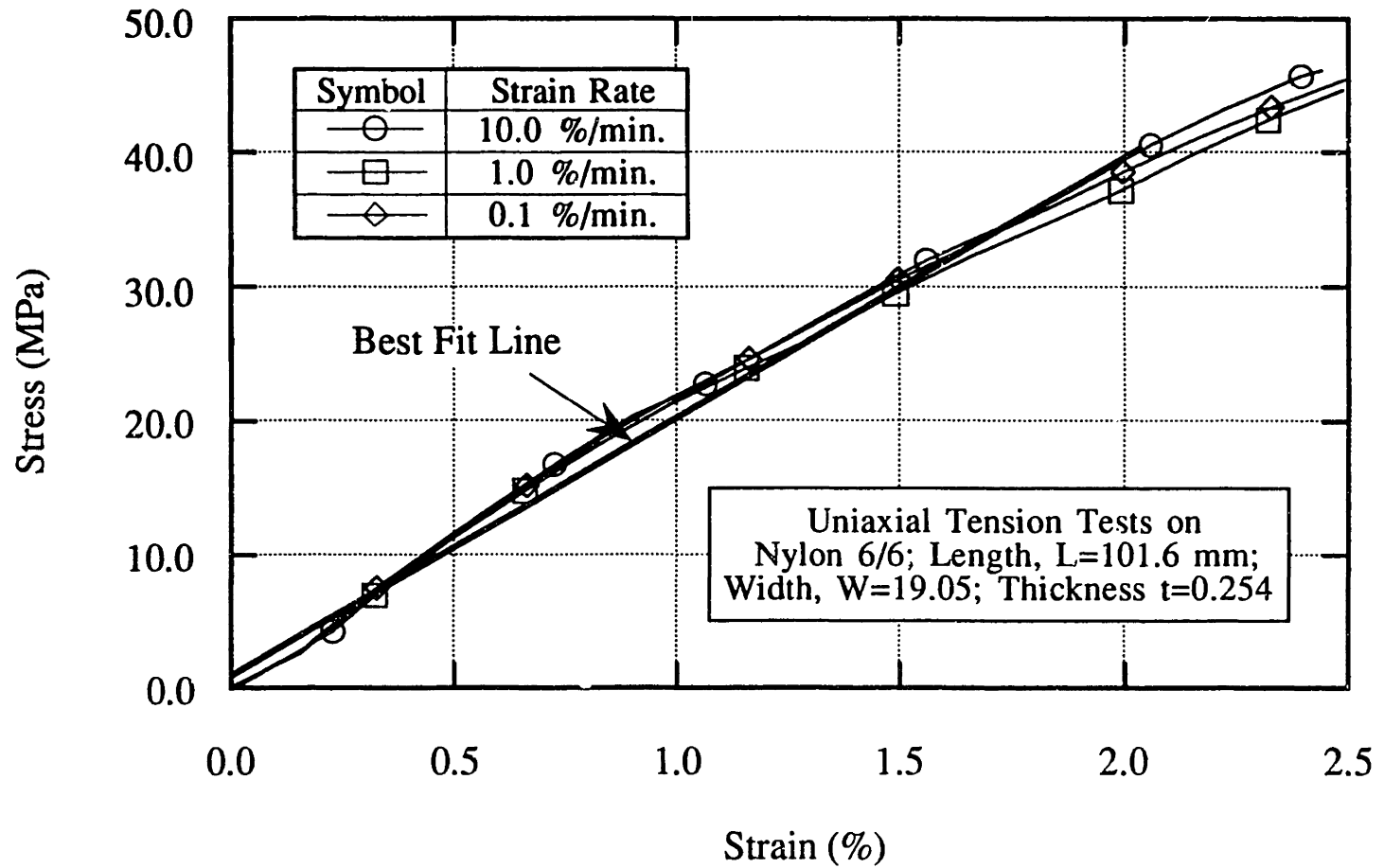
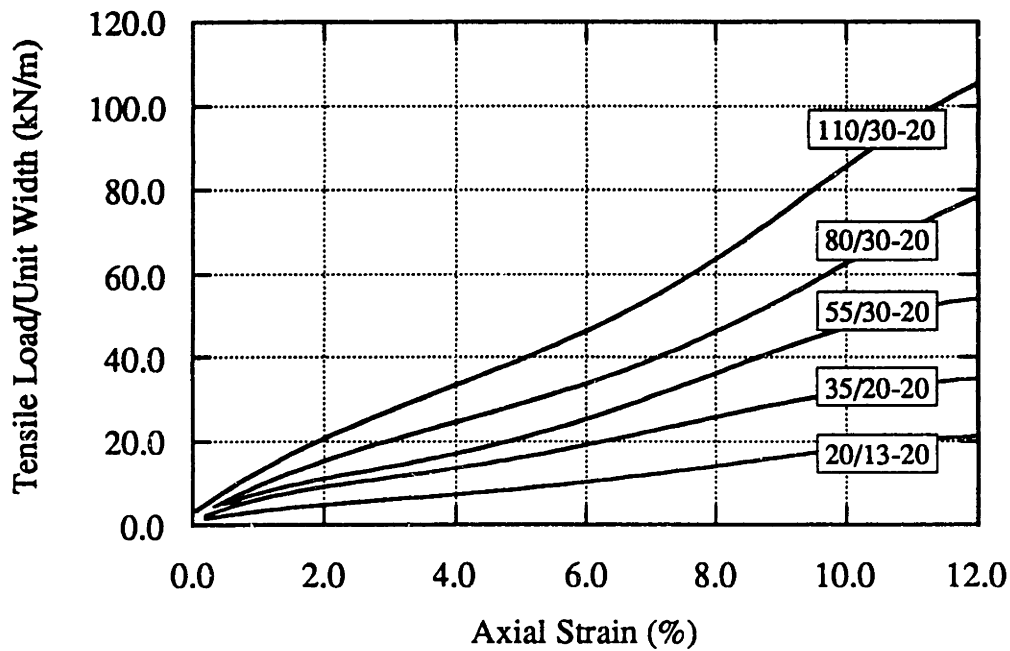
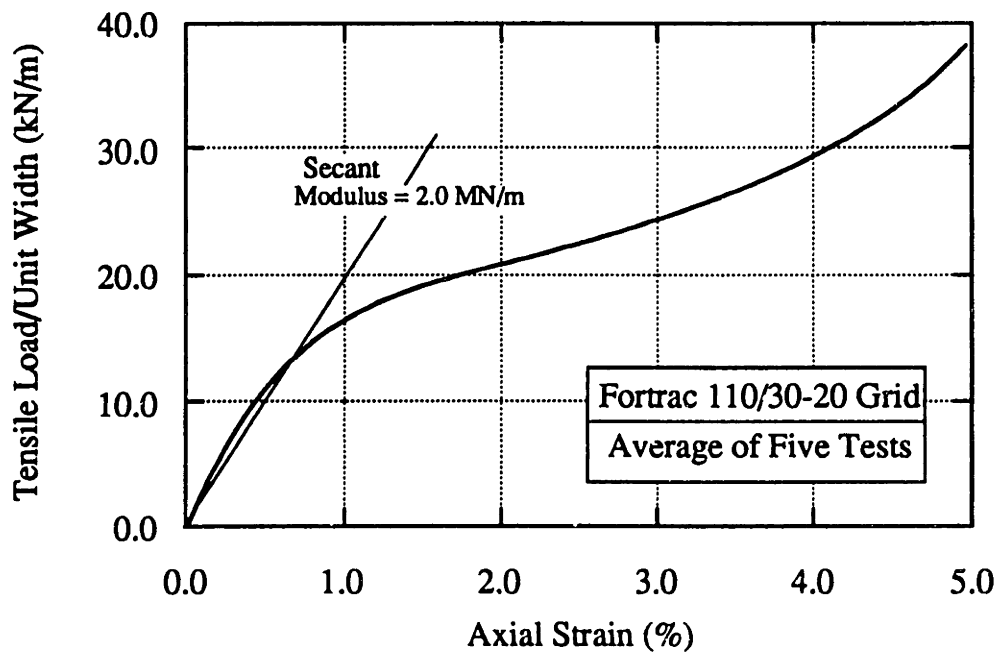


Figure 3.3: Typical Stress-Strain Results for In-Isolation Tension Tests on Nylon 6/6.



(a) Typical Tensile Stress-Strain Response of Different Fortrac Geogrids.



(b) Small Strain Behavior of Fortrac 110/30-20.

Figure 3.4: Stress-Strain Behavior of Fortrac Woven Geogrids  
(Source Huesker, 1994).

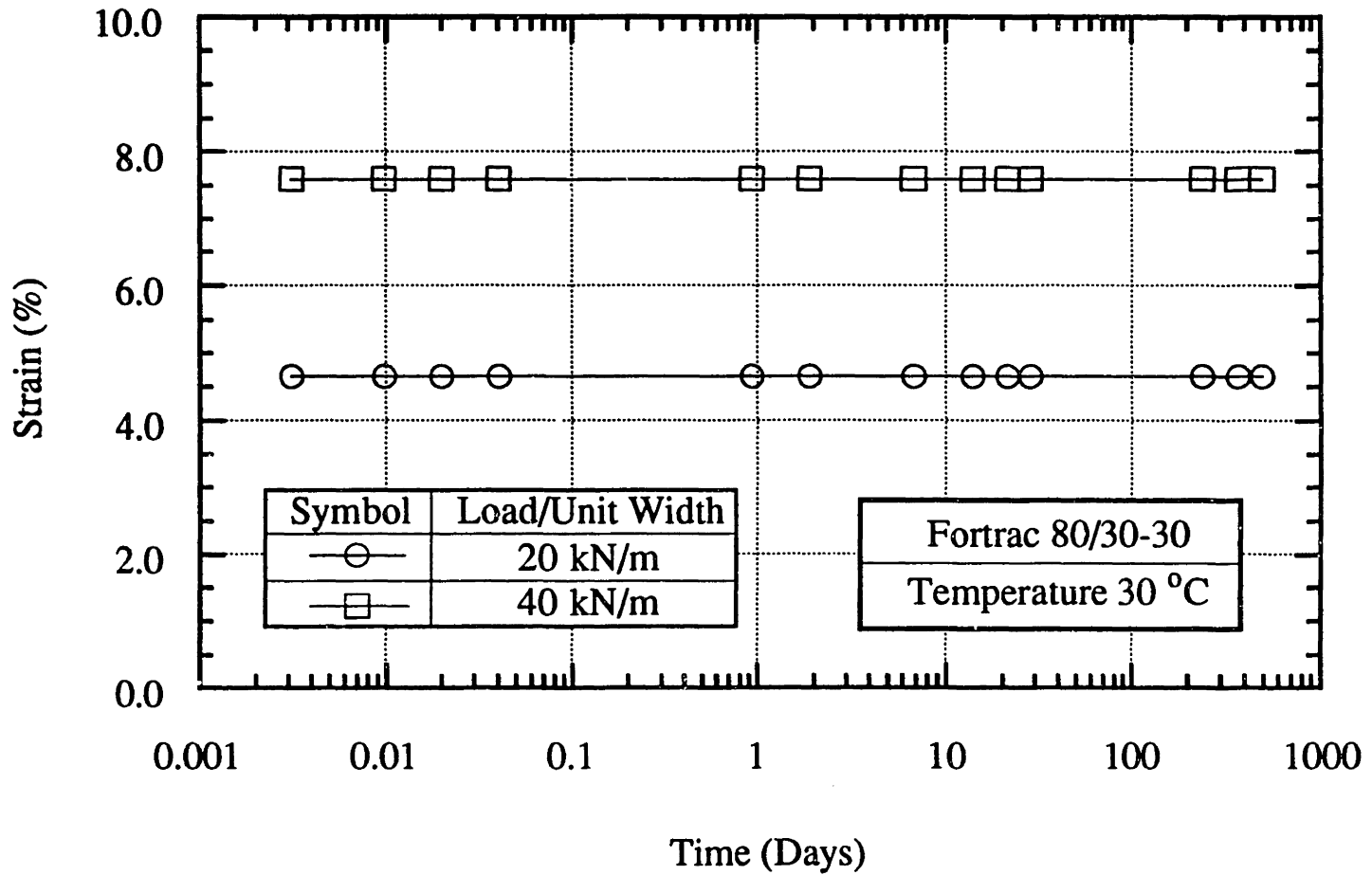
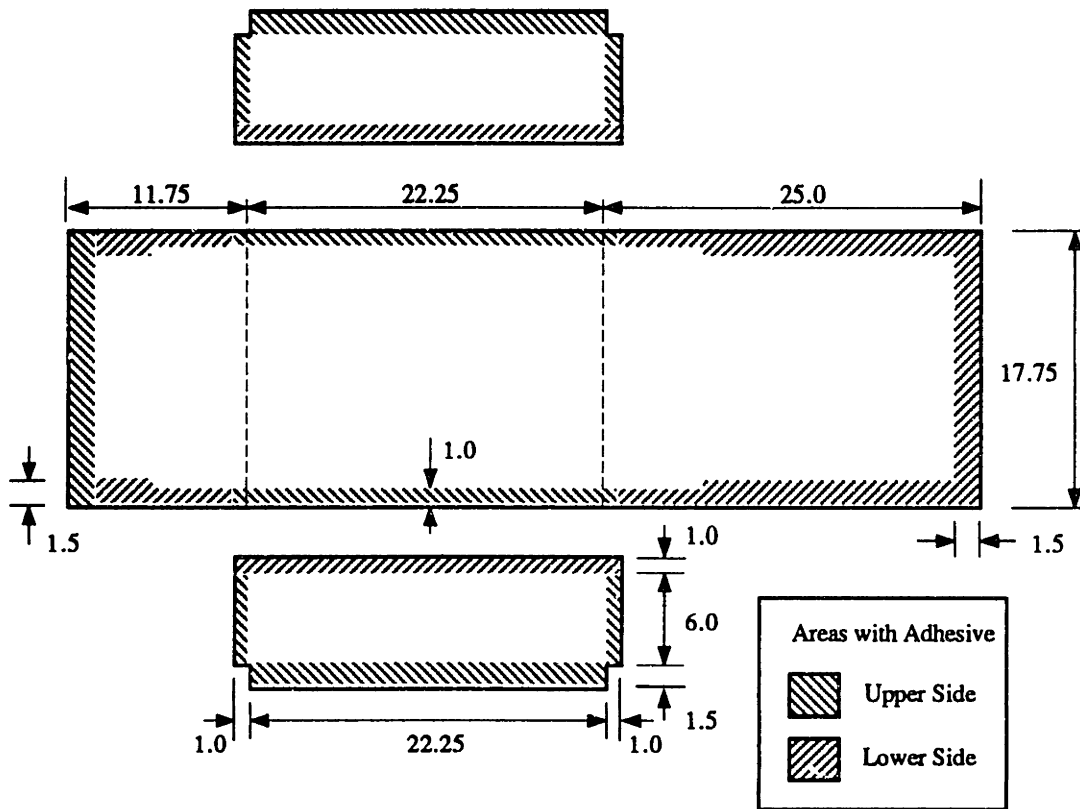
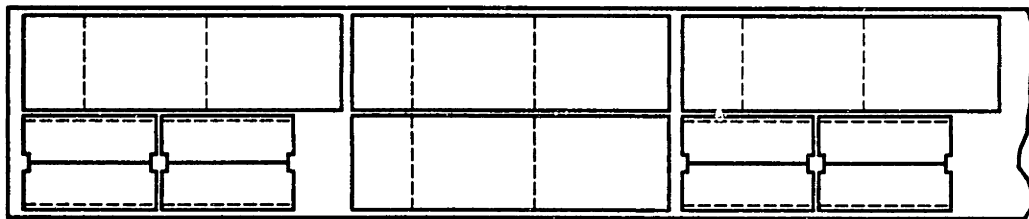


Figure 3.5: Dimensional Stability of Fortrac Grid Under Sustained Loading  
(Source Huesker, 1994).



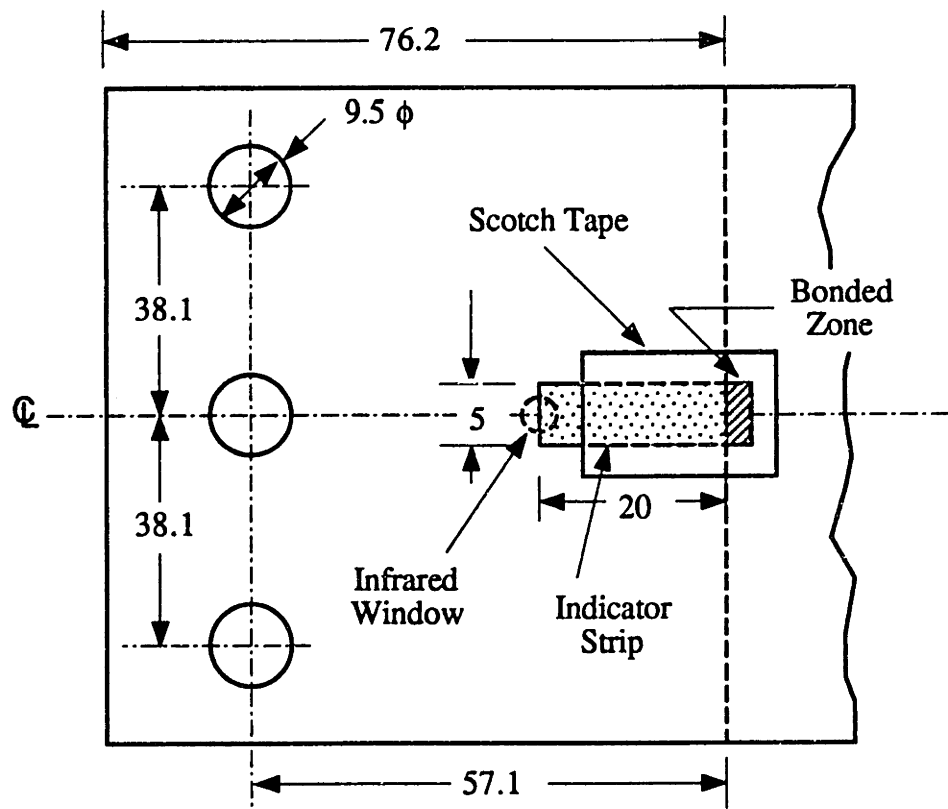
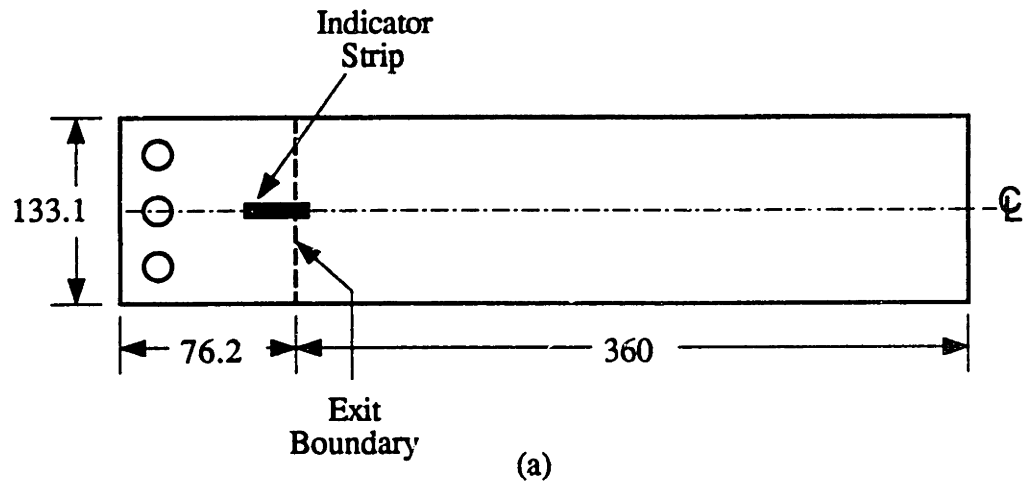
(a) Membrane Assembly

Note: All Dimensions are in Inches.



(b) Optimal Layout of Membrane Parts

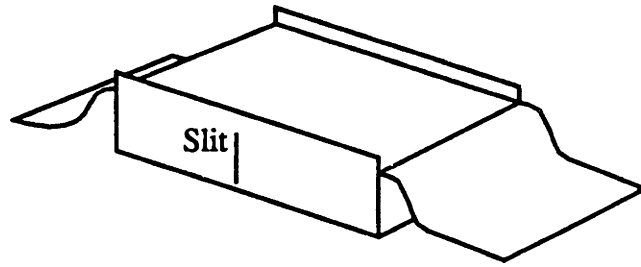
Figure 3.6: APSR Sample Membrane Parts and Layout.



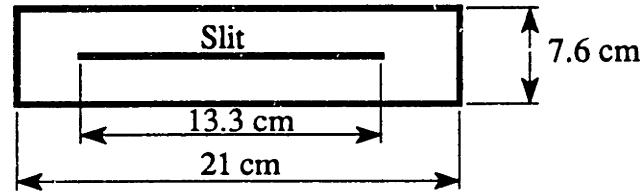
Note: All Dimensions are in mm.

(b)

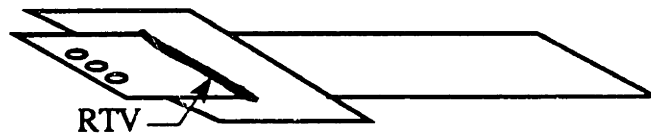
Figure 3.7: Details of Full Length Planar Reinforcement.



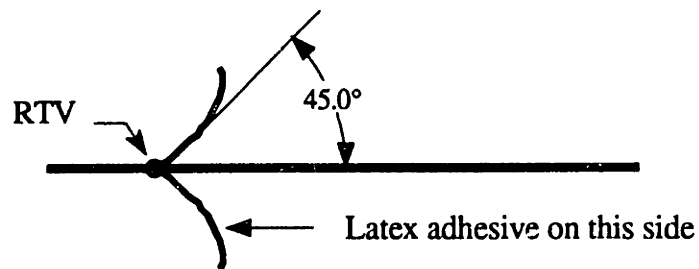
(a) Make a slit in the sample membrane



(b) Cut a rubber collar from 0.25 mm thick rubber sheet



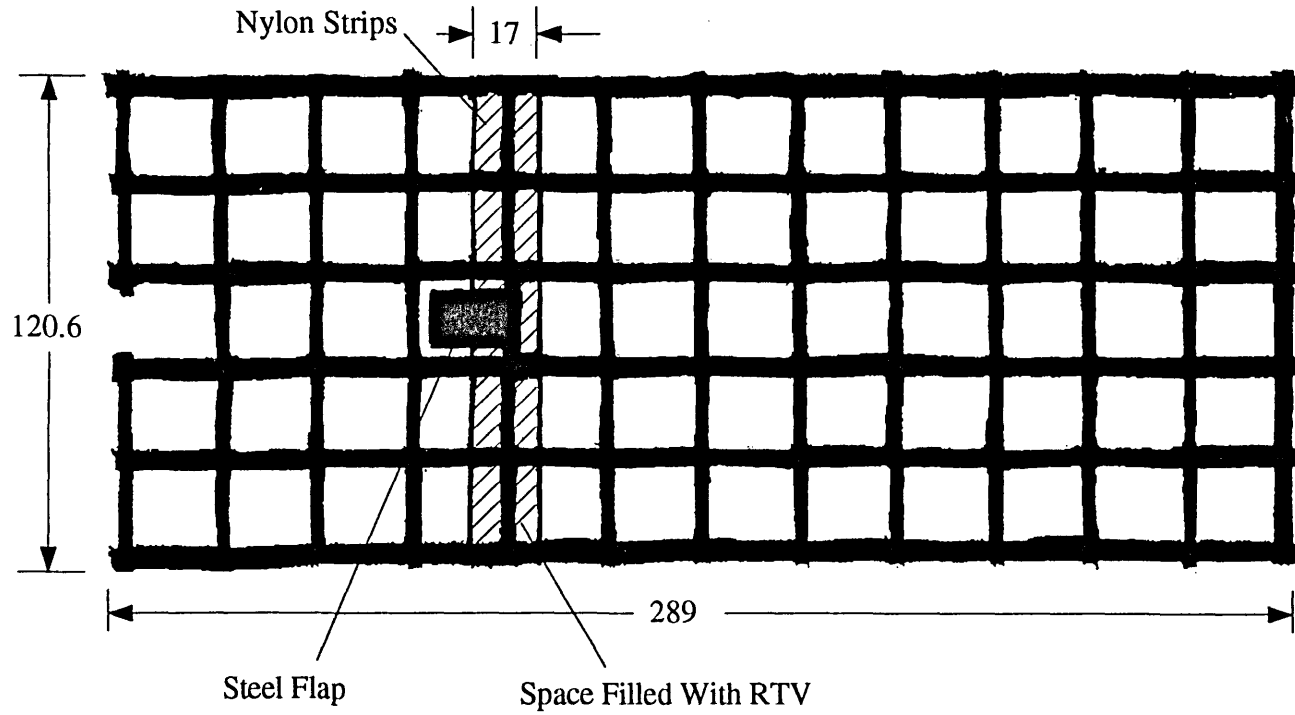
(c) Seal rubber collar to inclusion with RTV



(d) Ensure proper joint angle and allow RTV to cure

Figure 3.8: Connection between Planar Reinforcement and Specimen Membrane.

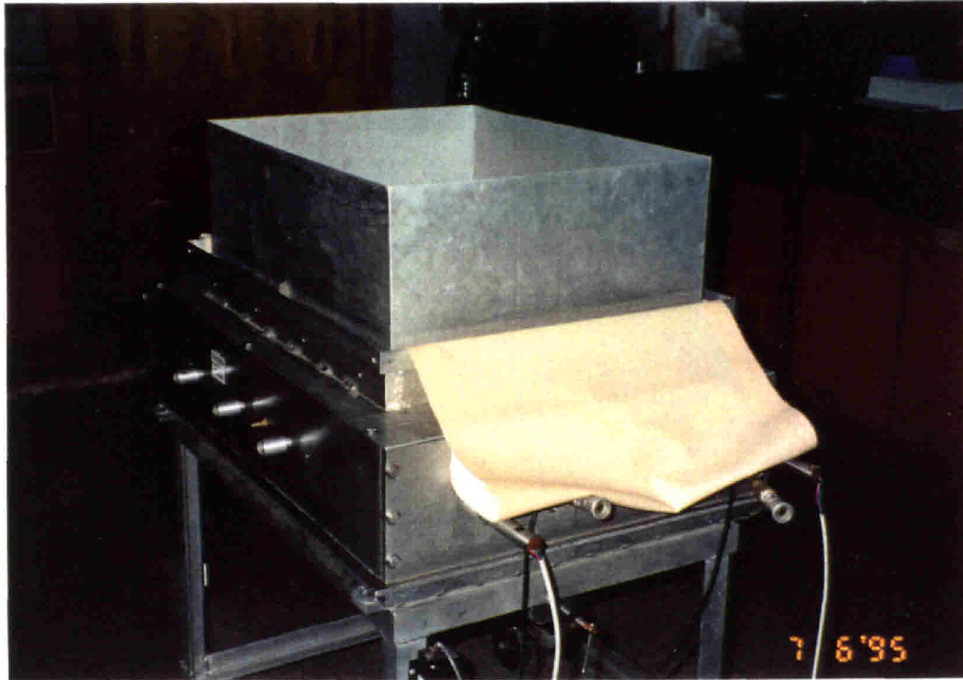




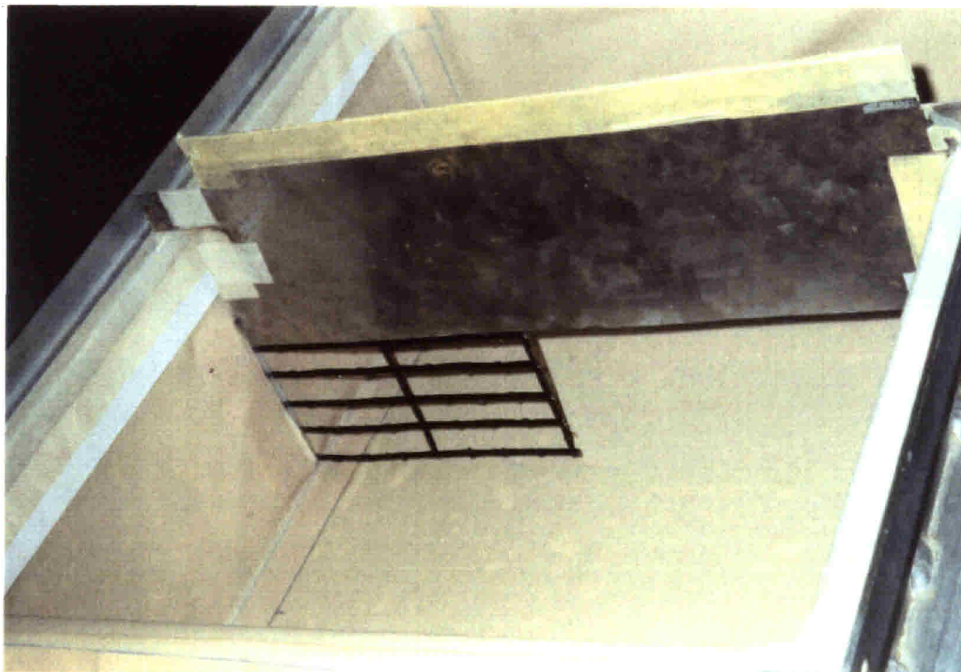
Note: All Dimensions are in mm.

Figure 3.9: Details of Geogrid and Indicator Flap Connection.





(a) Sand Deflector Shield.



(b) Reinforcement Position Holder.

Figure 3.10: Photographs of the APSR Cell Preparation.



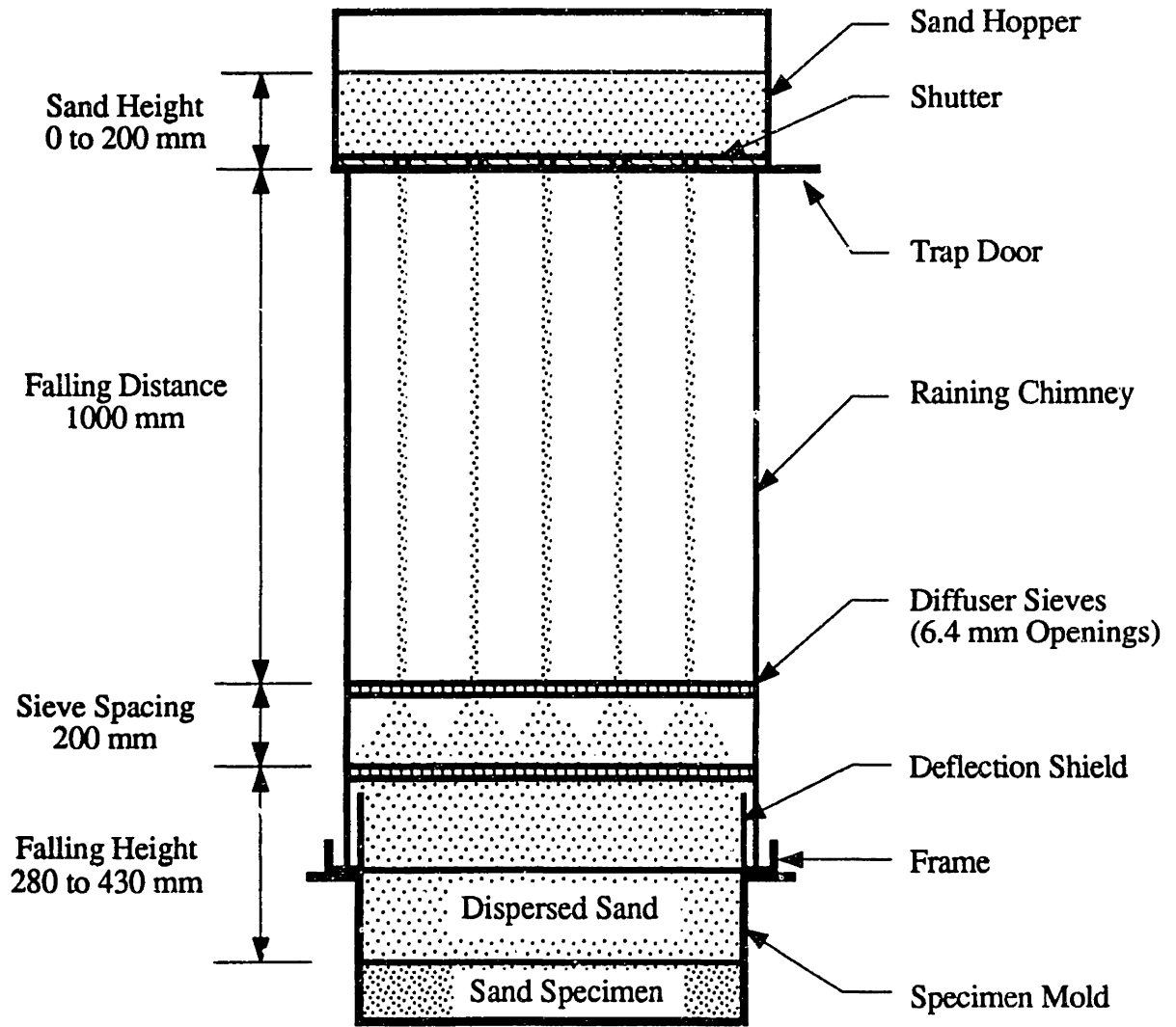


Figure 3.11: The APSR Cell Raining Apparatus.

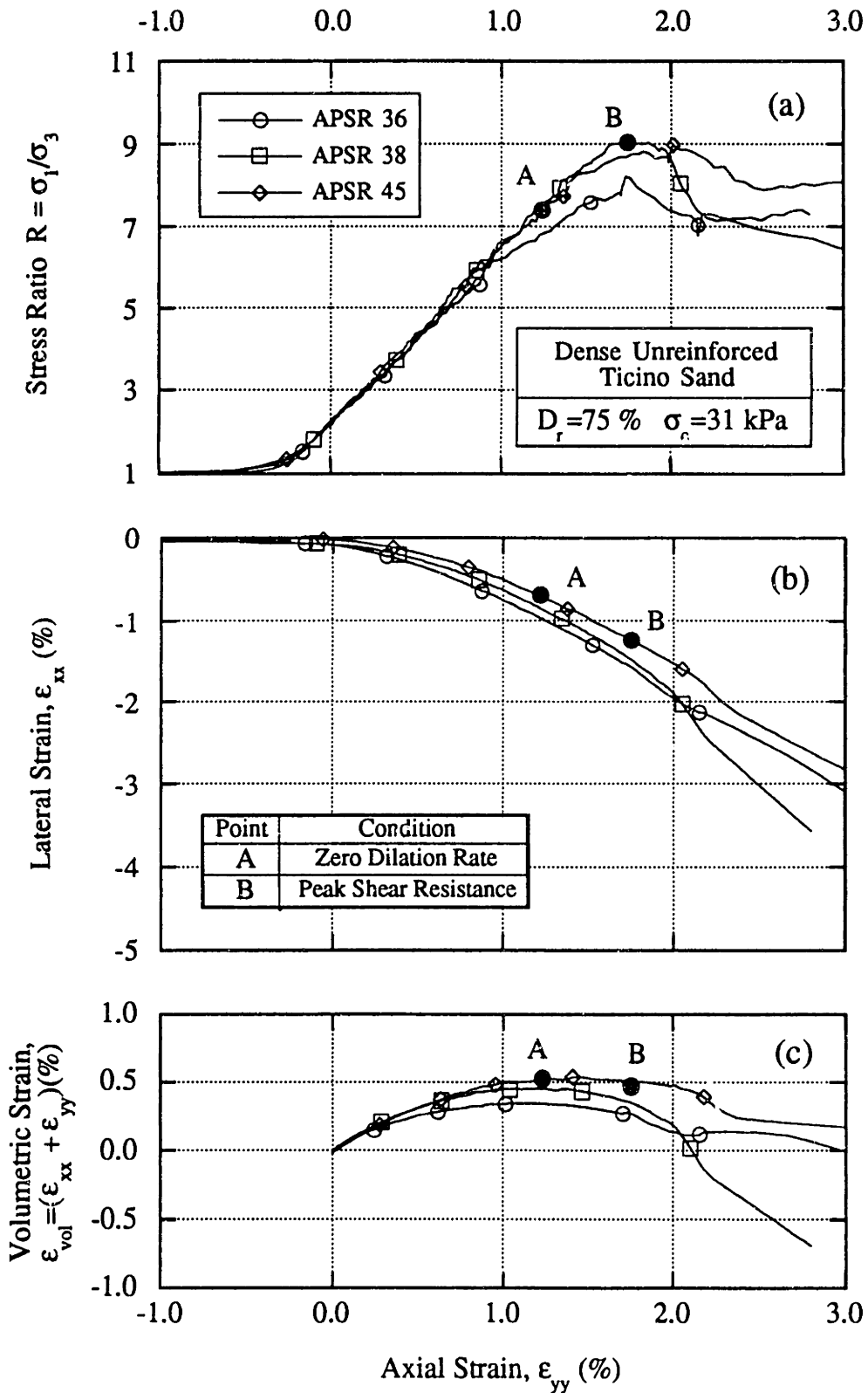


Figure 3.12: Stress-Strain Response of Dense Unreinforced Ticino Sand in the Original APSR Cell (after Larson, 1992).

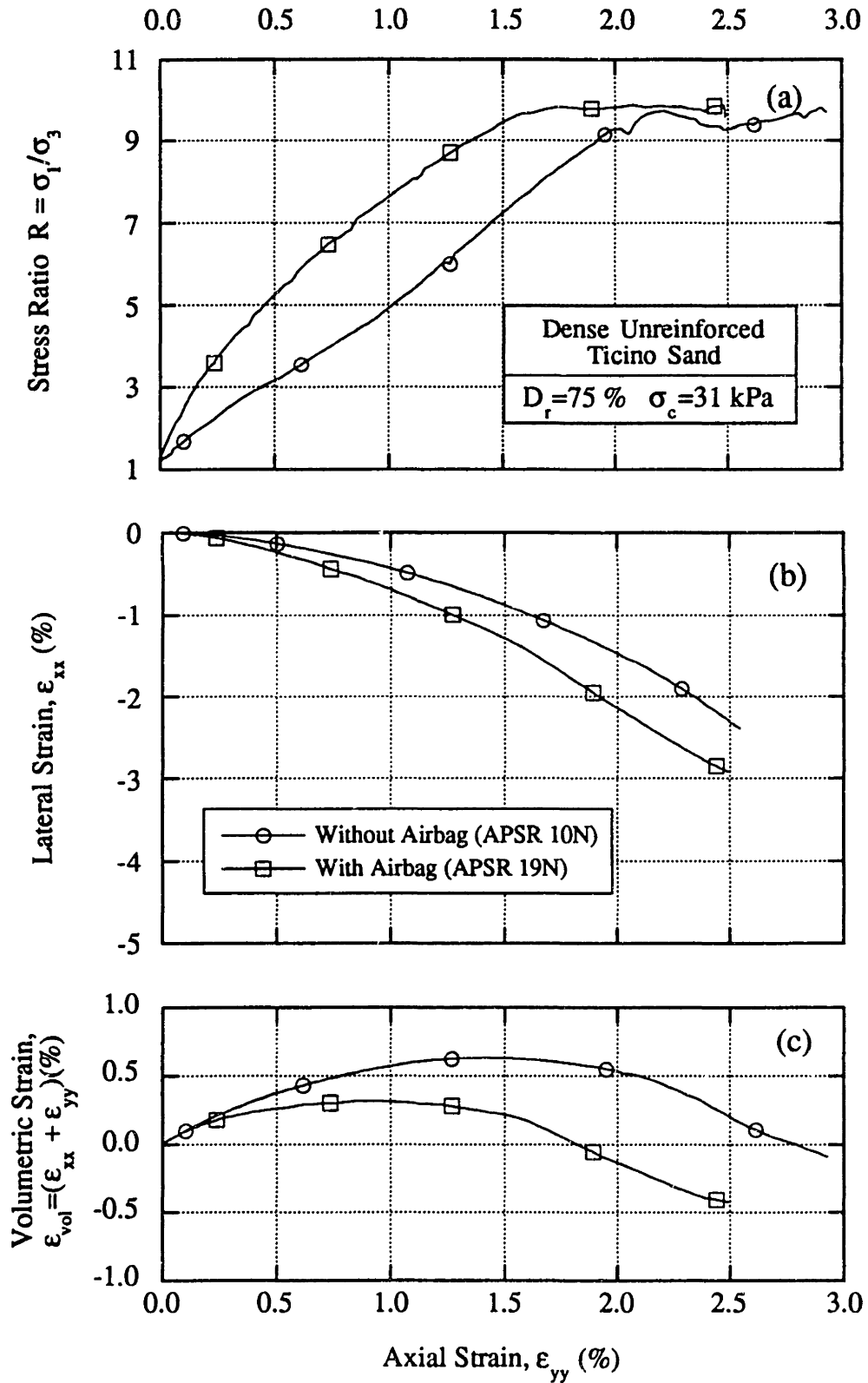


Figure 3.13: Comparison between the Shear Behavior of Dense Ticino Sand with and without the Airbag.

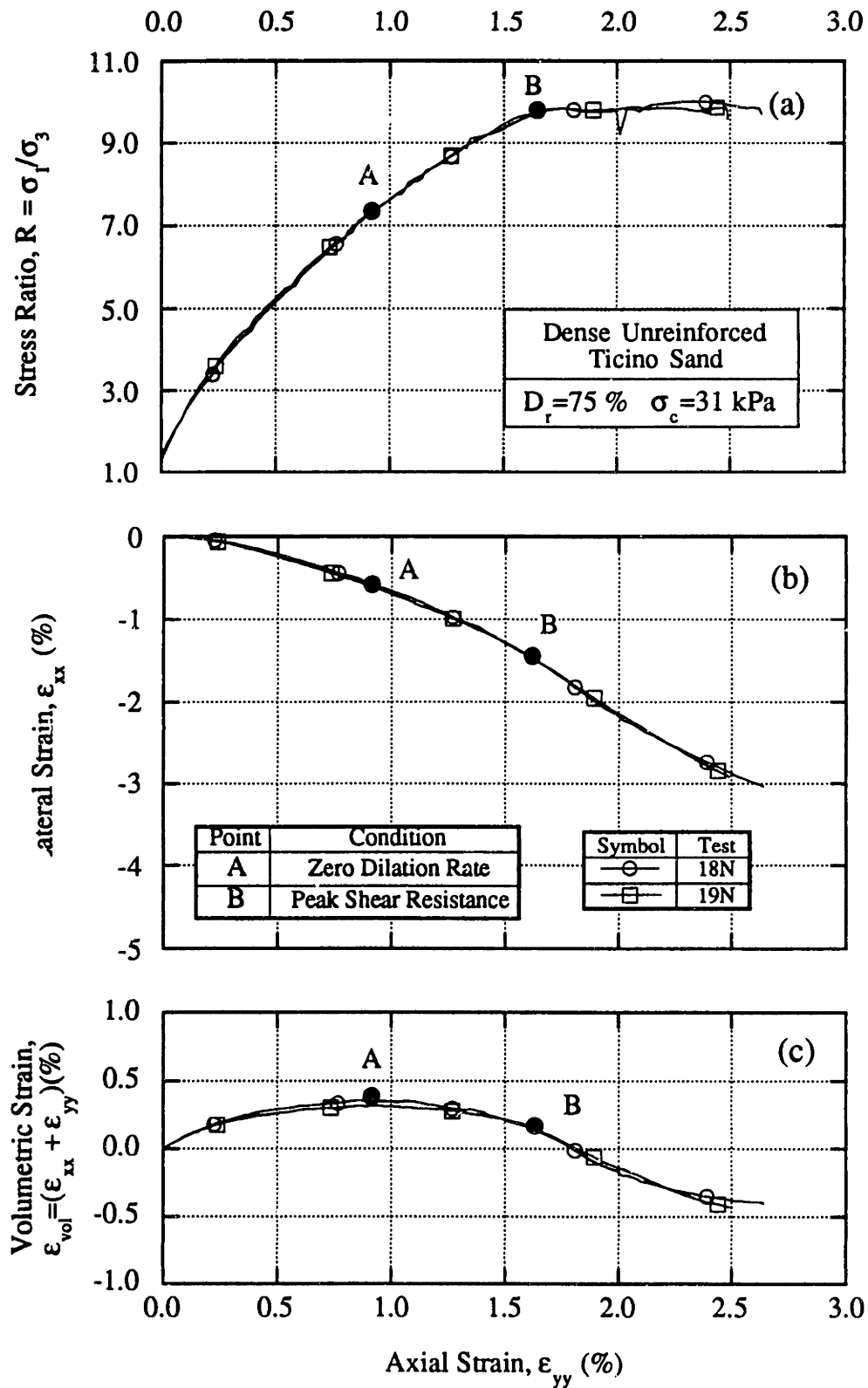


Figure 3.14: Repeatability of Externally Measured Shear Behavior in the Modified APSR cell.



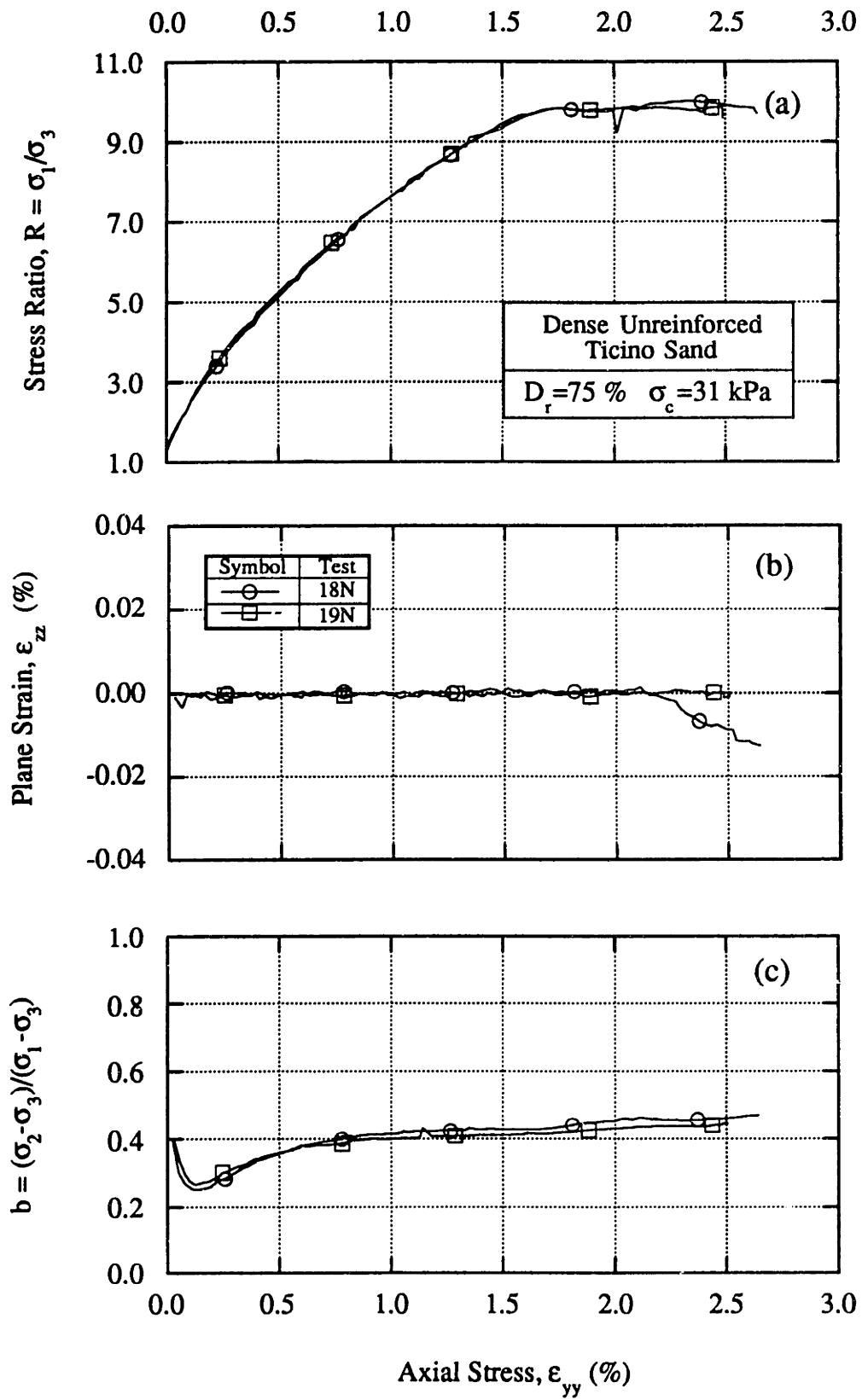


Figure 3.15: Variation of the Parameter b for Unreinforced Dense Ticino Sand.

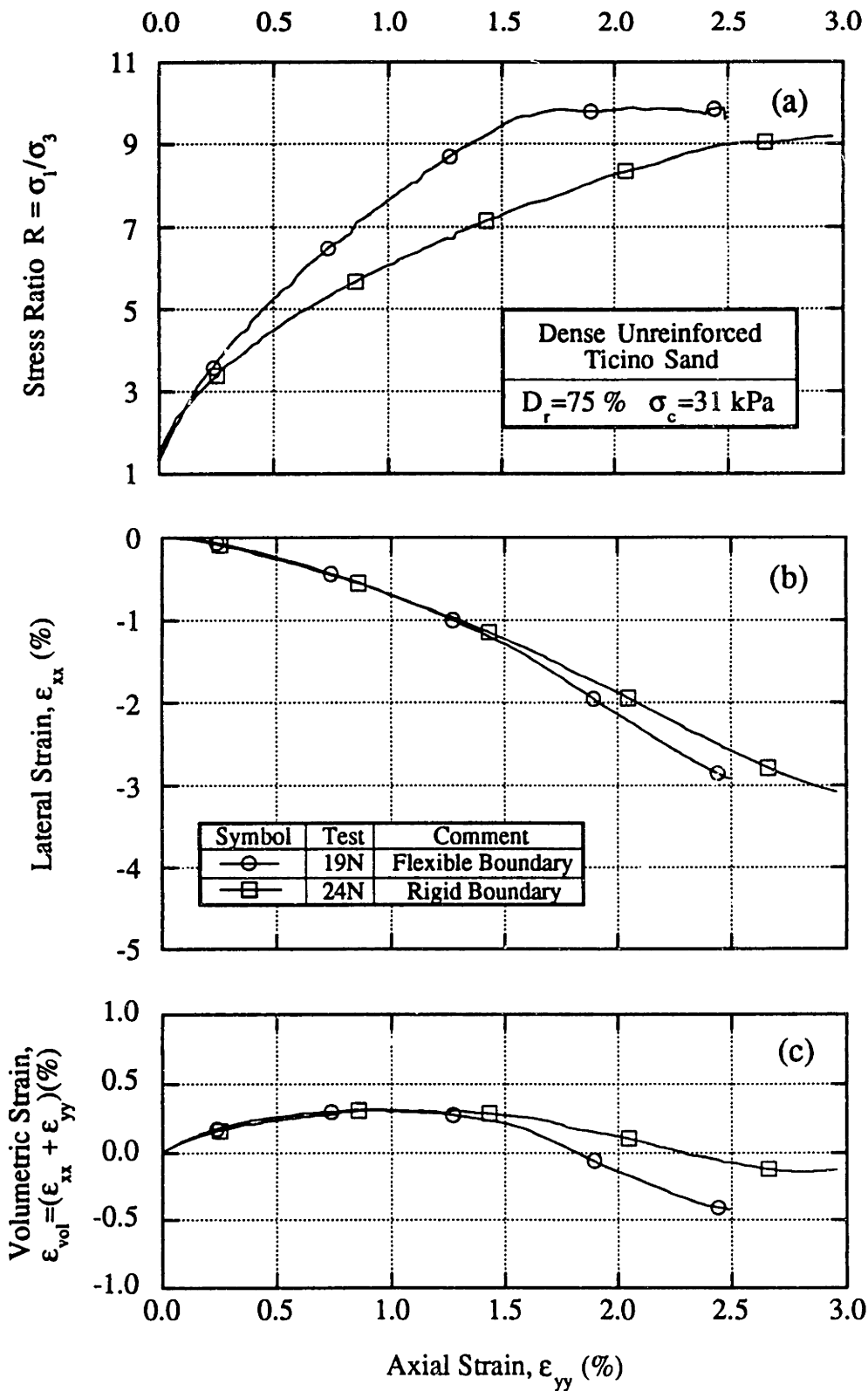


Figure 3.16: Effects of Rigid Major Principal Stress Boundary on the Measured Shear Behavior of Dense Ticino Sand.

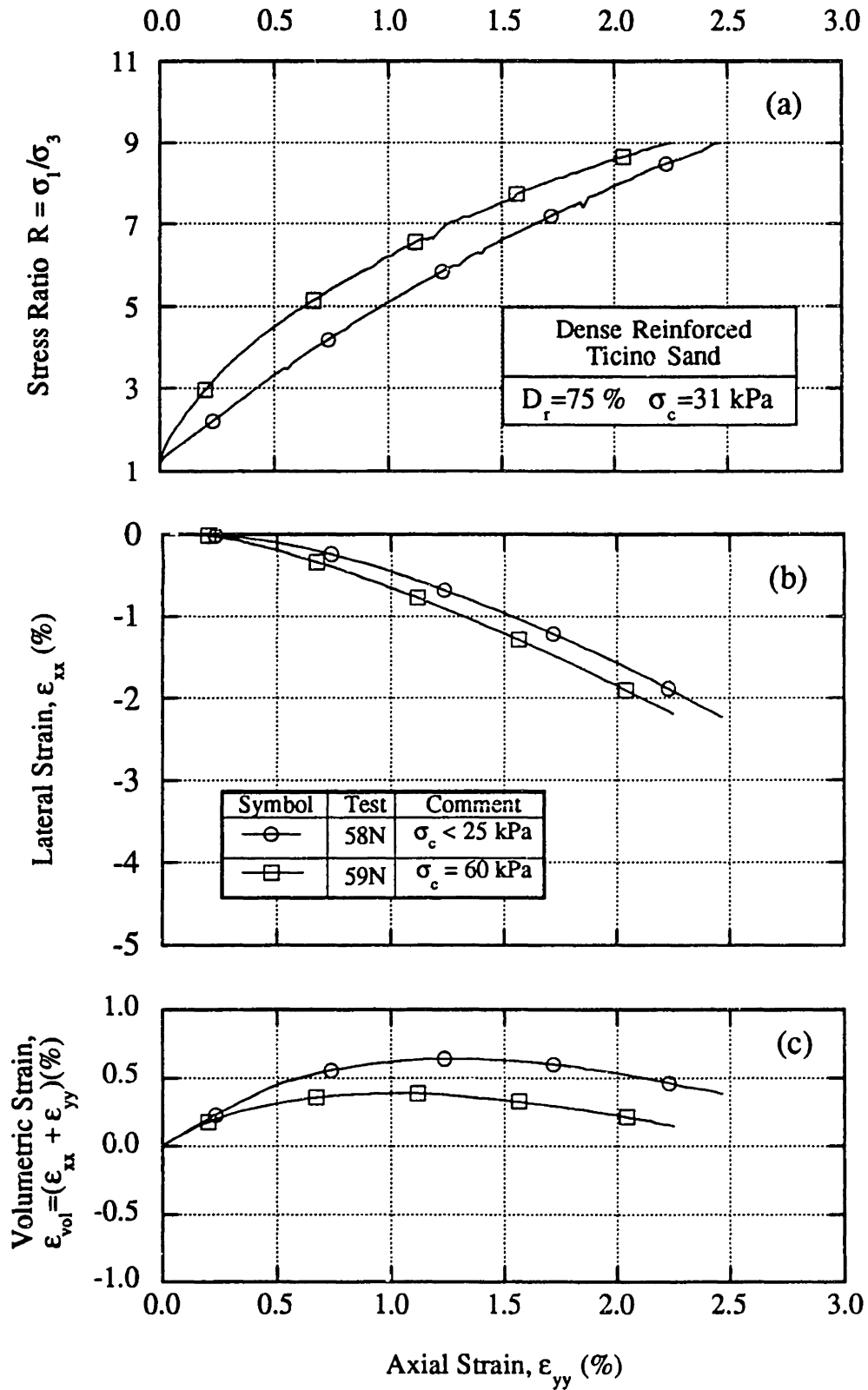


Figure 3.17: Effect of Overconsolidation due to Application of Vacuum during Sample Preparation.

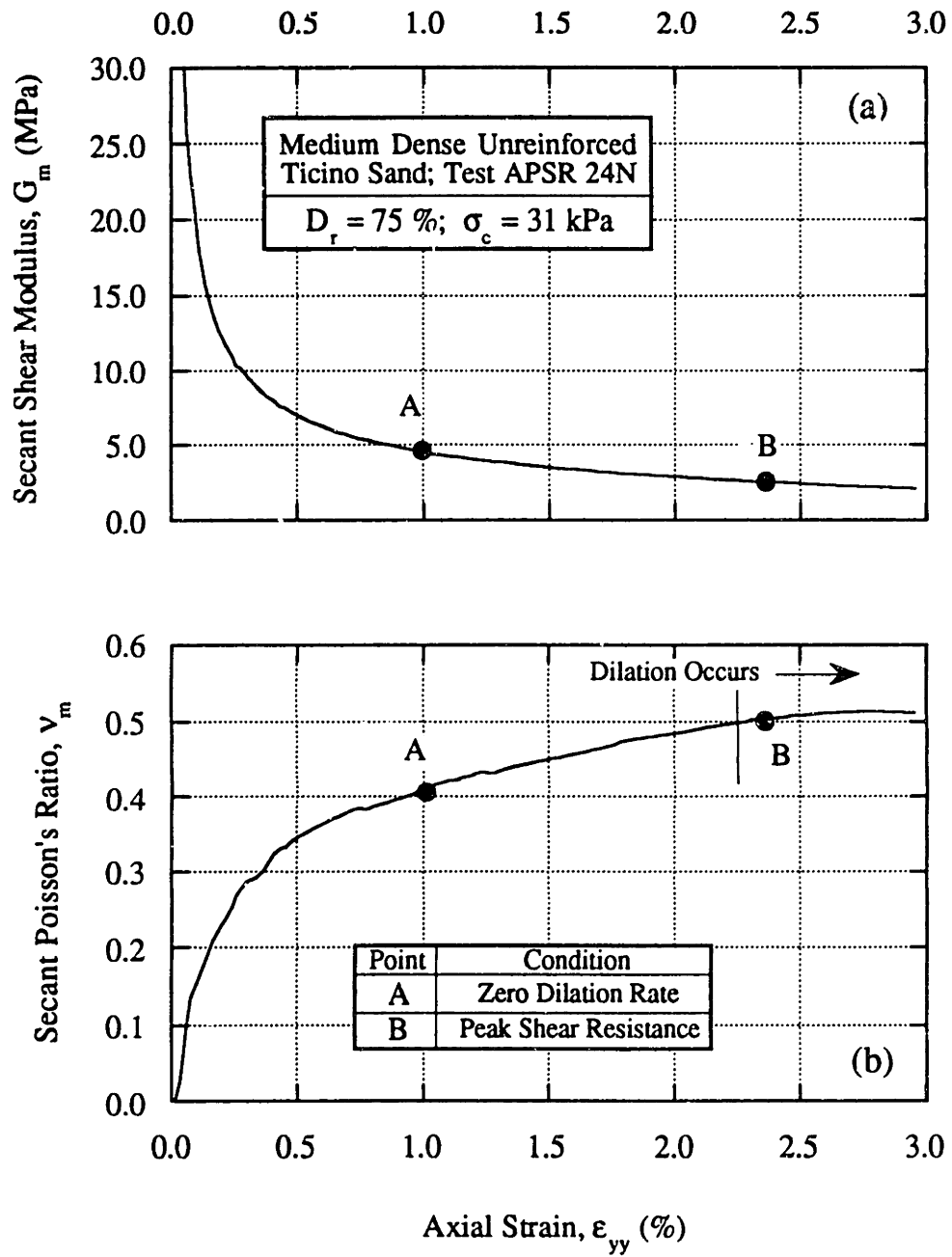


Figure 3.18: Variation of Secant Shear Modulus and Poisson's Ratio for Unreinforced Medium Dense Ticino Sand.

# Chapter 4: Load-Transfer in Planar Inclusions

## 4.1. Introduction

This chapter describes measurements of load-transfer for a series of APSR tests on dry Ticino sand reinforced with planar inclusions made from steel and Nylon 6/6. The chapter begins with a brief summary of important design changes made in the APSR cell during the present research program and their effects on the quality of the reinforced test results. Section 4.3 then presents load-transfer data for a series of APSR tests performed on dry, medium dense Ticino sand reinforced with various lengths of steel and Nylon 6/6 sheet inclusions, and establishes the reliability and repeatability of these measurements. Section 4.4 contains a discussion and evaluation of important trends in the load-transfer data. The final part of the chapter (Section 4.5) includes a brief review of shear-lag analysis; an approximate analytical method for estimating the tensile stresses in a single planar inclusion due to shearing of the surrounding soil matrix (Abramento and Whittle, 1993). A detailed comparison between the shear-lag predictions and load-transfer data from the APSR tests shows how the original linear, elastic shear-lag solution can be adapted to account for non-linear behavior of the matrix (sand) material.

## 4.2. Effects of Design Improvements on Test Data

Larson (1992) reported preliminary measurements of load-transfer in the original APSR cell for steel sheet reinforcements embedded in medium-dense Ticino sand. One of the main recommendations from Larson's thesis was the need for further refinements of the cell design in order to make the device more robust and hence, improve test reliability and efficiency. The first phase of the present research program comprised a total of 30 tests (approximately half of which were on unreinforced sand) which were performed to evaluate and refine the cell design and test procedures (refer to Appendix A for details). Table 4.1 summarizes these developmental tests and associated major changes in the hardware and software configuration of the APSR cell which include:

1. Implementation of full digital feedback control for the plane strain and reinforcement systems.
2. Use of an airbag for applying the confining air pressure,  $\sigma_3$ .
3. Conversion of the major principal stress boundary from flexible (uniform normal boundary traction) to rigid (uniform displacement) boundary condition.
4. An independent referencing frame and non-contact sensors for the plane strain sidewalls.
5. Development of a new modular, object-oriented, real-time software system integrating the control and data acquisition tasks.
6. Use of a rigid actuator arm for controlling the position of reinforcement at the exit point.
7. Use of an infrared sensor for reinforcement movement detection.

Figures 4.1 and 4.2 illustrate the cumulative effect of these improvements on measurement of load-transfer behavior by comparing results from a test performed in the refined APSR cell (APSR 35N) with those obtained from a typical test in the original cell (APSR 13N). Both tests were conducted on a 270 mm long steel sheet inclusion embedded in medium-dense Ticino sand ( $D_r = 75\%$ ). Fluctuations in the applied stress ratio,  $R$ , (Figure 4.1a) and poor control of the out-of-plane strain (Figure 4.1b, c) due to hardware and software problems occurred frequently in the original APSR cell and often masked the true trends in the inclusion load. However, the principal reason for erratic load pickup performance in the original APSR cell was inadequate control of the reinforcement exit point as illustrated in Figure 4.2. The figure shows that a very small change in position of the reinforcement exit point causes large fluctuations in the reinforcement tensile load. In order to measure the representative load pickup behavior for the steel sheet reinforcement, it is necessary to hold the inclusion within  $\pm 1 \mu\text{m}$  of its initial position (Appendix A). This level of precision in the reinforcement position control was finally achieved, beginning with test APSR 31N (Table 4.1), using a custom-built infrared sensor assembly (refer to Section 2.2.6 for details) and digital feedback control.

### 4.3. Presentation of APSR Test Data

This section describes the data obtained from a comprehensive program of load-transfer measurements using the refined APSR cell with different lengths of steel and Nylon 6/6 sheet reinforcements. The experimental program focuses on the effects of axial stiffness and inclusion length on the tensile stresses transferred to the inclusions. Table 4.2 presents a summary of the APSR tests performed to measure load-transfer

for planar inclusions as part of the present research program. All tests were performed on two ply sheet inclusions embedded in dry, medium-dense Ticino sand ( $D_r = 75\%$ ) at constant confining pressure  $\sigma_3 = 31$  kPa. Reference should be made to Chapter 3 for details of specimen preparation, test procedures, and material properties.

#### 4.3.1. APSR Tests on Steel Sheet Inclusions

A total of five high quality tests (APSR 32N to 35N) were performed in the modified APSR cell using 0.254 mm thick steel sheet inclusions. Inclusions with four different lengths ( $L/2 = 90, 180, 270, 360$  mm) were used to evaluate the effect of inclusion length on the tensile stresses transferred to the inclusions. The steel sheet reinforcements used in the APSR research each have stiffness per unit width<sup>1</sup>,  $J = 52500$  kN/m. Due to the relatively high axial stiffness of the steel reinforcements, the inclusion loads are very sensitive to any inaccuracies in the reinforcement position control. Reliable measurements of load-transfer for steel sheet inclusions have, therefore, proven to be the most difficult of all materials used in the APSR cell to date.

##### 4.3.1.1. Tensile Stress in the Reinforcement

The most important information provided by the APSR cell is the tensile load in the inclusion at the exit point, a location which is equivalent to the centerline of an inclusion with length,  $L$ . An external load cell attached to the reinforcement position control arm measures the tensile load in the inclusion (refer to Section 2.2.6). Figure 4.3a shows the measurements of 'centerline' tensile load,  $F_R$ , as a function of the

---

<sup>1</sup> The 'unit stiffness' of a planar reinforcement,  $J$ , [kN/m] has been defined as the product of elastic tensile modulus,  $E_f$ , and thickness,  $t$ .



externally applied stress ratio,  $R = \sigma_1/\sigma_3$ , for two tests on steel inclusions with half-lengths,  $L/2 = 270$  mm. The close agreement in the measurements of load-transfer for the two tests indicates that the results are highly consistent and repeatable. The tensile load-transfer does not begin until the applied stress ratio exceeds a characteristic value,  $R \geq R^*$ . During the initial phase of the test ( $R < R^*$ ), the reinforcing inclusion is allowed to move freely and the exit point displacements are continuously monitored (refer to Section 3.3.2.2). Figure 4.4 shows that this phase of the test is characterized by an outward movement of the inclusion accompanied by a slight drop in the initial seating load applied to the reinforcement. The active control of the reinforcement is established when the exit point displacements indicate the onset of tensile load-transfer, after which tensile loads in the inclusions increase monotonically with the applied shear stress in a non-linear manner. The load transfer gradient,  $dF_R/dR$  increases gradually with the applied stress ratio and becomes constant for  $R > 7$ .

Figure 4.5 compares the exit point tensile loads measured for steel sheet inclusions with four different lengths,  $L/2 = 90, 180, 270, 360$  mm. The load in the reinforcement, at a given applied stress ratio, is a function of the inclusion length. At an applied stress ratio,  $R = 7$  (corresponding to a mobilized friction angle,  $\phi'_{mob} = 48.6^\circ$ ), the measured centerline load ranges from  $F_R = 23.5$  kN/m at  $L/2 = 360$  mm to 2.8 kN/m at  $L/2 = 90$  mm. In general, decreasing the inclusion half-length,  $L/2$ , causes an increase the characteristic stress ratio,  $R^*$ , corresponding to the onset of tensile load-transfer (Table 4.2). The maximum value of load transfer gradient ( $dF_R/dR$  when  $R \rightarrow 9$ ) also increases with the inclusion length from 2.15 at  $L/2 = 90$  mm to 7.75 at  $L/2 = 360$  mm.

Table 4.3 shows the computed values of inclusion centerline loads and strains (corresponding to the exit point location) in the longest steel inclusion ( $L/2 = 360$  mm).

The data show that the maximum strain in the steel reinforcement,  $\epsilon_R$  is lower than 0.07%. At any given value of the applied stress ratio, the maximum reinforcement strain,  $\epsilon_R$ , is substantially smaller than the corresponding value of average lateral strain,  $\epsilon_{xx}$ , in the soil.

In test APSR 32N ( $L/2 = 180$  mm), a rapid unloading of the specimen occurred at applied stress ratio,  $R = 7.5$  due to an experimental problem with the loading platforms. Figure 4.5 shows that approximately 50% decrease in the applied stress ratio causes very little drop in the inclusion load. Upon subsequent loading and shearing of the specimen, the load pickup resumes along the previously established path. This result shows that shearing of the soil locks tensile stresses within the inclusion. The load-transfer mechanism, therefore, is a non-linear process in which full recovery of stresses does not take place upon unloading.

The reinforcement centerline loads are plotted against the externally measured (boundary) lateral strain in the soil,  $\epsilon_{xx}$ , in Figure 4.6. Except where noted otherwise, the lateral strains in the soil were computed by averaging displacement readings taken at two or more locations along the specimen front face (refer to Section 2.3.1). The data on Figure 4.6 show that centerline loads are well described as linear functions of the lateral strain,  $\epsilon_{xx}$ . The lateral strains in the soil matrix can be related to the maximum inclusion load through an 'apparent stiffness coefficient' defined as,  $E_a = dF_R/d\epsilon_{xx}$ . Table 4.4 summarizes the results from linear regression analyses which show that the apparent stiffness for the steel reinforcement ranges from  $E_a = 6800$  kN/m at  $L/2 = 360$  mm to 530 kN/m at  $L/2 = 90$  mm with the average regression coefficient,  $r^2 = 0.996$ . All of the regression lines in Figure 4.6 pass through the origin except for the shortest inclusion ( $L/2 = 90$  mm) which shows a small offset strain,  $\epsilon_{xx}^* = -0.14\%$ .

#### 4.3.1.2. Shear Behavior of Reinforced Ticino Sand

The presence of steel sheet reinforcement has significant effects on the plane strain shear behavior of Ticino sand in the APSR cell. The development of tensile stresses within the reinforcement increases the confining lateral stress,  $\sigma_{xx}$ , in the surrounding soil matrix, and hence, inhibits lateral strain in the matrix. Figure 4.7 shows the applied stress ratio,  $R = \sigma_1/\sigma_3$ , plotted against the average lateral strain in soil,  $\epsilon_{xx}$ , for different lengths of reinforcement. All reinforced tests gave smaller lateral strains than the unreinforced sand. For a given stress ratio, the average lateral strain in the soil decreases with an increase in the inclusion half-length,  $L/2$ . The longest steel sheet reinforcement ( $L/2 = 360$  mm) reduces the average lateral strain in the soil by as much as 50% (at a given applied stress ratio,  $R$ ). The effect of higher confinement due to reinforcement stresses can also be seen in more conventional presentations of the stress-strain behavior, Figure 4.8. For the specimen with the longest steel reinforcement ( $L/2 = 360$  mm), the secant elastic shear modulus,  $G_m = 9.5$  MPa, at axial strain,  $\epsilon_{yy} = 0.5\%$ , is approximately 35% higher than the shear modulus for unreinforced sand at the same axial strain (Figure 4.8a). The presence of the steel reinforcement tends to inhibit dilatancy in the sand matrix (Figures 4.8b and c). The average value of peak volumetric compressive strain in reinforced tests is about 50% higher than that for the unreinforced test. No failure was generated within the soil matrix<sup>2</sup> for the tests with reinforcing inclusions longer than 180 mm. Instead, these tests were terminated when the external load cell reached its capacity.

---

<sup>2</sup> In these APSR experiments, there is no tensile rupture of the reinforcing inclusion.

### 4.3.2. Tests on Nylon 6/6 Sheet Inclusions

A total of five successful tests (APSR 50N to 54N) were performed on 0.51 mm thick, two ply, Nylon 6/6 sheet inclusions embedded in dry, medium-dense Ticino sand ( $D_r = 75\%$ ) at confining pressure,  $\sigma_3 = 31$  kPa (Table 4.2). The elastic tensile modulus,  $E_f$ , of Nylon 6/6 is two orders of magnitude lower than that of steel (refer to Section 3.2.2 for material properties). For the Nylon 6/6 reinforcements used in this research, the stiffness per unit width,  $J = 980$  kN/m, is about 50 times lower than that for the steel reinforcements, and is comparable to many planar geosynthetics used in practice (Bonaparte and Schmertmann, 1987).

Figure 4.3b shows the reinforcement centerline load,  $F_R$ , recorded by the external load cell as a function of the applied stress ratio,  $R = \sigma_1/\sigma_3$ , for two Nylon sheet inclusions with half-length,  $L/2 = 270$  mm. The data in Figure 4.3b show that the experimental observations of load-transfer in Nylon 6/6 inclusions are highly repeatable and exhibit a response which is qualitatively similar to that reported previously for the steel sheet reinforcements (Figure 4.3a). However, the magnitude of the tensile load in the Nylon 6/6 reinforcements in Figure 4.3b is approximately a factor of four times smaller than for steel sheet inclusion of same length;  $F_R = 5.98$  kN/m at  $R = 8$  versus  $F_R = 23.43$  kN/m (refer to Table 4.3). Figure 4.9 shows an enlarged view of the initial response for a 180 mm long Nylon 6/6 inclusion. During the first phase of shearing, the inclusion is forced out of the cell and active position control occurs at  $R \approx 3$  marking the onset of tensile load-transfer. The total outward movement at  $R = 3$  is approximately  $10 \mu\text{m}$  and is much larger than comparable data for a steel sheet inclusion of the same length (Figure 4.4).

Figure 4.10 shows the effects of inclusion length on the load-transfer behavior of Nylon sheet inclusions. The offset stress ratio corresponding to initiation of tensile

load-transfer,  $R^*$ , as well as maximum value of load-transfer gradient,  $dF_R/dR$ , depend on the inclusion length (Table 4.2). The load in the reinforcement at a given applied stress ratio increases with the inclusion length, although the gain is not as large as that for the steel sheet reinforcements. At an applied stress ratio,  $R = 7$ , the measured centerline loads range from  $F_R = 4.25$  kN/m at  $L/2 = 360$  mm to 1.75 kN/m at  $L/2 = 90$  mm. Table 4.3 shows the inclusion centerline strains (corresponding to the exit point location) in a 360 mm long Nylon 6/6 inclusion at various values of the applied stress ratio. The data show that the maximum strain in the Nylon inclusion is of the order of 1%.

Figure 4.11 shows the inclusion loads plotted against the average lateral strain in the soil,  $\epsilon_{xx}$ , measured at the specimen boundary. As a first approximation, the inclusion load can be described as a linear function of the lateral strain, although the relationship is less linear than previously observed for steel, as indicated by the regression coefficients in Table 4.4. The apparent stiffness coefficient,  $E_a = dF_R/d\epsilon_{xx}$ , ranges from 370 kN/m at  $L/2 = 360$  mm to 150 kN/m at  $L/2 = 90$  mm. For shorter inclusions ( $L/2 = 90$  and 180 mm) in Figure 4.11, the load-transfer does not begin until a small amount of offset lateral strain,  $\epsilon_{xx}^* \approx -0.15\%$ , has occurred in the sand specimen.

Figures 4.12 and 4.13 compare the shear behavior of medium-dense Ticino sand reinforced with various lengths of Nylon 6/6 inclusions. As in the case of steel inclusions, the stresses and strain were measured at the specimen boundaries. The figures also include results from the base-case test on unreinforced Ticino sand (APSR 24N) for comparison. Figure 4.12b plots the applied stress ratio,  $R$ , against the

average<sup>3</sup> lateral boundary strain,  $\epsilon_{xx}$ . The figure shows that for both the reinforced and unreinforced tests, the average lateral deformations at the specimen boundary are essentially zero until the applied stress ratio,  $R > 1.8$  to 2.4. Figure 4.12a shows the same data as in Figure 4.12b, plotted on a larger scale. Within the experimental scatter, the plots in Figure 4.12 differ very little from each other indicating that the presence of Nylon 6/6 reinforcement has little or no restraining effect on the lateral deformations in the sand. Since the magnitude of reinforcement loads in Nylon 6/6 inclusions is relatively small, the additional confinement due to reinforcement stresses is not significant enough to change stress-strain behavior of the sand matrix. Figure 4.13 which shows the stress-strain data for Nylon 6/6 reinforcements in a more conventional presentation format also illustrates the same point.

## 4.4. Discussion and Evaluation of Results

The previous section presented the results from a total of 10 tests which were performed on dry, medium-dense Ticino sand reinforced with steel and Nylon 6/6 sheet inclusions. Based on the data collected in the APSR cell, the following characteristic features of load-transfer in planar, elastic inclusions can be identified.

---

<sup>3</sup> The lateral strain,  $\epsilon_{xx}$ , values for 270 mm long inclusion had to be computed from displacement measurements at only a single point close to the center of the specimen and hence, are less reliable.

#### 4.4.1. Outward Movements of Inclusion

Figures 4.4 and 4.9 show that during the initial phase of shearing, the reinforcing inclusion is forced out of the cell. The magnitude of this outward displacement from the rear wall of the cell is very small and ranges from less than a micrometer to about 30  $\mu\text{m}$  depending on the axial stiffness and length of the inclusion. Figure 4.14 shows the inclusion displacements occurring before feedback control is activated in the APSR experiments with Nylon 6/6 sheet inclusions of three different half-lengths,  $L/2 = 360, 180, \text{ and } 90 \text{ mm}$ . The figure clearly demonstrates that for a given type of reinforcement, the shorter the inclusion, the larger the outward movement. This result suggests that there are outward deformations at the rear wall of the cell during the initial phase of the test ( $R < 4$ ). Figure 4.15 shows one possible mechanism in which there is arching of lateral pressures in the soil around the mailbox opening, combined with compression of the lubrication layer<sup>4</sup> along the rear wall of the cell, such that small outward displacements of the soil can occur. The fact that a small amount of lateral strain,  $\epsilon_{xx}$ , accompanies the outward inclusion displacements also indicates a symmetrical pattern of deformations shown in Figure 4.15. This pattern of soil deformations is likely to prevail during the early phase of the test before being overwhelmed by the predominant mechanism of load-transfer associated with lateral deformations towards the front face of the cell. In other words, there is a plane of zero lateral deformations which initially forms within the cell (Figure 4.15). As the global pattern of shearing evolves, the plane of zero lateral displacement moves towards the rear wall of the cell and eventually coincides with it as presumed in the original design

---

<sup>4</sup> The rear wall of the cell is covered with two layers of silicone grease separated by 0.38 mm thick rubber sheet in order to minimize friction.

concept (Figure 2.2). The amount of stretching and hence, the outward movement in the reinforcement under such a condition would depend on: a) the initial position of the plane of zero lateral deformation with respect to the tip of the reinforcement (Figure 4.15), and b) axial stiffness of the inclusion.

The scenario outlined in this section has not been confirmed by independent measurements of strain distribution inside the cell. The current radiographic equipment (refer to Larson, 1992) can only resolve internal strains to a precision of approximately 0.2%. It should be emphasized that the magnitude of maximum inclusion movement is indeed very small compared to thickness of the inclusion or the mean soil particle size. Although the mechanism responsible for the outward movement of inclusion in the beginning of APSR test is not fully resolved, under the current test procedures, these movements are unlikely to affect the subsequent measurements of tensile load-transfer.

#### 4.4.2. Initiation of Tensile Load-Transfer

The APSR cell is designed such that the reinforcement is oriented parallel to the direction of the minor principal strain,  $\epsilon_{xx}$ , during shearing. As the soil specimen expands laterally, shear stresses that are developed at the soil-reinforcement interface induce tensile stresses in the reinforcement. The rate of lateral strain development in the APSR test varies in a non-linear fashion throughout shearing and can be related to changes in the shear stress ratio. The data from APSR tests on unreinforced and reinforced Ticino Sand (Figures 4.7, 4.12) indicate that there are two distinct regimes of lateral strains in the APSR cell. During the initial phase of shearing, the lateral strain measured at specimen boundary,  $\epsilon_{xx}$ , is negligible. During this phase of the test, the soil contracts at about the same rate as axial strain,  $\epsilon_{yy}$ , and hence, deforms one dimensionally. Tensile (outward) lateral strains are first observed when  $2.0 < R < 3.0$  and increase rapidly thereafter (Figure 4.12a). The transition point between the two



regimes of lateral deformations described above can be related to overconsolidation of soil ( $OCR \approx 2.5$ ) due to application of a vacuum during sample preparation (refer to Section 3.3).

Figure 4.16 shows enlarged view of the initial portions of load pickup curves for the steel and Nylon 6/6 sheet inclusions. The stress ratio corresponding to the beginning of tensile load-transfer,  $R^*$ , ranges between 2.0 to 4.3. The lower values are typical for long, stiff inclusions while the higher values are obtained for extensible and/or short inclusions. A large part of this variation in  $R^*$  reflects the true soil-reinforcement interaction behavior; the magnitude of shear strains in soil required to induce tensile load in a stiff inclusion is relatively small. However, as the discussion in the remainder of this section demonstrates, some of the variation in  $R^*$ , especially the dependence of  $R^*$  on inclusion length, can be attributed to additional factors.

In general, the value of  $R^*$  in a test is slightly larger than the stress ratio at which tensile lateral strains in the soil,  $\epsilon_{xx}$ , are first observed at the specimen boundary. Ideally, one would expect the initiation tensile load pickup in the inclusion to coincide with the onset of lateral deformations in the soil matrix, since the axial strains in the inclusion due to application of normal stress,  $\sigma_{yy}$  (Poisson's effect), are negligible<sup>5</sup>. The results of regression analyses summarized in Table 4.4 show that, there is a fairly linear relationship between inclusion tensile load,  $F_R$ , and the average lateral strain in the soil matrix,  $\epsilon_{xx}$ . For inclusions with large values of  $R^*$  (short inclusions), the regression lines show an offset strain,  $\epsilon_{xx}^*$ , ranging from -0.13% to -0.15%. These

---

<sup>5</sup> For relatively extensible Nylon 6/6 inclusions, reinforcement strain due to Poisson's effect is approximately 0.001% at  $R = 3$ .

data indicate that for short inclusions, a small amount of tensile lateral strain occurs within the soil before tensile loads develop within the inclusion. This is most likely due to small non-uniformity in the distribution of lateral strains within the soil specimen. Although the lubrication technique employed in the APSR cell is very effective in reducing friction (Section 2.2), the cohesion of the lubricant acting along the plane strain faces of the specimen can increase the effective confinement for points away from specimen boundaries and contribute to some strain non-uniformity. Larson (1992) used radiography to measure internal strains within unreinforced soil specimens in the APSR cell and concluded that lateral strains did not vary with distance from the rear wall of the cell. However, a strain variation of the order of 0.1% is below the resolution threshold of the X-ray measurement technique employed by Larson (1992).

The observed values of the characteristic stress ratio,  $R^*$  for short inclusions may be corrected assuming that  $F_R$  versus  $\epsilon_{xx}$  plots (Figures 4.6 and 4.11) should ideally pass through the origin (i.e.,  $\epsilon_{xx}^* = 0$ ). Figure 4.12a shows that for Nylon 6/6 sheet reinforcements<sup>6</sup>, in the small strain range ( $\epsilon_{xx} < 0.1\%$ ), the gradient,  $dR/d\epsilon_{xx} \approx 10$ . Thus, an offset strain  $\epsilon_{xx} = -0.15\%$  corresponds to an increase in the stress ratio,  $\Delta R^* \approx 1.5$ . Table 4.5 summarizes the calculations which show that this correction reduces significantly the variation in  $R^*$  with inclusion length.

---

<sup>6</sup> Data for steel sheet inclusions also indicate similar values of gradient,  $dR/d\epsilon_{xx}$ .

### 4.4.3. Tensile Stress in Inclusion

#### 4.4.3.1. Influence of Inclusion Length

One of the main objectives of the APSR cell is to study the effects of inclusion length on load-transfer. As shown in Figures 4.5 and 4.10, the inclusion length affects both the magnitude of 'centerline' load,  $F_R$ , and the load-transfer gradient,  $dF_R/dR$ .

The centerline load increases significantly with the length of steel sheet inclusion; at an applied stress ratio,  $R = 7$ , the measured centerline loads range between  $F_R = 2.8$  kN/m at  $L/2 = 90$  mm to  $F_R = 23.5$  kN/m for  $L/2 = 360$  mm. In contrast, the tensile loads in Nylon 6/6 reinforcements show a smaller variation with inclusion length. At an applied stress ratio,  $R = 7$ , the measured centerline loads range between  $F_R = 1.75$  kN/m at  $L/2 = 90$  mm to  $F_R = 4.25$  kN/m for  $L/2 = 360$  mm. Figure 4.17 summarizes the effect of inclusion length on inclusion tensile stresses for steel and Nylon 66/ inclusions.

Larson (1992) obtained preliminary measurements of load-transfer for steel sheet inclusions using the prototype APSR device. A comparison in Figure 4.18 shows that load-transfer behavior of the steel inclusions measured in the refined APSR cell is qualitatively different from that reported by Larson. The centerline loads measured in the present research are generally lower at low values of the applied shear ( $R < 6$ ) but increase rapidly as the specimen is further sheared. For example at  $R = 7$ , the centerline loads are approximately 20% higher for the steel inclusion with half-length,  $L/2 = 360$  mm.

Figures 4.6 and 4.11 show inclusion tensile loads as functions of the average lateral strain,  $\epsilon_{xx}$ , in soil matrix for steel and Nylon 6/6 sheet reinforcements respectively. The figures show that for a given type of reinforcement, the apparent stiffness,  $E_a = dF_R/d\epsilon_{xx}$ , increases with length inclusion length. For steel sheet reinforcements, the measured apparent stiffness range between,  $E_a = 2.8$  kN/m at  $L/2 =$

90 mm to  $E_a = 23.5$  kN/m for  $L/2 = 360$  mm (Table 4.4). Similarly for Nylon 6/6 sheet reinforcements, ranges from  $E_a = 2.8$  kN/m at  $L/2 = 90$  mm to  $E_a = 23.5$  kN/m for  $L/2 = 360$  mm. It should be noted that the apparent stiffness,  $E_a$ , depends on factors such as the specimen geometry, and boundary conditions. The values of  $E_a$  reported here, therefore, are specific to the APSR cell and cannot be easily generalized. The concept of apparent stiffness is introduced here only to demonstrate the fact that a linear relationship exists between tensile stresses in the reinforcement and lateral strains in the soil matrix. In the future it may be possible to exploit this relationship for interpretation reinforcement stresses in a reinforced soil wall from measurements of facing panel movements.

#### 4.4.3.2. Effects of Inclusion Stiffness

Steel and Nylon 6/6 inclusions were selected to represent the upper and lower limits of axial stiffness of the reinforcing materials used in practice. Figure 4.17 shows that for a given value of the applied stress ratio,  $R$ , tensile load in a Nylon reinforcement is a factor of four to five times smaller compared to the load in the steel reinforcement of the same length. The figure also shows that the increase in the centerline load with inclusion length is more pronounced in the stiffer steel sheet reinforcements. In fact, increasing the half-length of Nylon inclusions beyond 180 mm has only a marginal effect on the magnitude of 'centerline' load,  $F_R$ . At an applied stress ratio,  $R = 8$ , doubling the inclusion length from  $L/2 = 180$  mm to  $L/2 = 360$  mm, results in only about 20% increase in the inclusion load. The effects of inclusion stiffness on the magnitude of centerline load and its variation with the inclusion length are subsequently explained within the theoretical framework of the shear-lag analysis (refer to Section 4.5.3).

Table 4.3 summarizes the centerline inclusion loads and strains for both steel and Nylon 6/6 sheet reinforcements with half-length  $L/2 = 360$  mm at various stages of shear. The data show that although the stiffness per unit width,  $J$ , of Nylon 6/6 reinforcement is 50 times lower than that of steel reinforcement, the maximum strains in Nylon inclusions are only about 10 times larger than steel inclusions. The ratio of average lateral strains in the soil,  $\epsilon_{xx}$ , to centerline strains in steel sheet inclusions is approximately 7.5 while the same ratio for Nylon 6/6 inclusions is about 3.0. The ratio of lateral strains in the soil to strains in reinforcing inclusions is a function of geometry and boundary conditions that are specific to the APSR cell and hence, the results lack general applicability. However, these results clearly show that the conventional assumption of strain compatibility which equates lateral strains in the soil mass with reinforcement strains is unrealistic.

#### 4.4.4. Effect of Inclusion on Behavior of Soil Matrix

The load-transfer in a reinforced soil composite increases the confining stresses within the zone of reinforcement. The effect of higher confinement is to increase the value of externally applied stress ratio,  $R = \sigma_1/\sigma_3$  for a given value of axial strain,  $\epsilon_{yy}$ , as shown in Figure 4.8a. Figure 4.19b shows a free body diagram illustrating the equilibrium of stresses for a composite soil element. Equation 4.1 (Section 4.5.1) indicates that differences between the externally applied confining stress,  $\sigma_3$ , and the average lateral stress in the matrix,  $\bar{\sigma}_{xx}^m$ , depends on the axial stress in the inclusion. The effectiveness of an inclusion in modifying the overall (global) stress-strain behavior of the reinforced soil composite, therefore, depends on the magnitude of the load carried by the inclusion. Since the magnitude of tensile loads in Nylon 6/6 reinforcements in the APSR cell is relatively low, the Nylon reinforcements have little or no effect on the stress-strain behavior of the sand matrix (Figures 4.12 and 4.13).

This is in contrast with the results for steel sheet reinforcements (Figures 4.7, 4.8) which carry approximately five times higher loads. The presence of steel reinforcements increases the shear modulus of the soil by as much as 35%, and tends to inhibit lateral strains and dilatancy in the soil.

## 4.5. Comparison with Shear-lag Analysis

The shear-lag analysis developed by Abramento and Whittle (1993) expresses the reinforcement stresses as closed form functions of the inclusion geometry (thickness, spacing, and length) and elastic properties of the constituent materials (i.e., soil matrix and reinforcement). Since the analysis provides a simple and direct method for estimating the stress distribution within the inclusion and reinforced soil mass at working stress levels, it is used as a framework for interpreting the APSR results.

### 4.5.1. Review of Shear-lag Analysis

This section summarizes the important assumptions and solutions, and presents typical results for a range of soil and reinforcement properties. For a complete discussion of various steps in the analysis and derivations of the equations, the reader is referred to Abramento (1993).

The shear-lag formulation was first used in the mechanics of fiber reinforced composites by Cox (1952) and subsequently extended by numerous authors (refer to Abramento, 1993). Figure 4.19a shows the idealized geometry and boundary conditions considered in the present analysis. The reinforcing inclusion has thickness  $f$  and length  $L$ , and is embedded in the soil matrix that has overall vertical dimension,  $m+f$ . The soil specimen is initially subjected to uniform boundary tractions  $\sigma_1$  and  $\sigma_3$ .

The orientation of the inclusion is parallel to the minor principal stress,  $\sigma_3$ . The soil is sheared in plane strain compression by increasing the major principal stress,  $\sigma_1$ . The shear-lag analyses simplify the soil-reinforcement interaction problem by assuming a simplified deformation field within the soil such that the tensile stresses within the reinforcement can be solved from equilibrium considerations. The other assumptions made in the present analysis are:

1. The soil matrix and reinforcing material behave as linear, isotropic, and elastic materials with properties  $G_m$ ,  $\nu_m$  and  $E_f$ ,  $\nu_f$ , respectively.
2. The soil-reinforcement interface is purely frictional and characterized by a constant angle of friction  $\delta$ .
3. There is no axial stress acting at the ends of the inclusion as the inclusion is thin and is not bonded to the soil matrix.
4. The axial stresses in the soil matrix and in the inclusion are functions of the  $x$  coordinate only.

Figure 4.19b shows the equilibrium of the stresses considered for the soil matrix with a planar reinforcing inclusion. Based on assumption 4, the externally applied stress,  $\sigma_3$ , is balanced by the tensile stress carried by the inclusion ( $\sigma_{xx}^f$ ) and an 'average' lateral stress in the matrix ( $\bar{\sigma}_{xx}^m$ ) such that the overall horizontal equilibrium can be written as:

$$\bar{\sigma}_{xx}^m = \sigma_{xx}^m = a \sigma_{xx}^f + \sigma_3 (1 + a) \quad (4.1)$$

where,  $a = f/m$  is the 'inclusion ratio'. The sign convention assumes that compressive stresses are positive in the soil matrix, while the axial stress in the inclusion,  $\sigma_{xx}^f$ , is positive in tension. The normal stress acting at the inclusion interface can be determined directly from equilibrium in the vertical direction:

$$\sigma_{yy}^i = \sigma_1 + \frac{m}{4} \frac{d\sigma_{xy}^i}{dx} = \sigma_1 + \frac{mf}{8} \frac{d^2\sigma_{xx}^f}{dx^2} \quad (4.2)$$

In order to apply the shear-lag formulation, it is necessary to relate stresses and strains in the direction of the inclusion. This is accomplished by decoupling the normal and shear stresses acting in the soil matrix. Decoupling divides the soil matrix into two zones: a) an inner 'shear spring' layer of characteristics lateral dimension,  $\bar{m}$ , adjacent to the inclusion, in which only shear stresses occur, and b) an outer layer having the same dimension as the physical model (i.e.,  $m$ ) with normal stresses acting in  $x$  and  $y$  directions. The characteristic dimension,  $\bar{m}$ , is obtained by equating the elastic strain energy in this assumed shearing zone, with the total elastic strain energy of the physical system (Abramanto, 1993). Using this approach, the axial stress in the inclusion ( $\sigma_{xx}^f$ ) is obtained directly from equilibrium:

$$\frac{d^2\sigma_{xx}^f}{dx^2} - K_1 \sigma_{xx}^f + K_2 \sigma = 0 \quad (4.3a)$$

where 
$$K_2 \sigma = K_2^1 \sigma_1 + K_2^3 \sigma_3 \quad (4.3b)$$

with boundary conditions:

$$\text{for } x = 0, L \quad : \quad \sigma_{xx}^f = 0 \quad (4.4a)$$

$$\text{for } x = L/2 \quad : \quad \frac{d\sigma_{xx}^f}{dx} = 0 \quad (4.4b)$$

The coefficients  $K_1$ ,  $K_2^1$ , and  $K_2^3$  are constants, defined in terms of the material properties and geometry of the soil matrix and the inclusion (Figure 4.19a):

$$K_1 = \frac{6}{m f} \frac{\left[ (1 - \nu_m) a + 2 \frac{G_m}{E_f} (1 + \nu_f) (1 - \nu_f) \right]}{\left[ 1 + \frac{1}{4} \nu_m - \frac{3}{2} \frac{G_m}{E_f} (1 + \nu_f) \nu_f \right]} \quad (4.5a)$$



$$K_2^1 = \frac{6}{m f} \frac{\left[ v_m - 2 \frac{G_m}{E_f} (1 + v_f) v_f \right]}{\left[ 1 + \frac{1}{4} v_m - \frac{3}{2} \frac{G_m}{E_f} (1 + v_f) v_f \right]} \quad (4.5b)$$

$$K_2^3 = -\frac{6}{m f} \frac{\left[ (1 - v_m) (1 + a) \right]}{\left[ 1 + \frac{1}{4} v_m - \frac{3}{2} \frac{G_m}{E_f} (1 + v_f) v_f \right]} \quad (4.5c)$$

The general solution for the axial stress in the reinforcement is obtained directly by solving equation 4.3 using boundary conditions expressed in equation 4.4 as:

$$\sigma_{xx}^f = \frac{K_2 \sigma}{K_1} \left[ 1 - \frac{\cosh \sqrt{K_1} \left( \frac{L}{2} - x \right)}{\cosh \sqrt{K_1} \frac{L}{2}} \right] \quad (4.6)$$

The maximum axial stress at the centerline of the inclusion ( $x = L/2$ ) is:

$$\sigma_{\max}^f = \frac{K_2 \sigma}{K_1} \left[ 1 - \operatorname{sech} \sqrt{K_1} \frac{L}{2} \right] \quad (4.7)$$

For a very long inclusion, where  $L/2 \rightarrow \infty$ , the maximum inclusion load is:

$$\sigma_{xx}^f (\infty) = \sigma_{\infty}^f = \frac{K_2 \sigma}{K_1} \quad (4.8)$$

Equation 4.8 shows that the maximum tensile stress in a long inclusion is controlled by three factors:

1. The shear stress mobilized in the soil matrix,  $\sigma_1/\sigma_3$
2. The relative stiffness of the inclusion and soil,  $E_f/G_m$
3. The volume ratio of the reinforcement to the soil matrix,  $a = f/m$ .

An extensive verification through comparisons with finite element analyses indicates that the centerline loads predicted by the shear-lag are accurate (within 5%) for a wide range of elastic material properties and geometry (Abramanto, 1993).

Figure 4.20a shows the distribution of the inclusion tensile stress,  $\sigma_{xx}^f$ , in the inclusion normalized by the major principal stress,  $\sigma_1$ , for inclusions with half-lengths,  $L/2 = 0.25, 0.5, \text{ and } 1.5 \text{ m}$ , at an external stress ratio,  $\sigma_1/\sigma_3 = 6$ . The calculations assume typical values for the material properties, spacing and thickness of reinforcement. In the zone close to the tip of the inclusion, the tensile stress accumulates ('builds up'), due to the development of interface shear stresses ( $\sigma_{xy}^i$ ). For inclusions longer than some critical length,  $L > L_I$ , a point is reached where there are no shear tractions at the soil-reinforcement interface and the axial stress becomes constant (i.e.,  $\sigma_{xx}^f = \sigma_{\infty}^f$ ). The 'maximum load-transfer ratio',  $\sigma_{\max}^f/\sigma_{\infty}^f$ , (Equation 4.7, 4.8) is a convenient parameter for characterizing the length,  $L_I$  of reinforcement which mobilizes maximum axial stress in the inclusion. The 'critical length',  $L_I$  is primarily a function of stiffness ratio,  $E_f/G_m$ , and spacing and thickness of inclusion ( $m, f$ ). Figure 4.21 shows the maximum load-transfer ratio as a function of the inclusion length and stiffness ratio of typical soil reinforcement,  $10^2 \leq E_f/G_m \leq 10^5$  for an inclusion with thickness,  $f = 1 \text{ mm}$ , and the spacing,  $m = 0.5 \text{ m}$ . The pickup length ranges from  $L_I = 0.8 \text{ m}$  for a soft inclusion ( $E_f/G_m = 10^2$ ) up to  $3.2 \text{ m}$  for stiff reinforcements. The stiffness ratio also affects significantly the load-transfer for short inclusions (i.e.,  $L < L_I$ ).

Figure 4.20b shows the development of axial inclusion stress for a inclusion with half-length,  $L/2 = 0.5 \text{ m}$  as a function of applied stress ratio,  $\sigma_1/\sigma_3$  in the soil matrix. For a soil matrix with linear, isotropic properties, the lateral strains in the soil are controlled by the elastic Poisson's ratio,  $\nu_m$ . The analysis predicts that tensile stresses develop in the reinforcement only when  $\sigma_1/\sigma_3 > 1/K_0 = (1-\nu_m)/\nu_m$ .

The preceding results assume that there is no slip between the soil matrix and the reinforcing inclusion. The results in Figure 4.20c show the effects of the friction

angle,  $\delta$ , on the load-transfer for an inclusion of half-length  $L/2 = 0.5$ , at a stress ratio  $\sigma_1/\sigma_3 = 6$ . For the selected material properties and geometry, the interface slippage has very little influence on tensile stresses in the reinforcement for  $\delta \geq 17^\circ$  ( $\mu = \tan\delta \geq 0.3$ ). However, there are significant reductions in load-transfer when the friction ratio is artificially low ( $\mu = 0.1$ ,  $\delta = 7^\circ$ ). Further studies (Abramanto and Whittle, 1993) also show that, for practical values of interface friction,  $\delta = 10^\circ$ - $30^\circ$ , interface slippage has little effect on the expected load-transfer for a wide range of material properties and geometry.

#### 4.5.2. Selection of Input Parameters

Input parameters for shear-lag analyses of the APSR test include the elastic properties of soil ( $G_m$ ,  $\nu_m$ ) and reinforcement ( $E_f$ ,  $\nu_f$ ), and specified geometry ( $L$ ,  $f$ ,  $m$ ). The elastic material properties are determined from plane strain shear tests on the unreinforced sand in the APSR cell, and in-isolation uniaxial tension tests on the planar inclusions. The interface friction angle has very little effect on predictions of load-transfer when the inclusion is oriented parallel to the direction of minor principal stress (Abramanto, 1993).

Within the range of strains encountered in the APSR cell, both steel and Nylon 6/6 reinforcements can be treated as linearly elastic materials with the axial stiffness,  $E_f$ , obtained from in-isolation uniaxial tension tests. The shear-lag predictions of load-transfer are not very sensitive to the Poisson's ratio of reinforcement (Figure 4.22a) and any realistic value of  $\nu_f$  may be used. Table 4.6 lists the values of elastic constants selected for the steel and Nylon 6/6 reinforcements. The shear stress-strain behavior of medium-dense Ticino sand in the APSR cell (Figure 3.15) is clearly non-linear and representative values of elastic material parameters must be used as an approximation. The reader is referred to Section 3.4.4 for the equations used to obtain secant values of

shear modulus,  $G_m$ , and Poisson's ratio  $\nu_m$ , from unreinforced test data. Figure 4.23 shows the variation of secant  $G_m$  and  $\nu_m$  as functions of the applied stress ratio,  $R$ , for the base-case unreinforced test (APSR 24N) on medium-dense Ticino sand ( $D_r = 75\%$ ,  $\sigma_3 = 31$  kPa). The Poisson's ratio of unreinforced Ticino sand increases in a non-linear manner throughout shearing and approaches the upper limit for an elastic material,  $\nu_m = 0.5$  at stress ratio,  $R \approx 9$ . Figure 4.23b shows that shear modulus of the sand starts out at a very high value,  $G_m > 30$  MPa, and decreases rapidly to  $G_m < 5$  MPa as the soil specimen is sheared.

The simplest way of approximating non-linear soil behavior by a set of linear elastic material parameters is to assume constant values of secant shear modulus and Poisson's ratio. Larson (1992) and Abramanto (1993) used an average value of shear modulus corresponding to 50% of failure strain,  $G_{m50} = 6000$ , and Poisson's ratio,  $\nu_m = 0.31$ , based on the initial slope of the lateral strain curve to represent the sand behavior. However, this simple approximation does not perform satisfactorily for the entire range of inclusion lengths and materials employed in the present research program. Figure 4.22b compares the shear-lag predictions of centerline loads using  $\nu_m = 0.2, 0.3,$  and  $0.4$  with the APSR measurement of load-transfer for 270 mm long steel sheet inclusion. The predictions were obtained using a constant value of soil modulus,  $G_m = 6000$  kPa since the shear-lag analysis is not very sensitive to the value of  $G_m$  for the case of steel reinforcements where the stiffness ratio,  $E_f/G_m \approx 35000$ . The figure shows that the predictions using constant values of Poisson's ratio of soil matrix,  $\nu_m$ , do not capture the essential characteristics of the load-transfer behavior. The predictions based on Poisson's ratio,  $\nu_m = 0.3$  describe realistically the initial portion of the experimental load pickup, but underpredict inclusion loads at high values of applied stress ratio,  $R$ . Results obtained using  $\nu_m \approx 0.4$  provide a good fit for  $R > 7$  but overestimate significantly the measured stresses in the reinforcement at  $R < 5$ .

Since the secant values of shear modulus and Poisson's ratio of Ticino sand vary continuously throughout the test, it is proposed that more realistic predictions of load-transfer can be obtained by using different values of  $G_m$  and  $\nu_m$  in the shear-lag calculations of inclusion load at each point along the load-transfer curve in Figure 4.22b. The non-linearity in soil behavior is modeled by expressing  $G_m$  and  $\nu_m$  as functions of the applied stress ratio,  $R$ , using a fourth order polynomial:

$$G_m(R) = X_0 + X_1 R + X_2 R^2 + X_3 R^3 + X_4 R^4 \quad (4.9)$$

where  $X_0, X_1, \dots$  are polynomial coefficients. A similar expression is used for  $\nu_m(R)$ . Table 4.7 lists the polynomial coefficients obtained by performing regression analyses on results of the base-case test on unreinforced, medium-dense Ticino sand (Figure 4.23).

The use of secant material parameters obtained from unreinforced tests to predict behavior of reinforced soil implies that a uniform state of stress exists throughout the soil specimen. However, in reinforced soil experiments, the stress distribution is non-uniform. Tensile stresses in the inclusion cause increased confining pressures locally within the soil matrix and hence, reduce the effective value of stress ratio,  $R$ , within the specimen. Therefore, it is reasonable to use an average value of the effective stress ratio,  $\bar{R}$ , in Equation 4.9 such that:

$$\bar{R} = \beta R \quad (4.10)$$

where the coefficient  $\beta < 1.0$ . The procedure for computing inclusion tensile stress at a given value of the applied stress ratio,  $R$ , consists of three steps: a) calculate the effective stress ratio,  $\bar{R}$  using Equation 4.10, b) obtain elastic parameters,  $G_m$  and  $\nu_m$ , for the soil matrix using  $\bar{R}$  and Equation 4.9, and c) use these values of  $G_m$  and  $\nu_m$  in the shear-lag calculations (Equations 4.5 to 4.7) to compute inclusion stresses.

In general, for a given set of soil and reinforcing material properties and geometry, increasing the value of  $\beta$  increases tensile stresses in the reinforcement and reduces the stress ratio,  $R^*$ , which marks the onset of tensile load-transfer. At present there is no procedure for predicting the value of  $\beta$  since the internal stress distribution is not known. This thesis assumes a constant value of  $\beta$  for each type of reinforcing material independent of stress ratio,  $R$ , and inclusion length. With this assumption, the coefficient  $\beta$  can be selected to provide the best overall fit with the experimental data. This approach has been used in the remainder of this section to 'predict' reinforcement loads in steel and Nylon 6/6 reinforcements using the shear-lag analysis.

### 4.5.3. Interpretation of APSR results

Figure 4.24 compares the shear-lag predictions with APSR measurements of centerline loads for steel sheet inclusions with half-lengths,  $L/2 = 90$  to 360 mm. These predictions were obtained using the secant distributions of shear modulus,  $G_m$ , and Poisson's ratio,  $\nu_m$ , shown in Figure 4.23 along with the coefficient  $\beta = 0.78$ . The predictions show excellent overall agreement with the measured loads for all but the shortest inclusion ( $L/2 = 90$  mm), where the analysis tends to underestimate the inclusion forces. The analyses also underpredict the inclusion forces for the initial portion of load-transfer ( $R < 5$ ). It should be noted that the predictions are based on data obtained from unreinforced tests. This is important as soil properties are functions of confining stress and development of  $F_R$  increases effective confinement within the soil. Therefore, one set of material properties is unlikely to perform equally well for the entire range of load-transfer.

Figure 4.25 replots the APSR data at specified stress levels as functions of the inclusion length. The measured data are in excellent agreement with the shear-lag predictions. These results demonstrate the applicability of the shear-lag analysis in

extrapolating results from small-scale laboratory tests to prototype (field) scale applications. For the relatively inextensible elastic steel sheet inclusions used in the APSR research, the analyses show that full load-transfer (i.e., stresses for field scale situation) is only achieved with inclusions of half-length,  $L/2 > 1$  m. The magnitude of tensile loads in a sufficiently long inclusion is expected to be more than two times the maximum loads measured for the longest inclusion ( $L/2 = 360$  mm) in the APSR cell.

Shear-lag predictions for the Nylon 6/6 reinforcements were obtained using the same set of elastic input parameters for unreinforced Ticino sand (Figure 4.23) along with the material properties of Nylon 6/6 reinforcements provided in Table 4.6. The coefficient  $\beta$  (Equation 4.10) shows some dependence on axial stiffness of the reinforcement. For the Nylon 6/6 reinforcement whose axial stiffness is about 50 times lower than that of the steel reinforcement, the best fit with experimental data is obtained for  $\beta = 0.65$ . This is about 16% lower than  $\beta$  value used for the predictions of load-transfer in the steel sheet inclusions.

Figure 4.26 compares the shear-lag predictions with APSR measurements of inclusion centerline loads for various lengths of Nylon 6/6 inclusions. The predictions again show good overall agreement with the measured loads, except for lower values of the applied stress ratio ( $R < 6$ ) where the analyses tend to under predict inclusion forces. Figure 4.27 highlights the influence of inclusion length on centerline loads at various values of the applied stress ratio,  $R$ . For the relatively flexible Nylon 6/6 reinforcements, the shear-lag analysis predicts, and APSR data confirms, that when the inclusion half-length,  $L/2 > 0.4$  m, the tensile loads approach an asymptotic value which is a function of the applied stress ratio.

## 4.6. Conclusions

This chapter describes measurements of load-transfer in the APSR cell for two types of reinforcements with planar (flat sheet) geometry, embedded in dry, medium-dense ( $D_r = 75\%$ ) Ticino sand. The main focus of this work is to study the influence of inclusion length and stiffness on magnitude of tensile stresses in the reinforcement. The results demonstrate that it is possible to obtain reliable and repeatable measurements of centerline tensile loads in the APSR cell provided the inclusion position is controlled at the exit point within  $\pm 1 \mu\text{m}$ . The APSR experiments provide comprehensive load-transfer measurements for steel and Nylon 6/6 reinforcements with four different half-lengths,  $L/2 = 90, 180, 270,$  and  $360$  mm that show the following:

1. The relationship between the inclusion centerline load,  $F_R$ , and externally applied stress ratio,  $R = \sigma_1/\sigma_3$ , is typically non-linear. The reinforcing inclusion is forced out of the APSR cell in the initial phase of the test and tensile load-transfer does not begin until the applied stress ratio,  $R \geq R^*$ . The characteristic stress ratio,  $R^*$ , varies from 2.5 to 4.3 depending on the length and axial stiffness of the reinforcement. The data indicate that dependence of  $R^*$  on inclusion length is primarily due to non-uniform lateral strain distribution in the device and the 'true' value of  $R^*$  is close to 2.5. The magnitude of the outward movement of the inclusion is a function of the length and axial stiffness of the reinforcement and ranges between 1 and  $50 \mu\text{m}$ .
2. The centerline tensile loads in steel sheet reinforcement increase significantly with length of the reinforcement. Loads measured for the longest steel inclusion ( $L/2 = 360$  mm) at  $R = 7$ , are almost 8 times the magnitude of those for the shortest inclusion ( $L/2 = 90$  mm). Although the tensile loads in the Nylon 6/6 inclusions tend to increase with the inclusion length, there



is only a small increase in loads measured for inclusions with half-lengths,  $L/2 > 180$  mm.

3. The magnitude of load-transfer increases with the axial stiffness of the reinforcement. Tensile loads in the steel sheet reinforcements range from  $F_R = 2.8$  kN/m at  $L/2 = 90$  mm to  $F_R = 23.5$  kN/m at  $L/2 = 360$  mm (at  $R = 7$ ) while less stiff Nylon 6/6 reinforcements carry loads between  $F_R = 1.75$  kN/m at  $L/2 = 90$  mm to  $F_R = 4.25$  kN/m at  $L/2 = 360$  mm at the same stress ratio. Thus, for a given stress ratio and inclusion length, there is a reduction by a factor of 1.6 to 5.5 associated with change in stiffness of  $J_{steel}/J_{Nylon} = 50$ . The maximum inclusion strains computed from measured loads range between 0.5% to 1.0% for the same range of inclusion stiffness.
4. The reinforcement tensile loads can be well described as linear functions of the average lateral strain in the sand specimen,  $\epsilon_{xx}$ , for both the steel and Nylon 6/6 reinforcements. For short inclusions ( $L/2 \leq 180$  mm) the relationship between  $F_R$  and  $\epsilon_{xx}$  shows a small offset strain which occurs before the onset of tensile load-transfer. This offset is due to non-uniform distribution of lateral strain within the soil specimen and hence, is an artifact of the device. The value of transfer gradient,  $dF_R/d\epsilon_{xx}$ , termed as 'apparent stiffness coefficient',  $E_a$  of an inclusion, increases with length of the inclusion. Typical values of the apparent stiffness coefficient,  $E_a$ , for the steel and Nylon reinforcements range between 7000 to 550 kN/m and 370 to 150 kN/m respectively.
5. The steel reinforcement causes a significant increase in the axial stiffness of the soil matrix. It also reduces the average lateral strain in the soil matrix,  $\epsilon_{xx}$ , for a given applied stress ratio,  $R$ , by as much as 50%. In contrast

reinforcement, the Nylon 6/6 reinforcement has virtually no effect on the externally measured shear behavior of Ticino sand in the APSR cell.

The shear-lag analysis developed by Abramento and Whittle (1993) provides closed-form expressions relating the magnitude and distribution of axial stresses in the reinforcement to: a) elastic properties of the soil and reinforcement, b) the length and spacing of the reinforcement, and c) the external level of shear stress in the soil matrix. The analysis provides a useful framework for interpretation of the APSR test results by giving physical insight into the constituent properties controlling the load-transfer in reinforced soil. The shear-lag predictions of load-transfer are not very sensitive to the Poisson's ratio of the reinforcement,  $\nu_f$ , but are found to vary greatly with the value of Poisson's ratio of the soil,  $\nu_m$ .

The comparisons described in this chapter demonstrate that the shear-lag analyses give reasonable agreement with the measured data for planar (sheet) reinforcements with stiffness ratio,  $320 < E_f/G_m < 35000$  when the non-linearity in the sand behavior is simulated by varying the secant shear modulus,  $G_m$ , and Poisson's ratio,  $\nu_m$ , of the matrix material. The shear-lag analysis shows that the tensile stresses measured in the longest steel sheet inclusion used in APSR test ( $L/2 = 360$  mm) represent half the stresses transferred to an infinitely long inclusion. However, for relatively flexible Nylon 6/6 reinforcement, the analysis shows that the stresses in the 360 mm long inclusion are very close to what may be expected in a field situation.

Test Number	Inclusion Half Length (mm)	No. of Strain Gages	Problem	Solution
APSR 5N	140	0	Poor b value control	New plane-strain reference frame.
APSR 8N	140	0	Low load pick up	Rigid reinforcement position control arm, New QuickBasic software.
APSR 11N	260	0		
APSR 12N	260	4	Poor b value control due to problems with side wall LVDT's.	Use of non-contact proximity sensors for side wall monitoring.
APSR 13N	260	4		
APSR 14N	260	4	Air leakage, low and erratic load pick up.	Introduction of airbag.
APSR 15N	260	4		
APSR 20N	260	4	Low load pick up.	Switch to rigid end plates for applying $\sigma_1$ .
APSR 21N	260	4		
APSR 22N	260	4		
APSR 25N	270	4	Low and irregular load pick up.	Full digital control of side walls and reinforcement implemented.
APSR 26N	270	4		
APSR 28N	270	4	Inward movement of the back wall detected.	Infrared sensor and related hardware developed.
APSR 29N	270	4		
APSR 30N	270	4		
APSR 31N	270	4	First test using the infrared sensor. Smooth load pick up.	

Table 4.1: Summary of APSR Proof Tests with Steel Sheet Reinforcements.

	Test Number	Inclusion Half- Length, L/2 (mm)	Offset Stress Ratio, R*	Maximum Slope, $dF_R/dR_{max}$
Steel	APSR 34N	360	2.05	7.75
	APSR 31N	270	2.60	6.45
	APSR 35N	270	2.65	6.52
	APSR 32N	180	2.98	3.52
	APSR 33N	90	4.30	2.15
Nylon 6/6	APSR 52N	360	3.18	1.83
	APSR 50N	270	3.11	1.73 <sup>†</sup>
	APSR 51N	270	2.95	1.69 <sup>†</sup>
	APSR 53N	180	3.50	1.64
	APSR 54N	90	4.25	0.85

<sup>†</sup> Based on average value of slope for  $5 < R < 8.5$ .

Table 4.2: Summary of APSR Tests with Steel and Nylon 6/6 Sheet Reinforcements.

Applied Stress Ratio, $R = \sigma_1/\sigma_3$	Steel Sheet Inclusions, $L/2 = 360 \text{ mm}; J = 52500 \text{ kN/m}$			Nylon 6/6 Sheet Inclusions, $L/2 = 360 \text{ mm}; J = 980 \text{ kN/m}$		
	Centerline Load $F_R$ , (kN/m)	Centerline Strain <sup>†</sup> $\epsilon_R$ , (%)	Lateral Strain in Soil, $\epsilon_{xx}$ (%) <sup>‡</sup>	Centerline Load $F_R$ , (kN/m)	Centerline Strain $\epsilon_R$ , (%) <sup>†</sup>	Lateral Strain in Soil, $\epsilon_{xx}$ (%) <sup>‡</sup>
4	5.15	0.01	-0.069	-	-	-
5	10.72	0.02	-0.147	1.3206	0.13476	-0.381
6	16.28	0.031	-0.227	2.7030	0.27582	-0.649
7	23.43	0.044	-0.334	4.2575	0.43444	-1.048
8	31.10	0.061	-0.456	5.9809	0.61030	-1.567
9	-	-	-	7.9818	0.81447	-2.345

<sup>†</sup> Reinforcement Centerline Strain,  $\epsilon_R = F_R/J$  (positive in tension).

<sup>‡</sup> Computed from external lateral displacement measurements (positive in compression).

Table 4.3: Magnitude of Reinforcement Centerline Loads and Strains for the Longest Planar Inclusions used in the APSR Cell.

Inclusion Half-Length L/2, (mm)	Steel Sheet Reinforcements			Nylon 6/6 Sheet Reinforcements		
	Apparent Stiffness, $E_a^\dagger$ (kN/m)	Offset Strain, $\epsilon_{xx}^*$ (%)	Least Squares Coefficient, $r^2$	Apparent Stiffness, $E_a^\dagger$ (kN/m)	Offset Strain, $\epsilon_{xx}^*$ (%)	Least Squares Coefficient, $r^2$
360	6795.2	0.007	0.9994	366.37	0.0167	0.9873
270	2218.6	0.018	0.9981	347.36	-0.0246	0.9989
180	1082.5	-0.065	0.9837	272.85	-0.1322	0.9944
90	531.61	-0.144	0.9966	155.3	-0.1520	0.9924

$\dagger$  Apparent Stiffness,  $E_a = dF_R/d\epsilon_{xx}$ .

Table 4.4: Linear Regression Analyses of Tensile Loads in Inclusions  
as Functions of the Average Lateral Strain.

Half- Length (mm)	Steel				Nylon 6/6			
	R*	$\epsilon_{xx}^*$	dR/d $\epsilon_{xx}$	R* <sub>corr.</sub>	R*	$\epsilon_{xx}^*$	dR/d $\epsilon_{xx}$	R* <sub>corr.</sub>
180	3.0	-0.065	12.0	2.22	3.5	-0.1322	9.5	2.24
90	4.3	-0.144	12.0	2.57	4.25	-0.1520	9.5	2.81

Table 4.5: Offset Strain Corrections for Short Inclusions in the APSR Cell.

Property	Sand	Steel	Nylon 6/6
Modulus	Distribution shown in Figure 4.22b	$E_f = 2.07 \times 10^8$ kPa	$E_f = 1.92 \times 10^6$ kPa
Poisson's Ratio	Distribution shown in Figure 4.22a	$\nu_f = 0.2$	$\nu_f = 0.2$
Lateral Dimension	m = 570 mm	f = 0.254 mm	f = 0.51 mm
Coefficient $\beta$	-	$\beta = 0.78$	$\beta = 0.65$

Table 4.6: Input Parameters for Shear-lag Predictions of Load Pickup in Steel and Nylon 6/6 Sheet Reinforcements.

Coefficients	Shear Modulus, $G_m$ (MPa)	Poisson's Ratio, $\nu_m$
$X_0$	125.05	-0.40571
$X_1$	-75.954	0.33849
$X_2$	17.968	-0.05536
$X_3$	-1.8679	0.0043388
$X_4$	0.071205	-0.00012357
Least Squares Coefficient, $r^2$	0.97864	0.99734

Table 4.7: Regression Coefficients Describing Variation of Elastic Material Parameters for Unreinforced Ticino Sand.



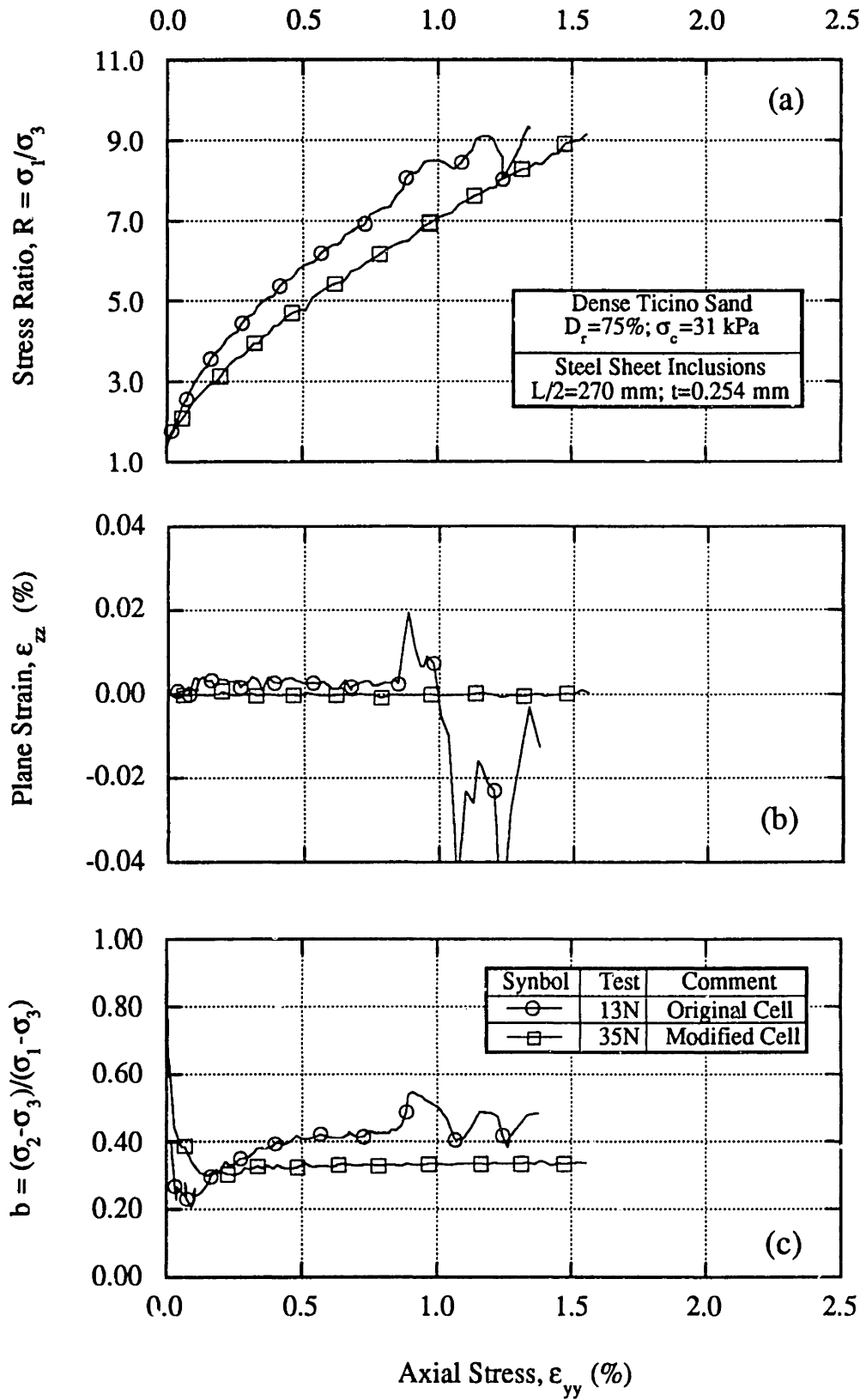


Figure 4.1: Comparison of Measurements Obtained from the Original and Modified APSR Cell.

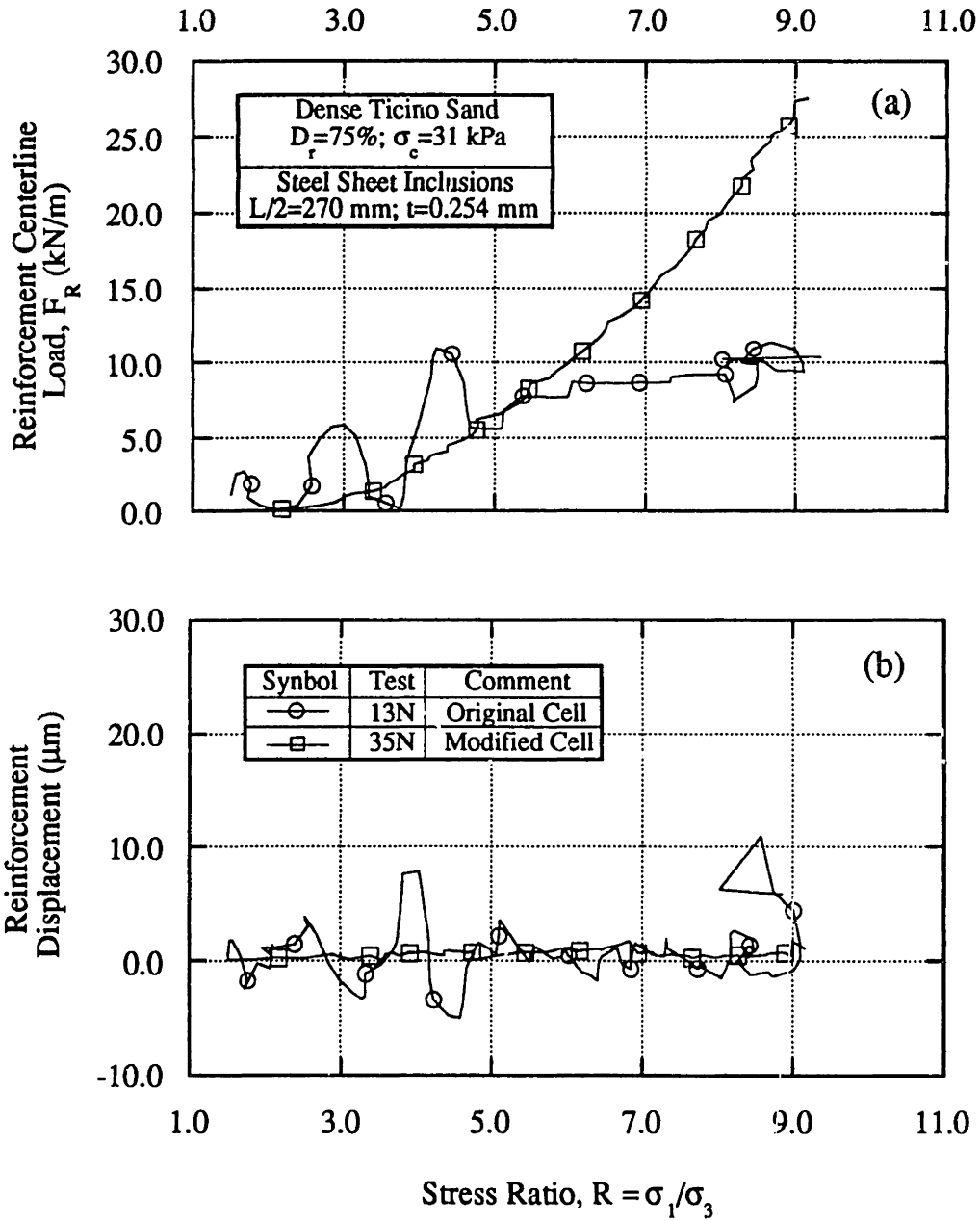
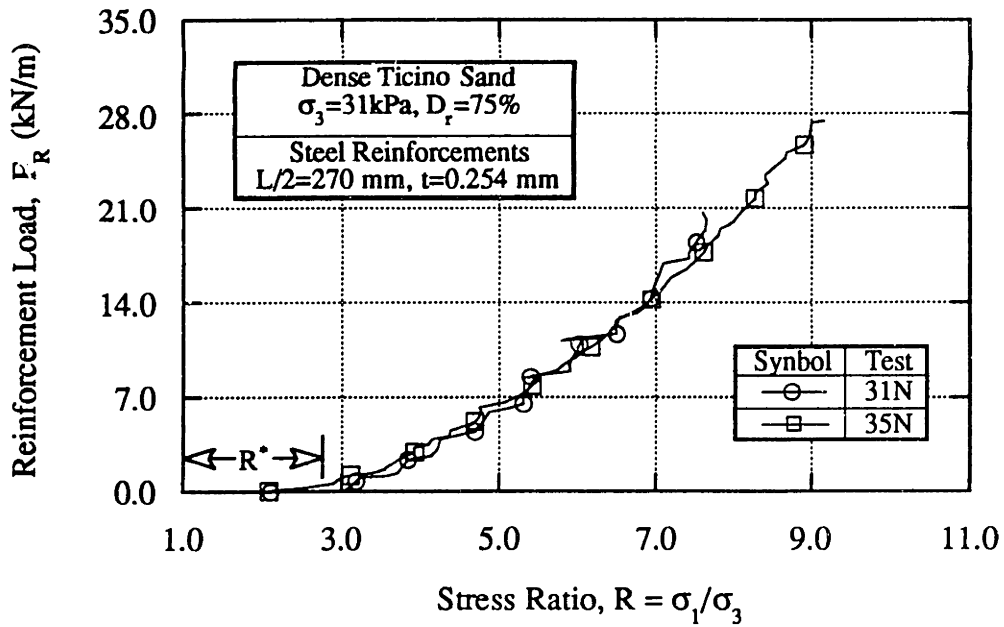
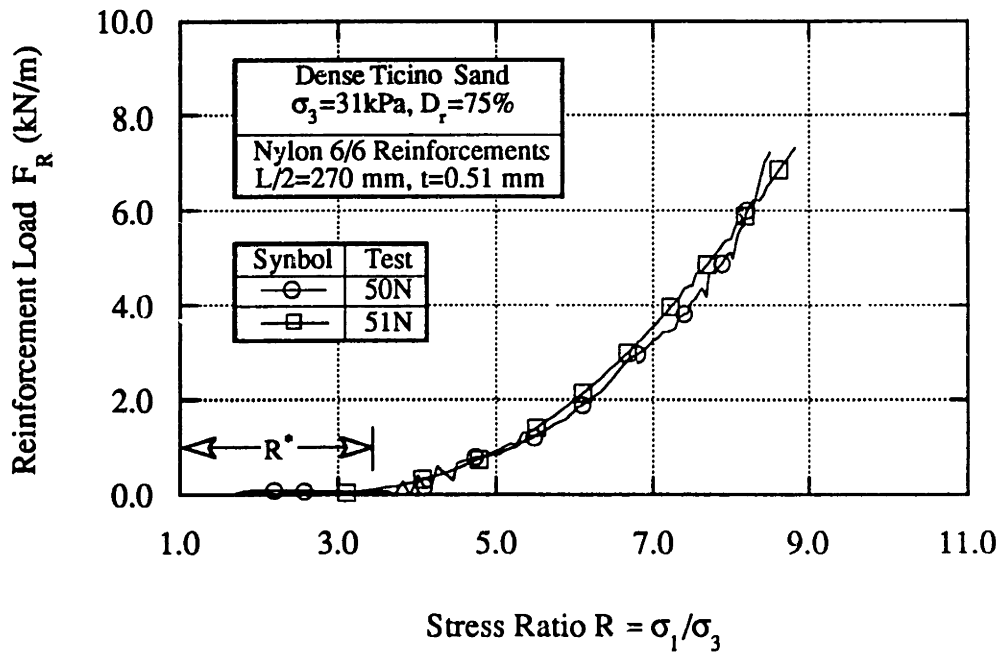


Figure 4.2: Improvements in the Control of Steel Reinforcement Position Using the Infrared Sensor.



(a) Load Pickup in Steel Sheet Reinforcement.



(b) Load Pickup in Nylon 6/6 Sheet Reinforcement.

Figure 4.3: Repeatability of Tensile Stress Measurements in the Improved APSR Cell.

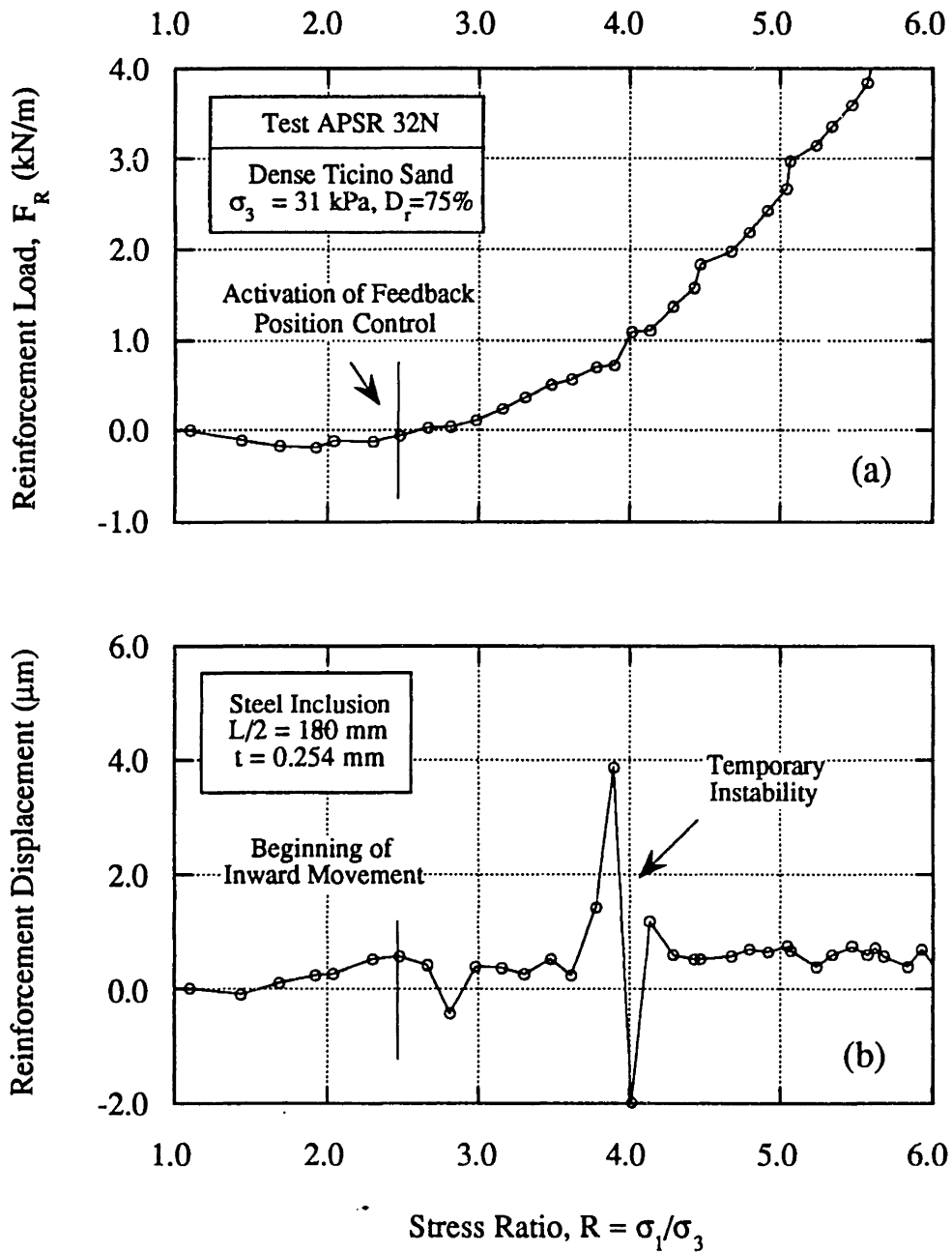


Figure 4.4: Initiation of Load Pickup in 180 mm Long Steel Sheet Inclusion.

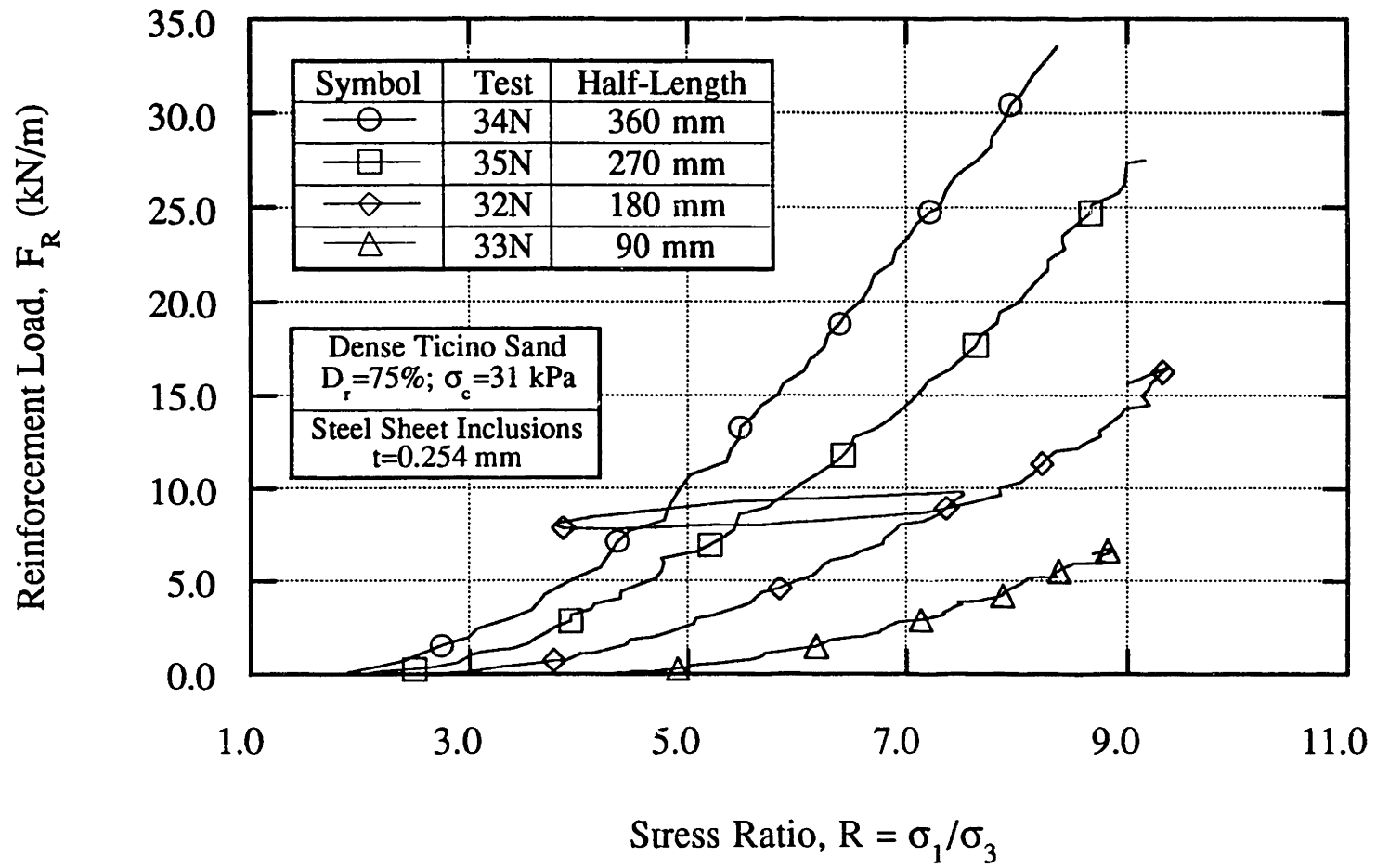


Figure 4.5: Effect of Inclusion Length on Tensile Stresses Measured for Steel Sheet Reinforcements in the APSR Cell.

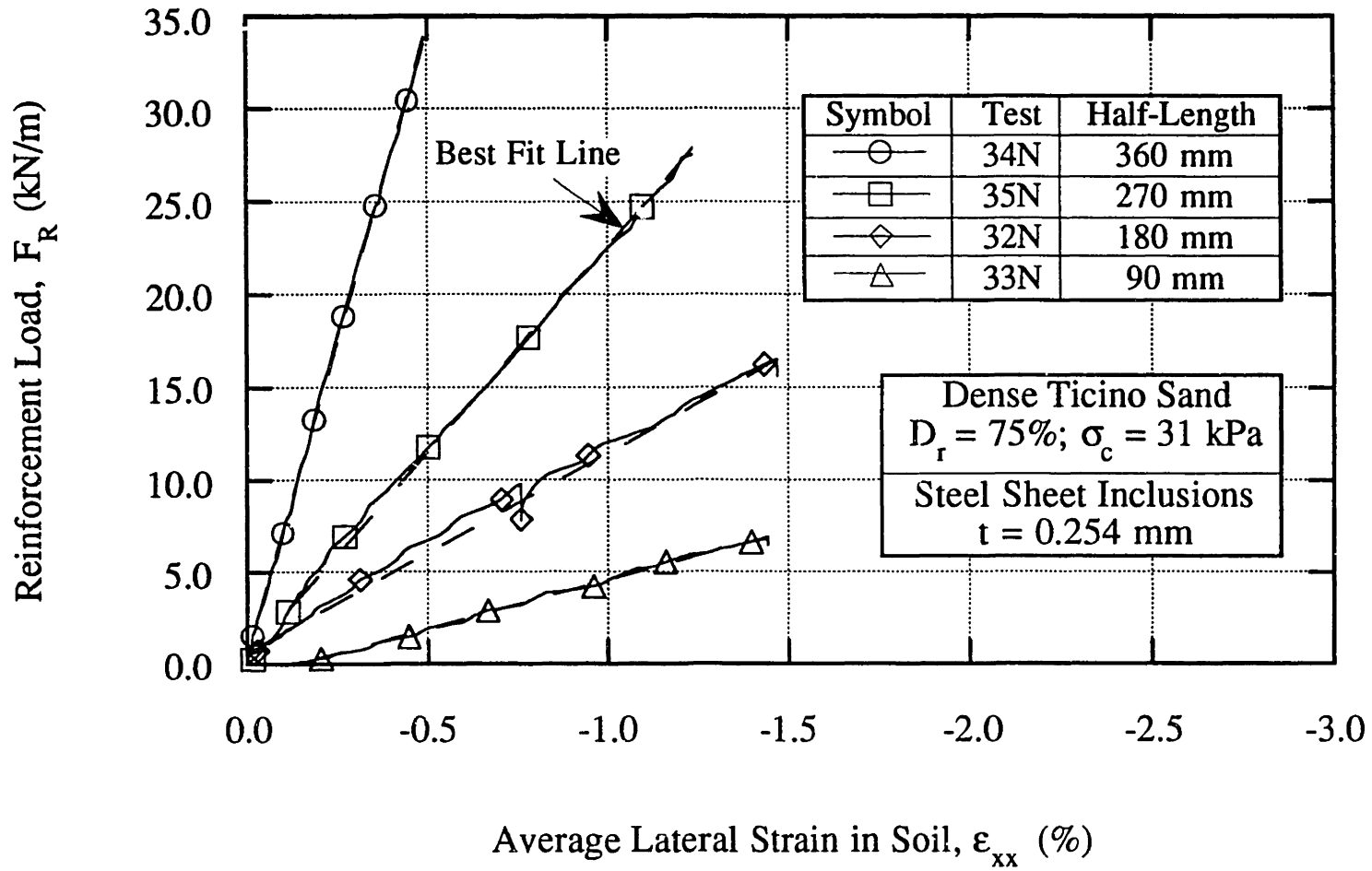


Figure 4.6: Relationship Between Inclusion Load and Average Lateral Strain in Soil Matrix for Steel Sheet Reinforcements.

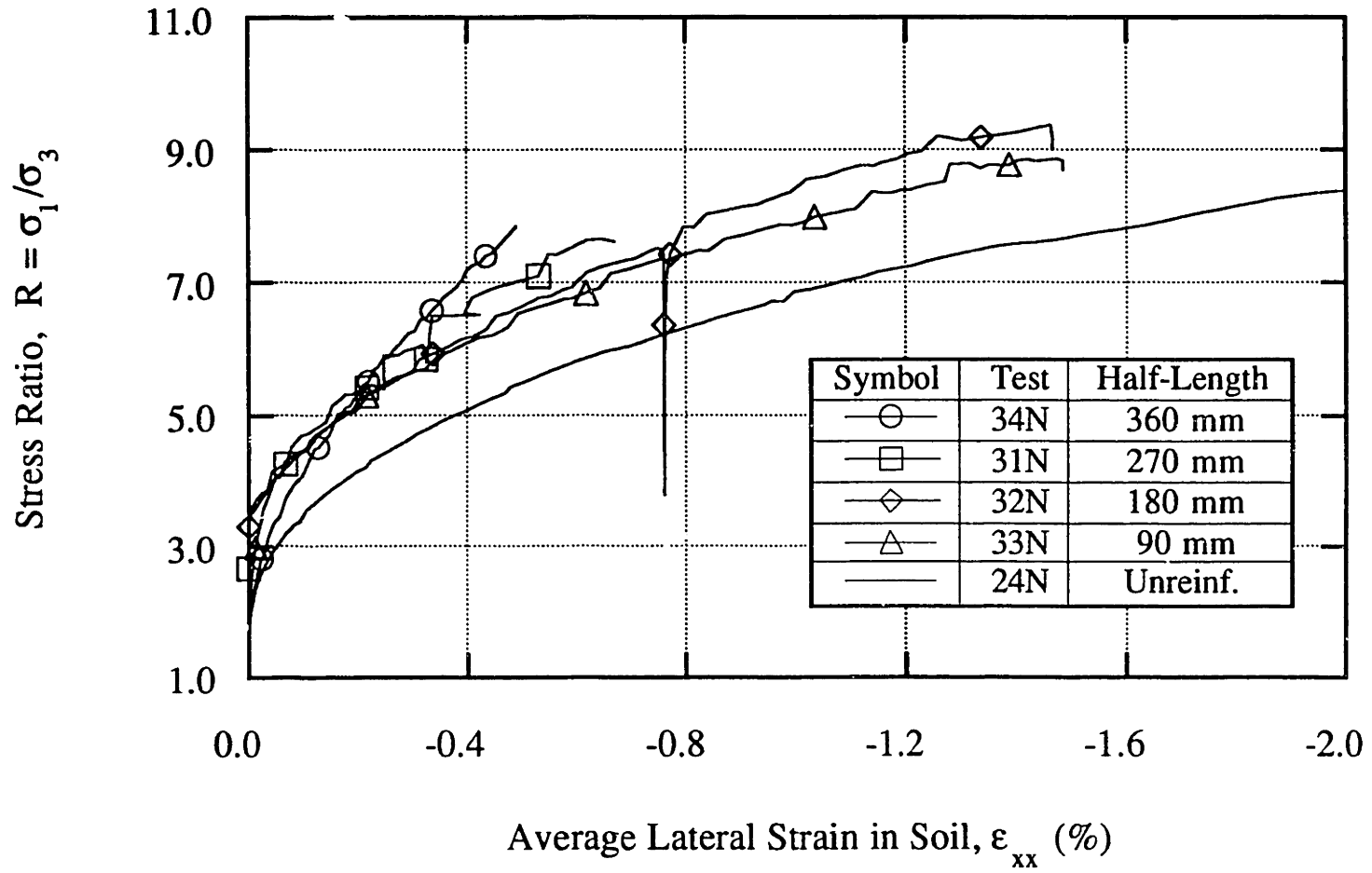


Figure 4.7: Effectiveness of Steel Sheet Reinforcement in Reducing Lateral Deformations in the Sand Matrix.

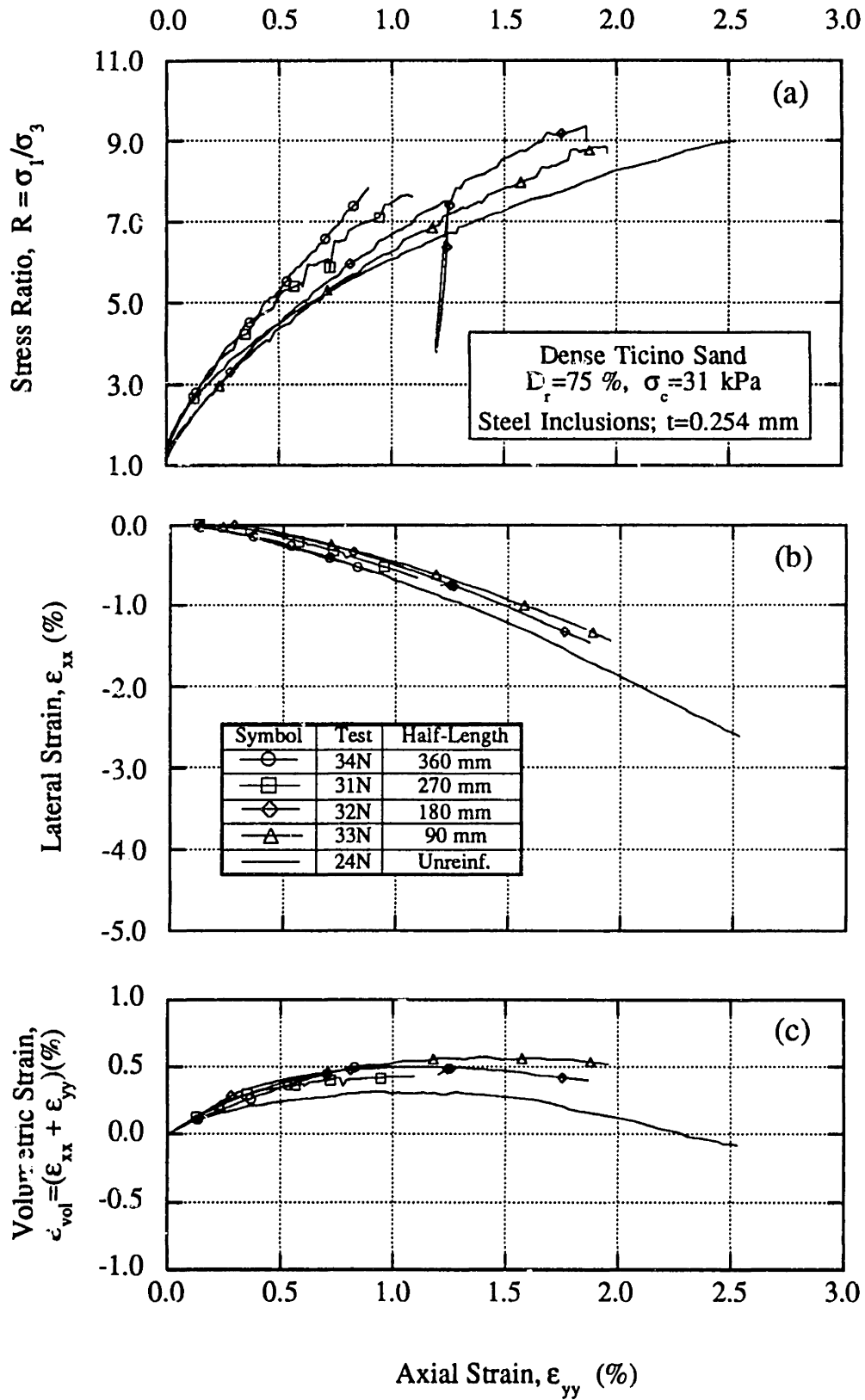


Figure 4.8: Externally Measured Shear Behavior of Dense Ticino Sand Reinforced with Steel Sheet Inclusions.



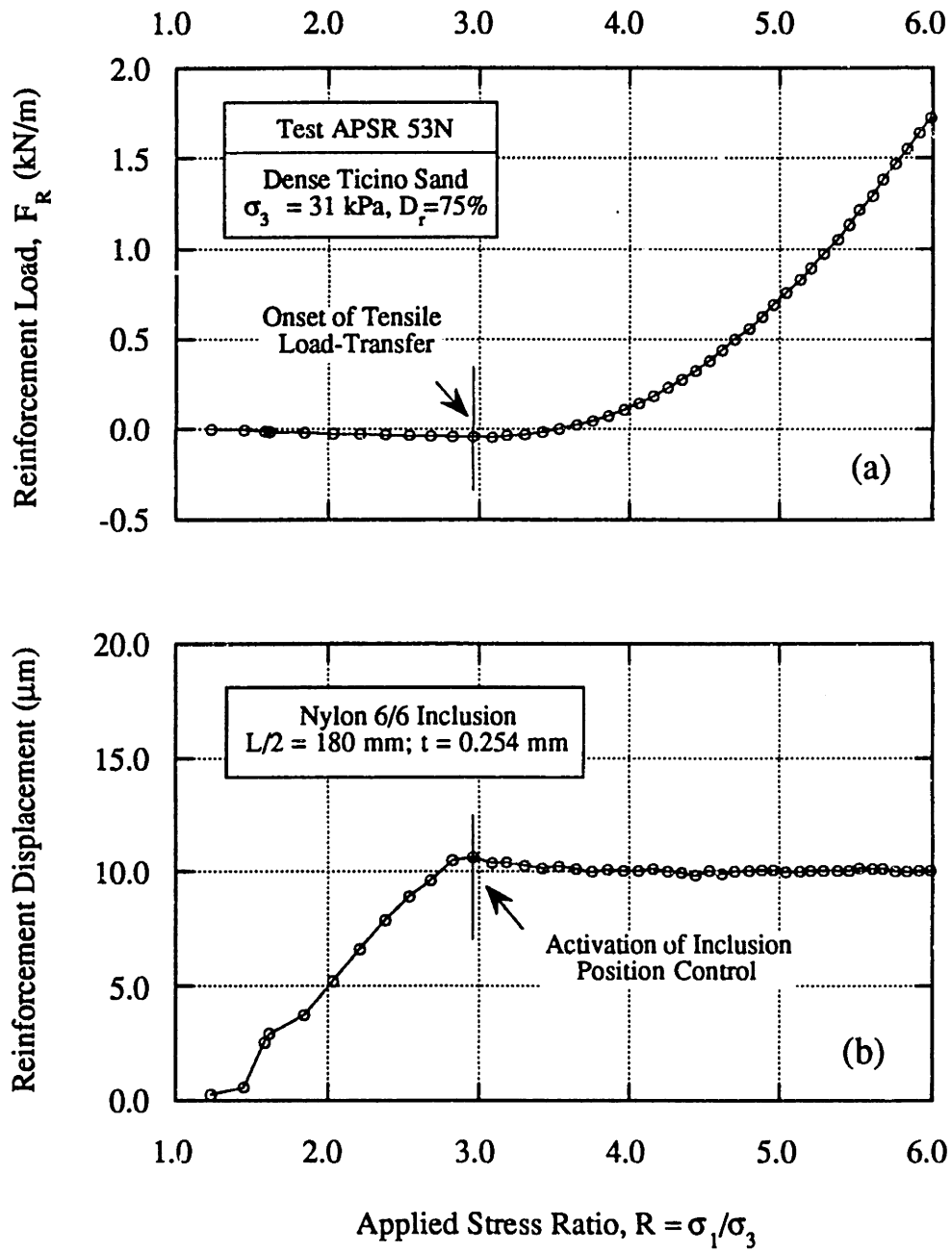


Figure 4.9: Beginning of Load Transfer in the Nylon 6/6 Sheet Reinforcement.

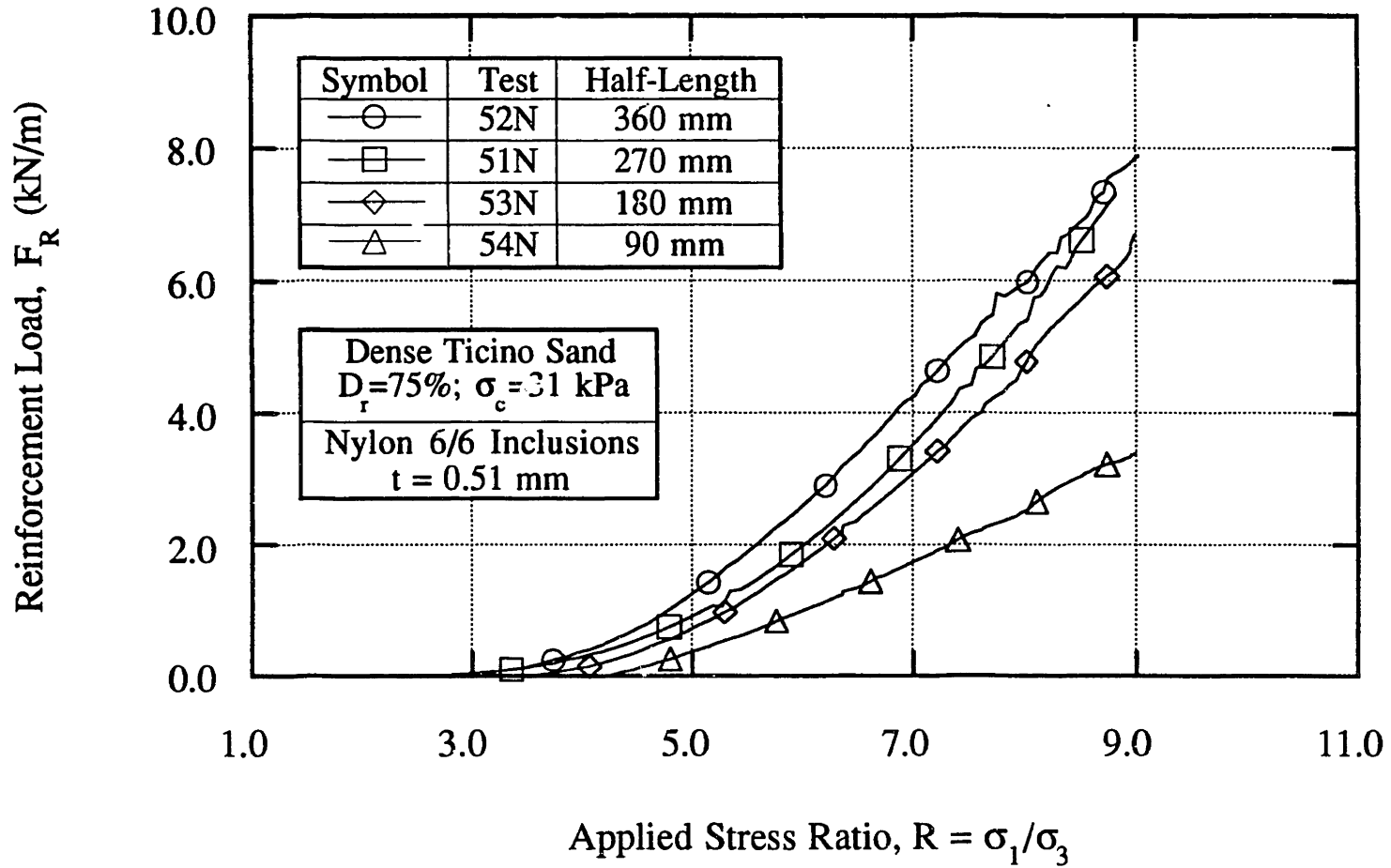


Figure 4.10: Effect of Inclusion Length on Tensile Stresses Measured for Nylon 6/6 Sheet Reinforcements in the APSR Cell.

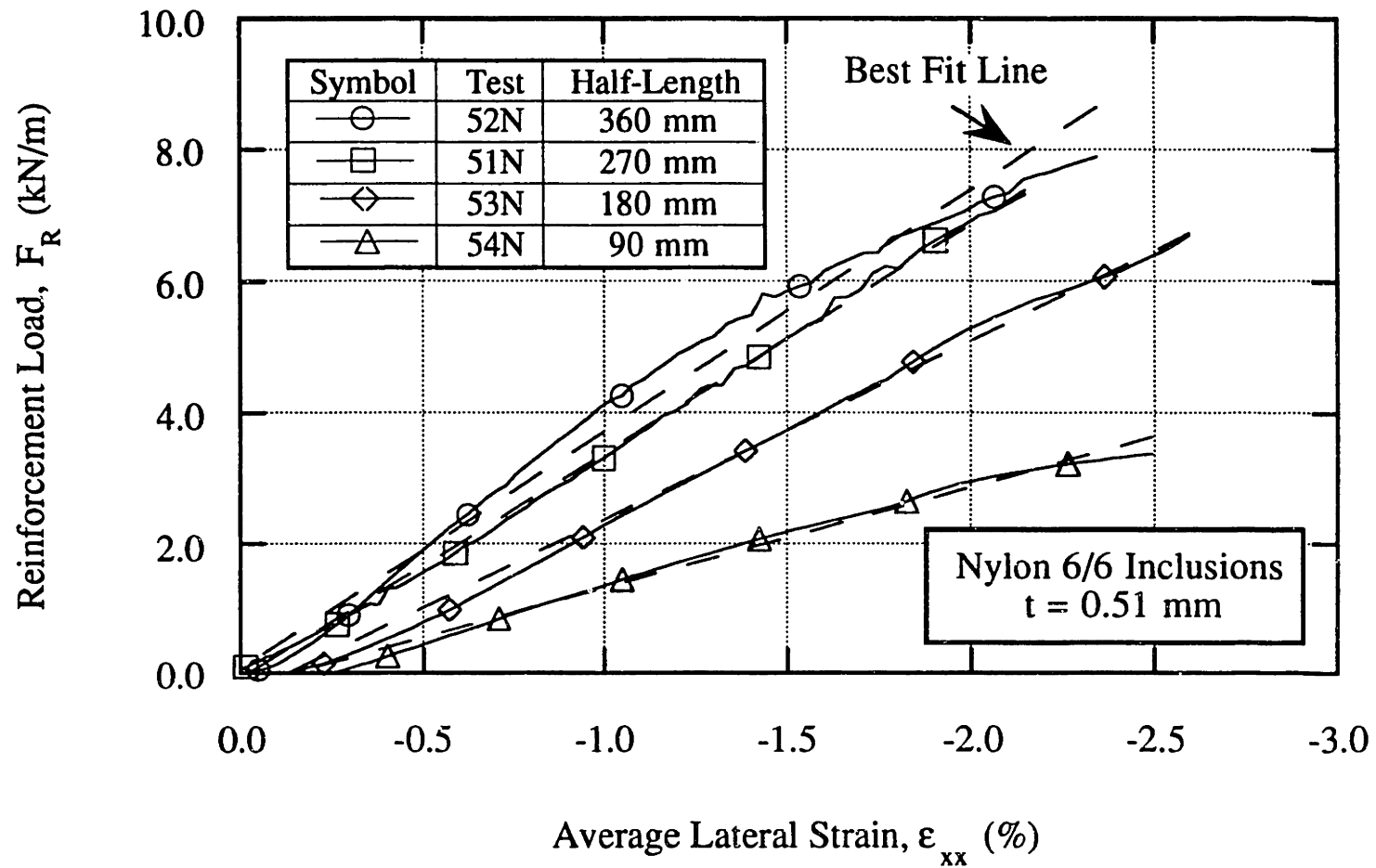


Figure 4.11: Inclusion Tensile Loads as Functions of the Average Lateral Strain in Soil Matrix for Nylon 6/6 Sheet Reinforcements.

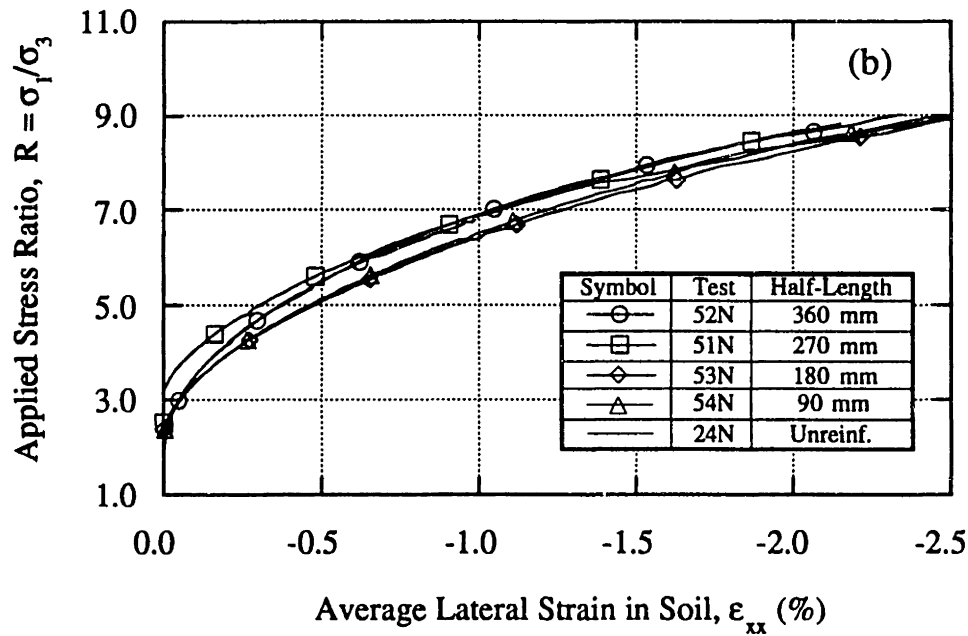
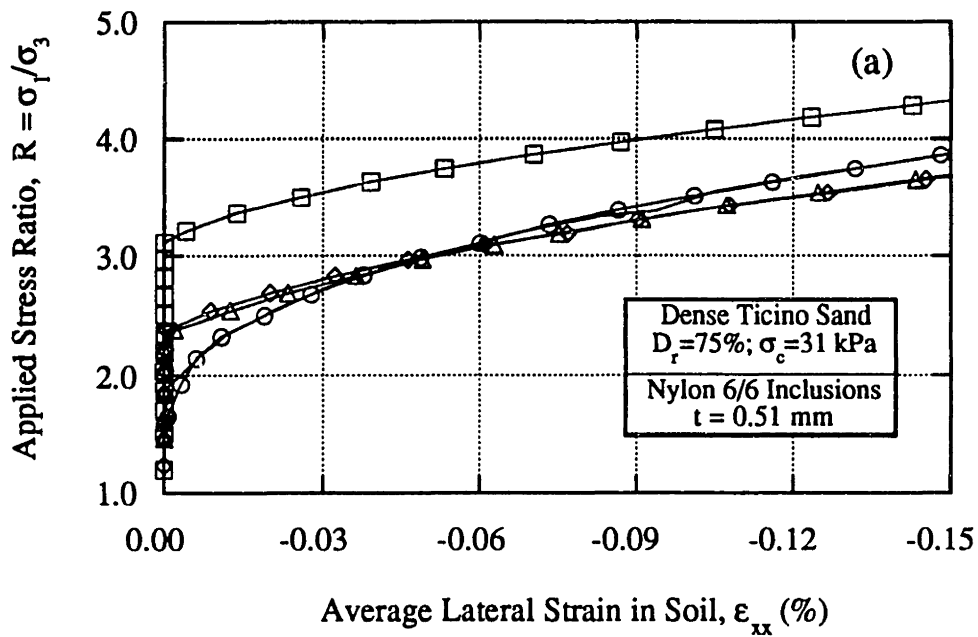


Figure 4.12: Relationship between the Applied Stress Ratio and Average Lateral Strain in APSR Tests on Nylon 6/6 Reinforcements.

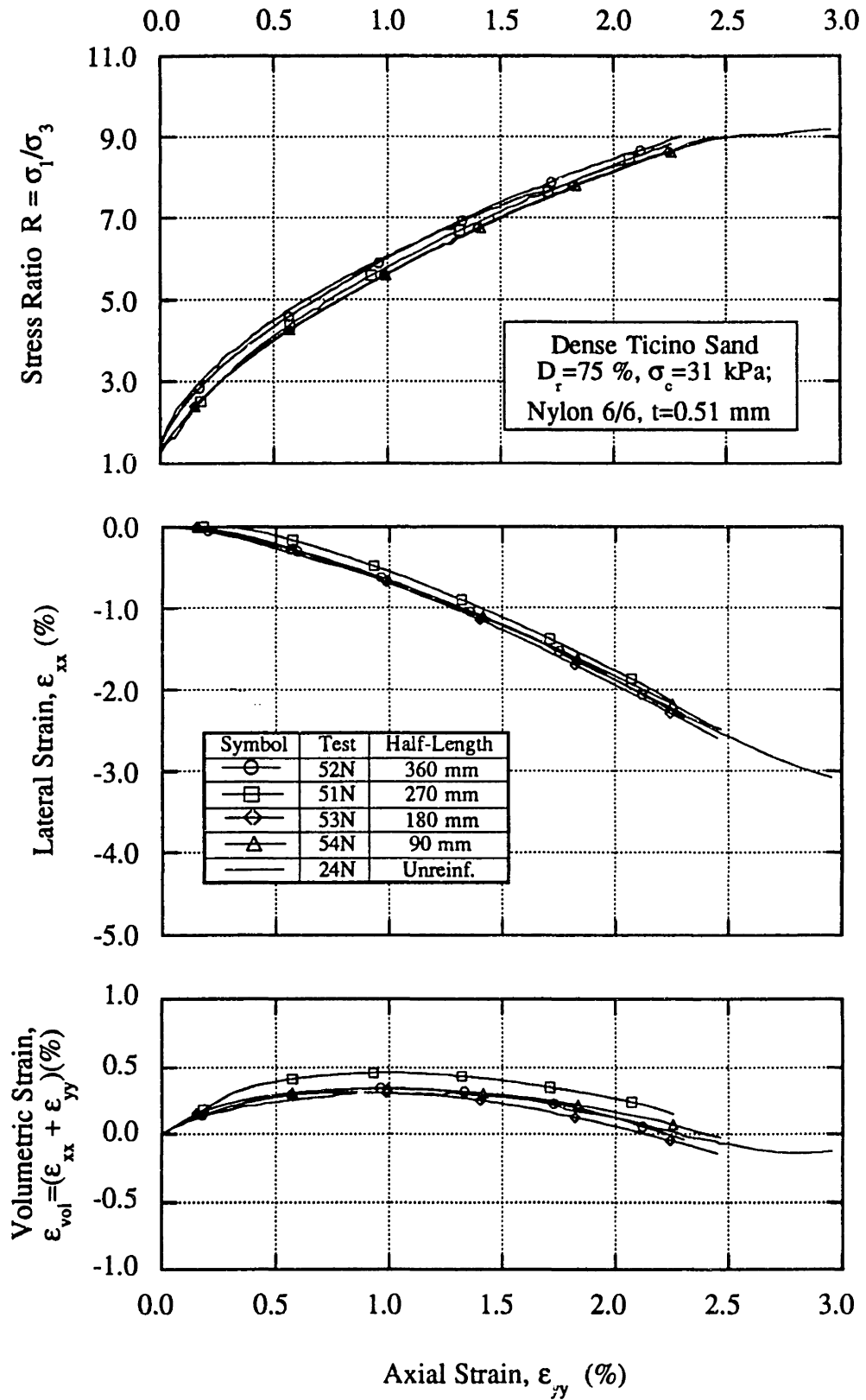


Figure 4.13: Externally Measured Shear Behavior of Dense Ticino Sand Reinforced with Nylon 6/6 Sheet Reinforcements.

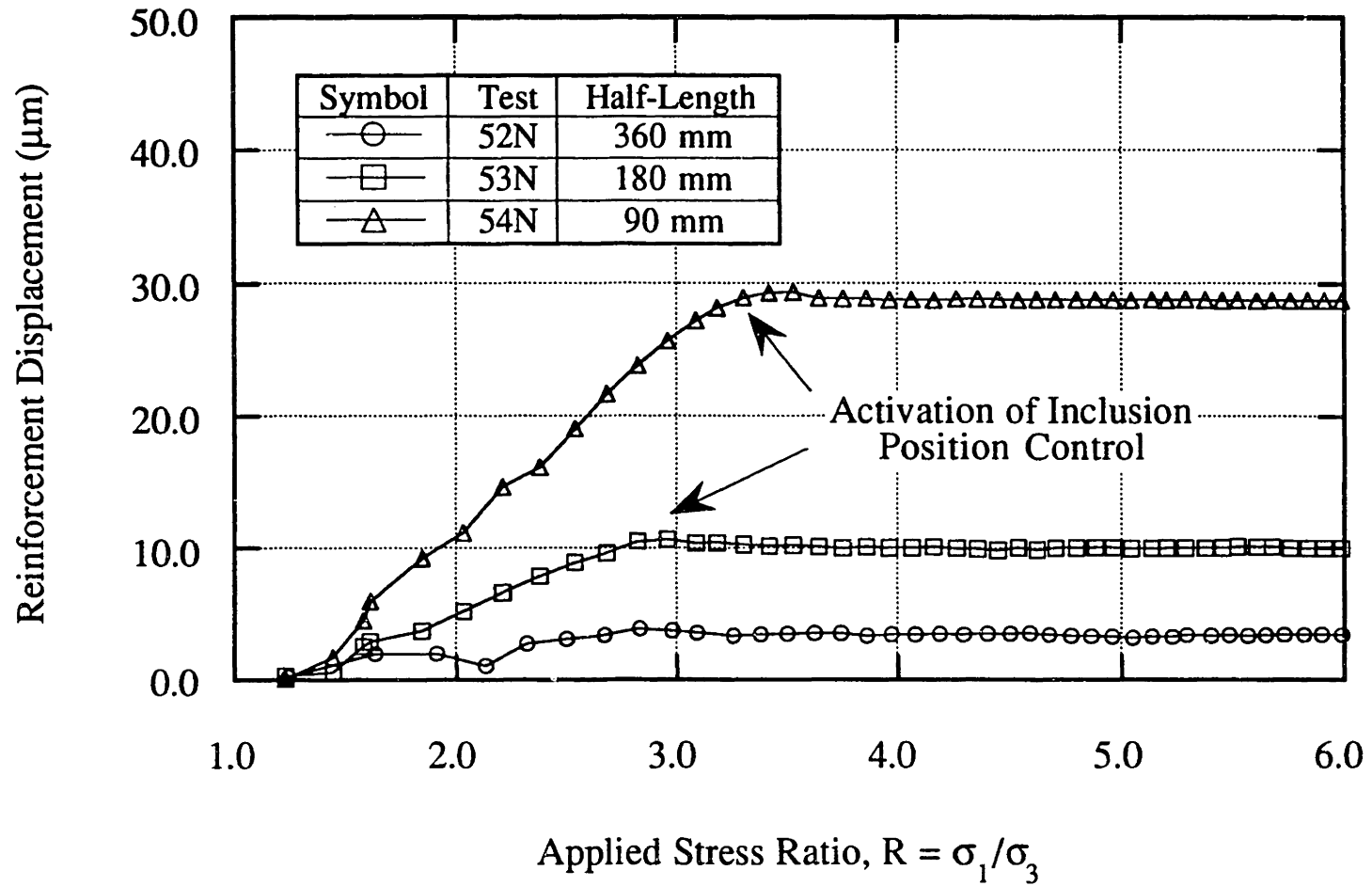


Figure 4.14: Outward Movements of Nylon 6/6 Inclusions during the Initial Phase of the APSR Test.

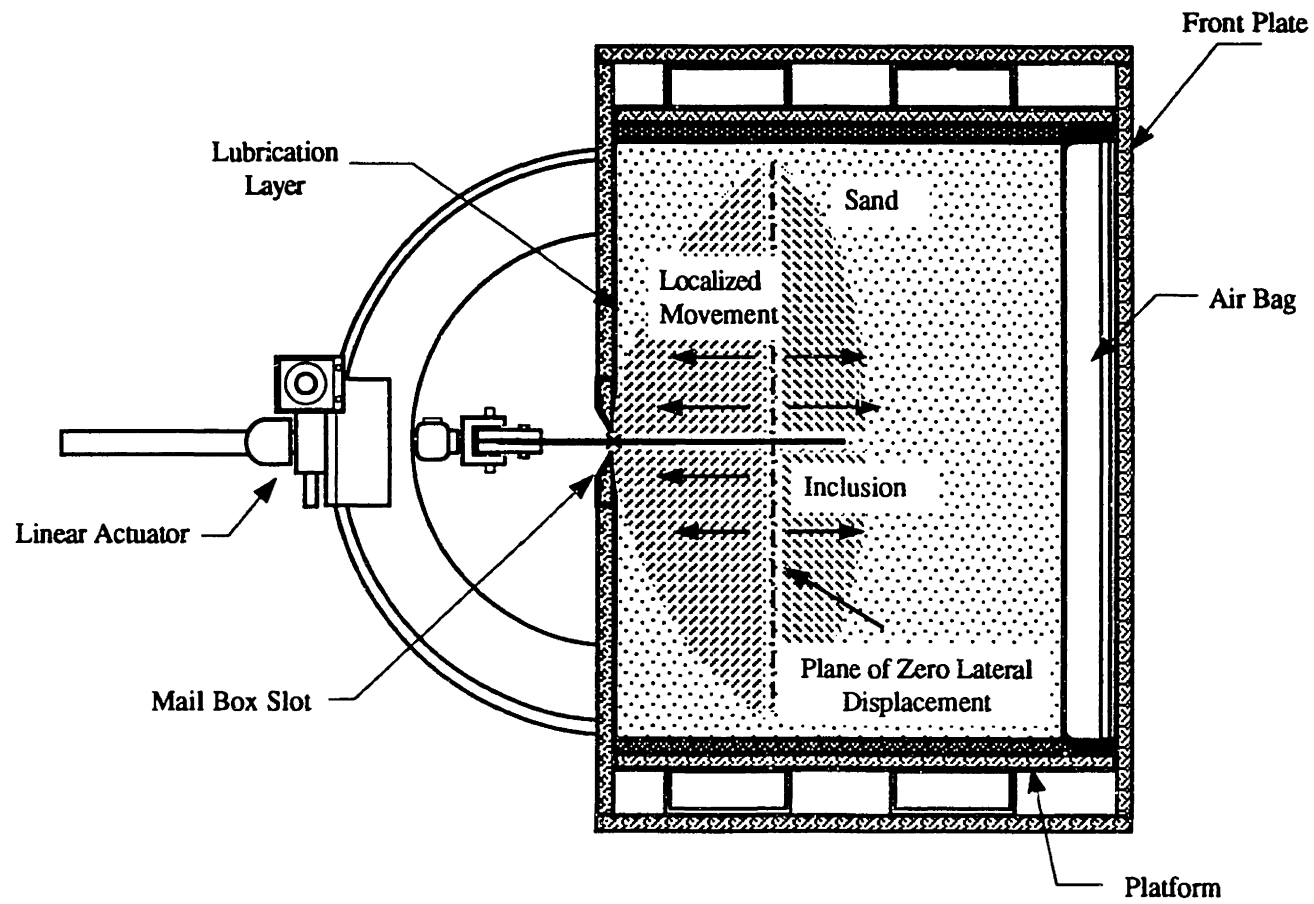


Figure 4.15: Possible Effects of Arching at Rear Wall of the Cell on Displacement Field within Reinforced Soil Specimen.

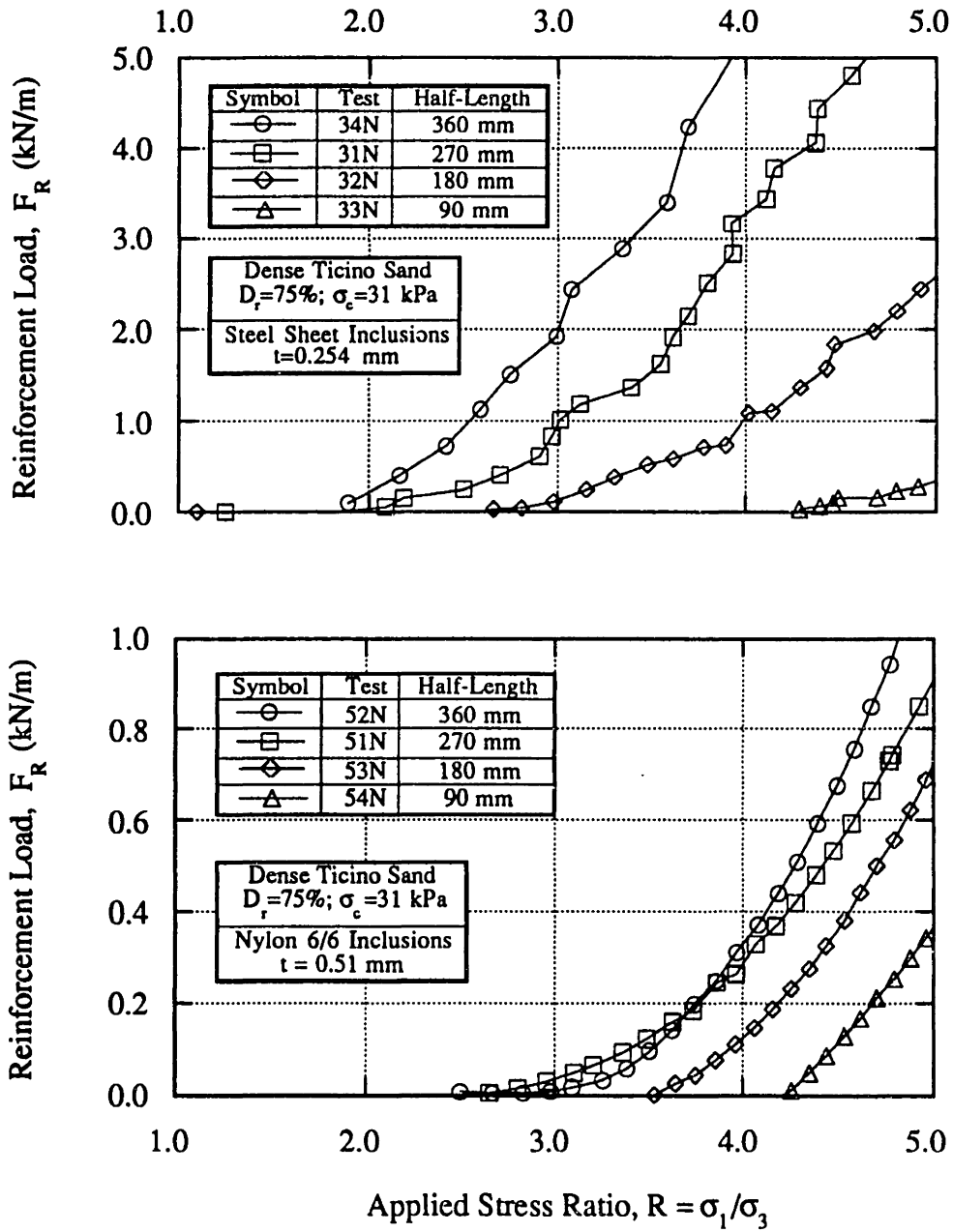


Figure 4.16: Development of Tensile Loads for Steel and Nylon 6/6 Sheet Inclusions in the APSR Cell.



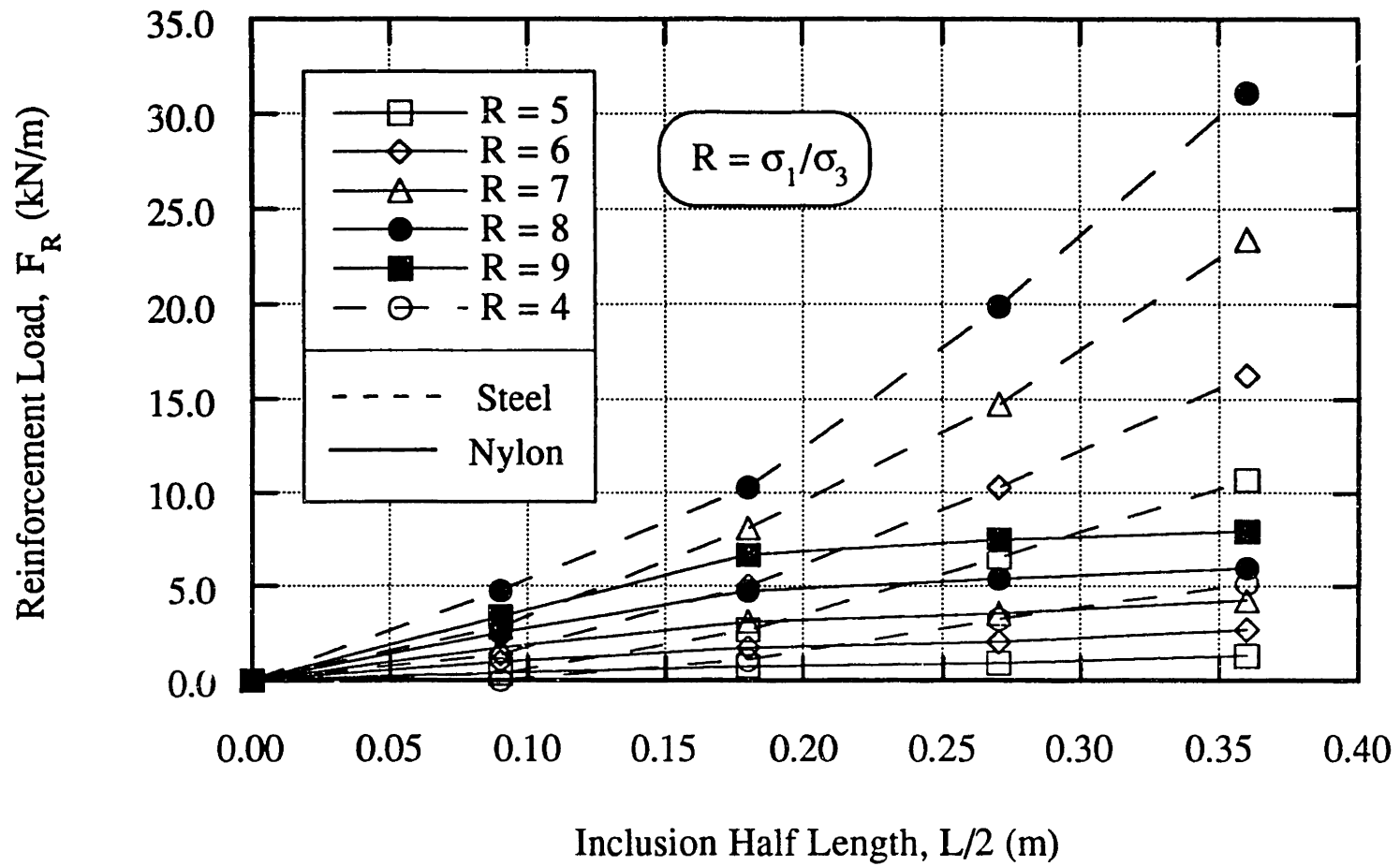


Figure 4.17: Effect of Inclusion Length on Centerline Tensile Loads for Steel and Nylon 6/6 Sheet Inclusions.

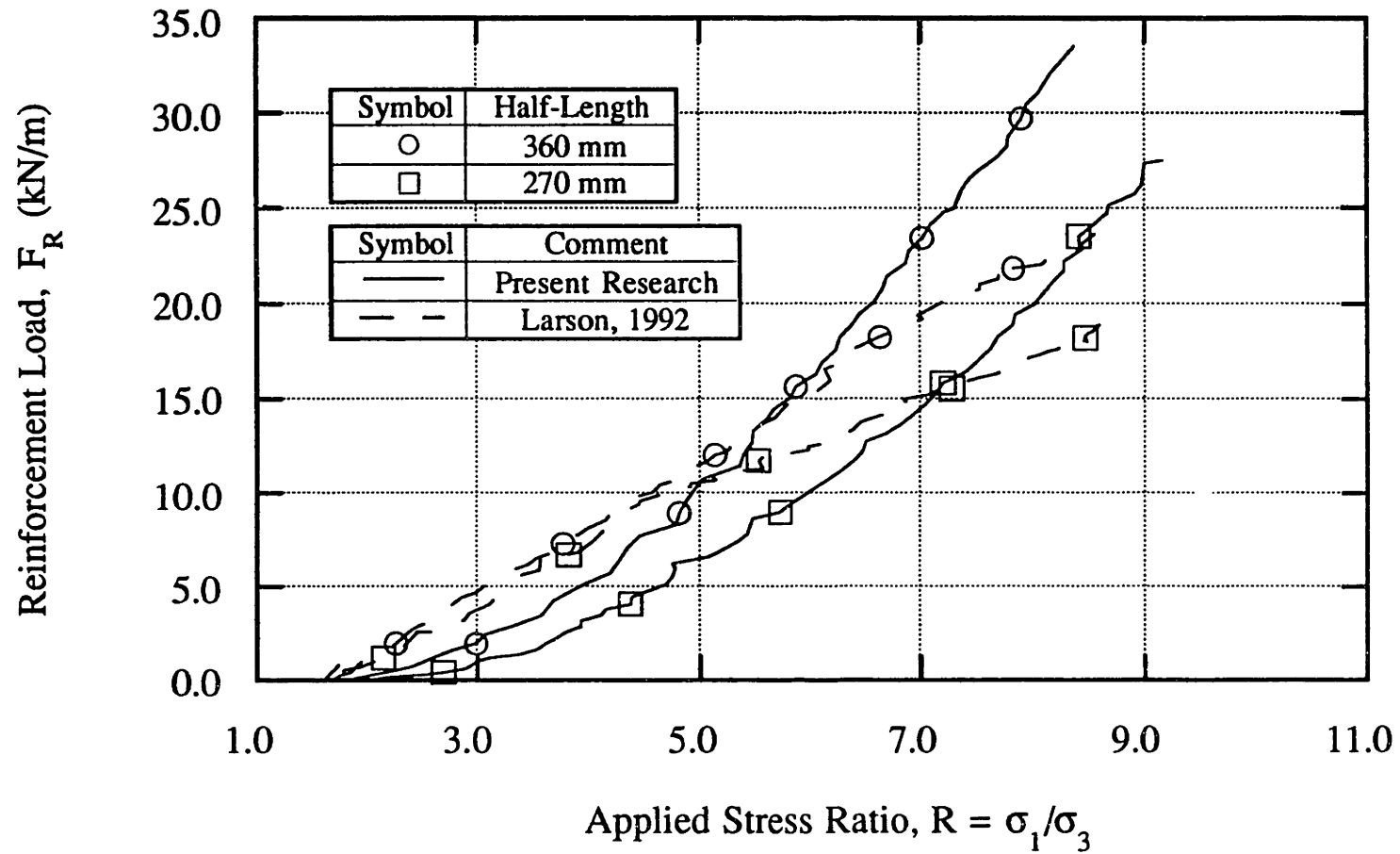
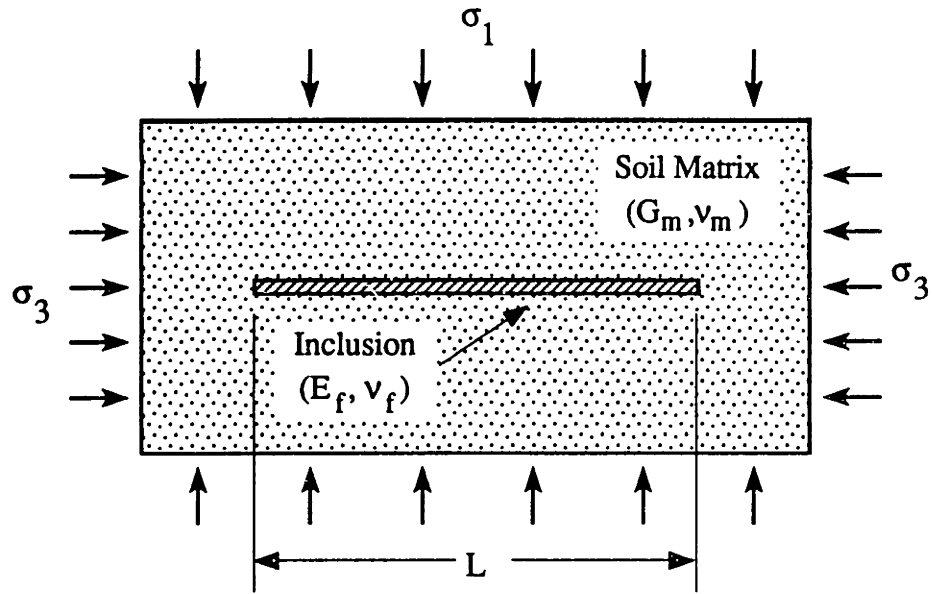
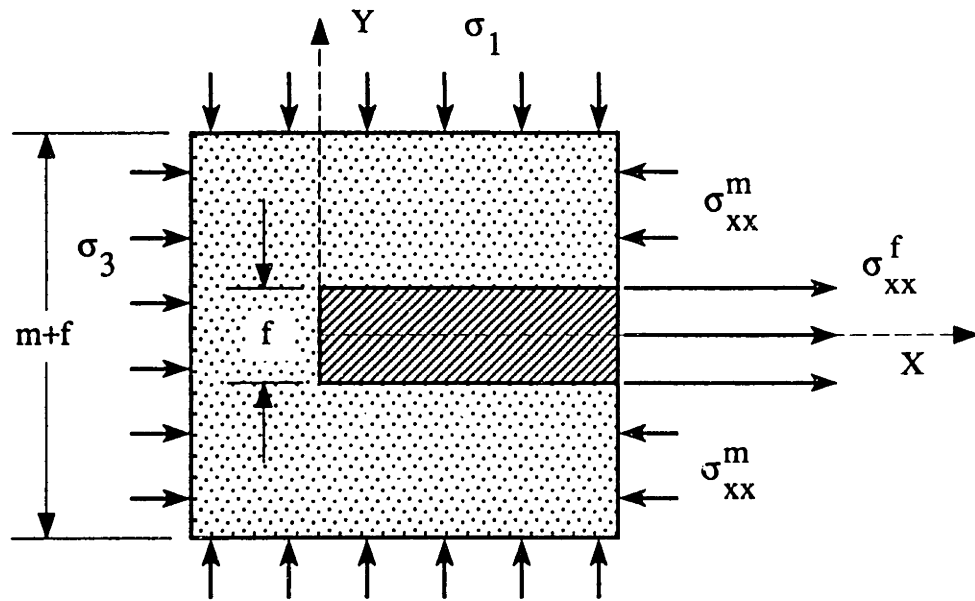


Figure 4.18: Comparison of Load-Transfer Measurements for Steel Inclusions Obtained from the Original and Modified APSR Cell.



a) Geometry and Notations Used in the Shear-lag Analysis



b) Equilibrium of Stresses in Soil Matrix and Planar Inclusion

Figure 4.19: Plane Strain Geometry for an Element of Reinforced Soil (after Abramanto, 1993).

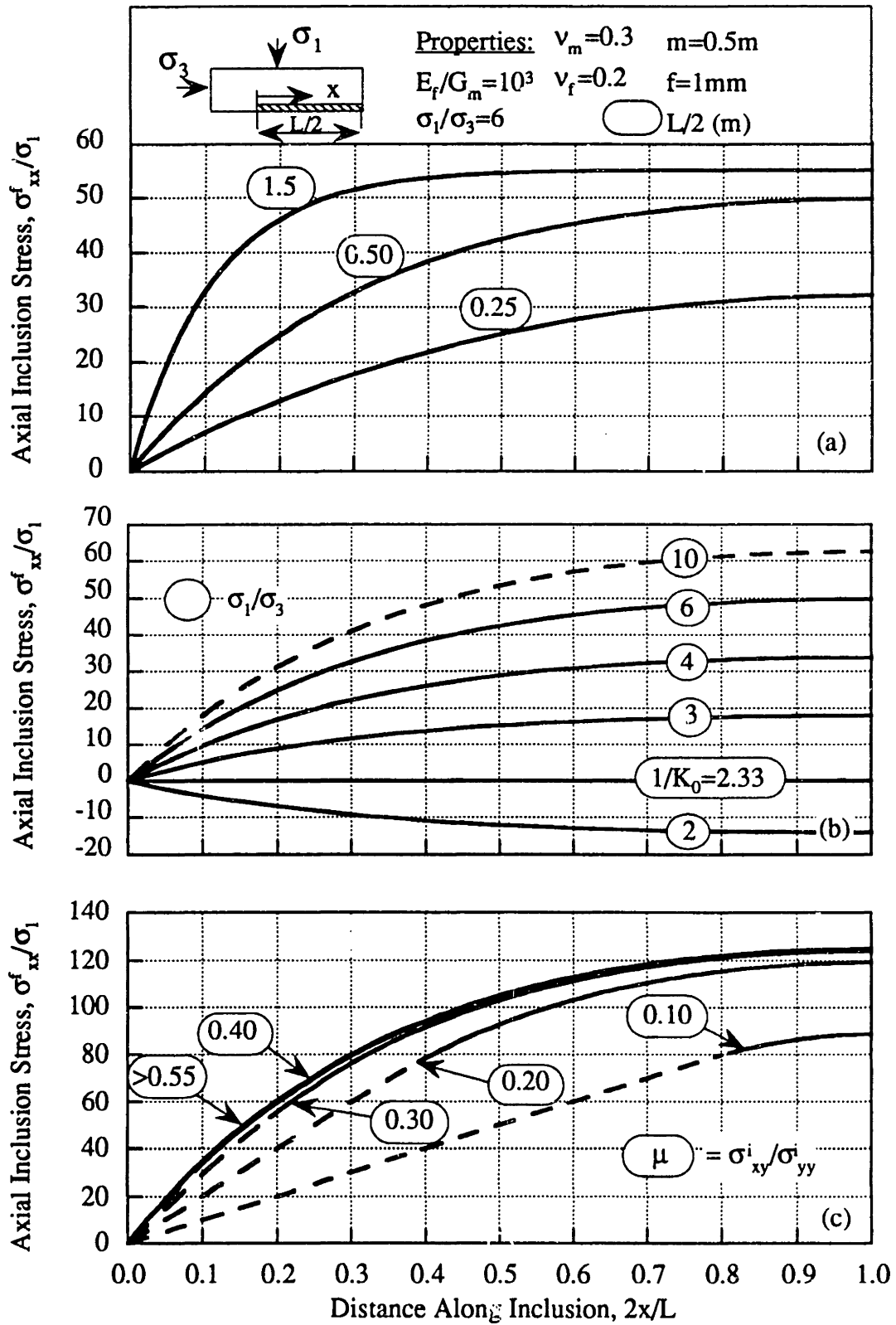


Figure 4.20: Distribution of Tensile Stresses in a Planar Inclusion (after Abrameto, 1993).

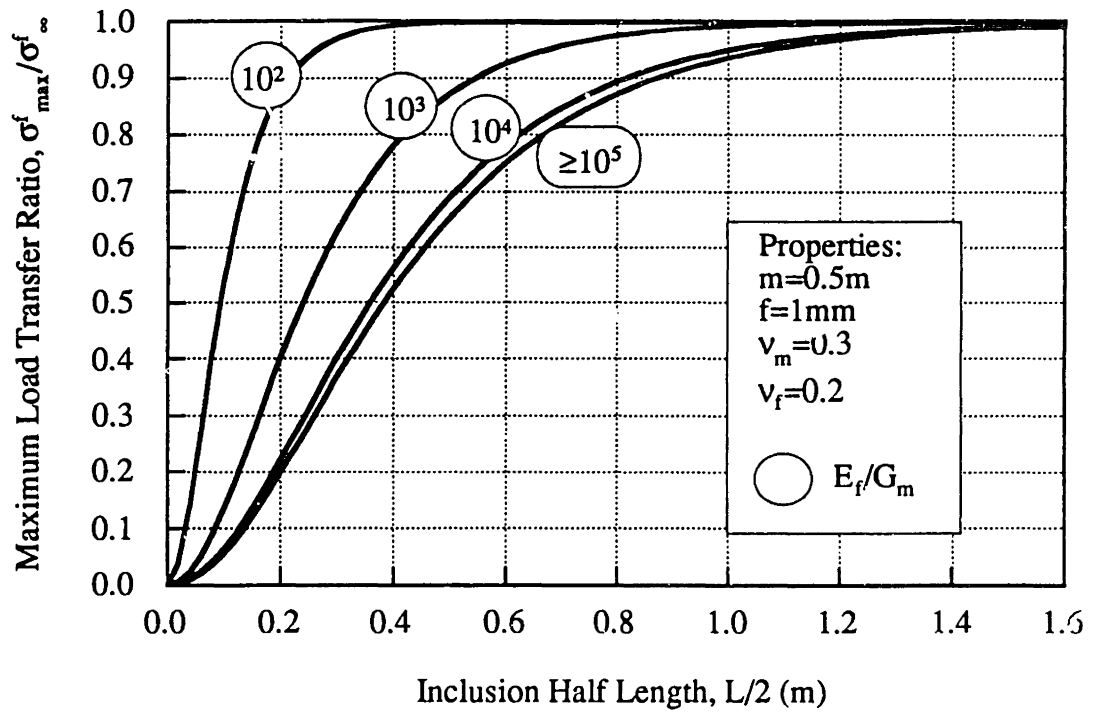
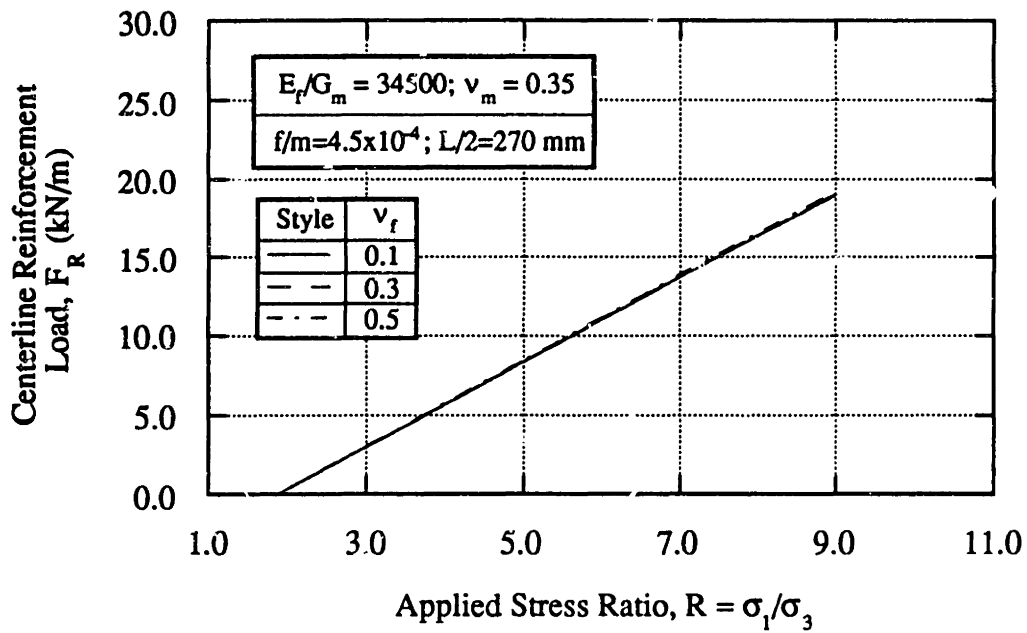
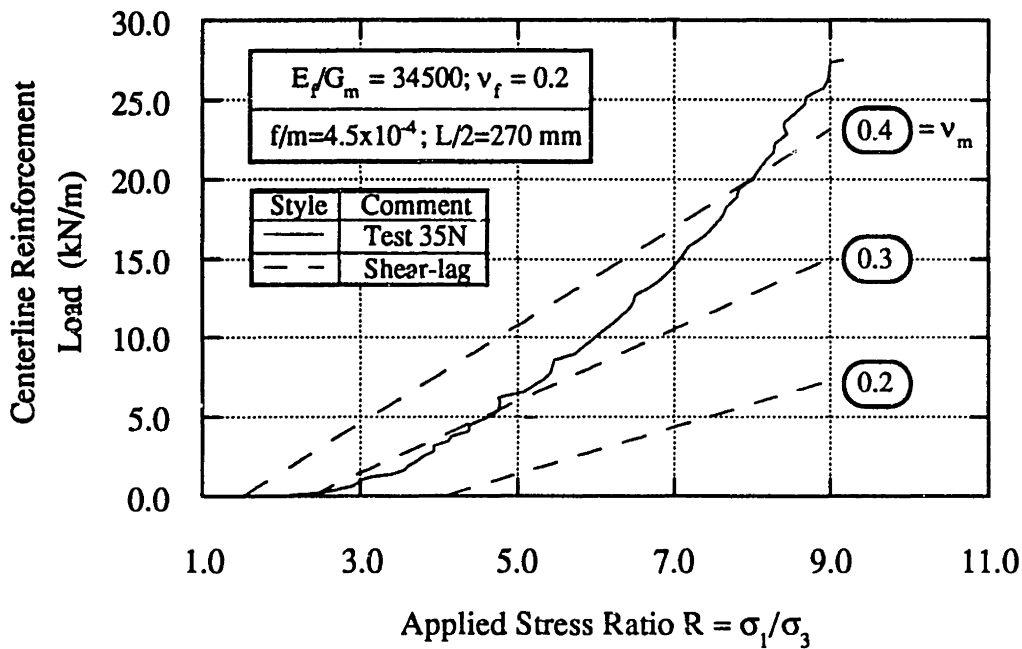


Figure 4.21: Effect of Inclusion Length and Stiffness on the Maximum Load Transfer Ratio (after Abrameto, 1993).



(a) Effect of  $\nu_f$  variations in the Poisson's Ratio of Reinforcement.



(b) Effect of Variations in the Poisson's Ratio of Soil Matrix.

Figure 4.22: Sensitivity of the Shear-lag analysis to the Values of Elastic Input Parameters.

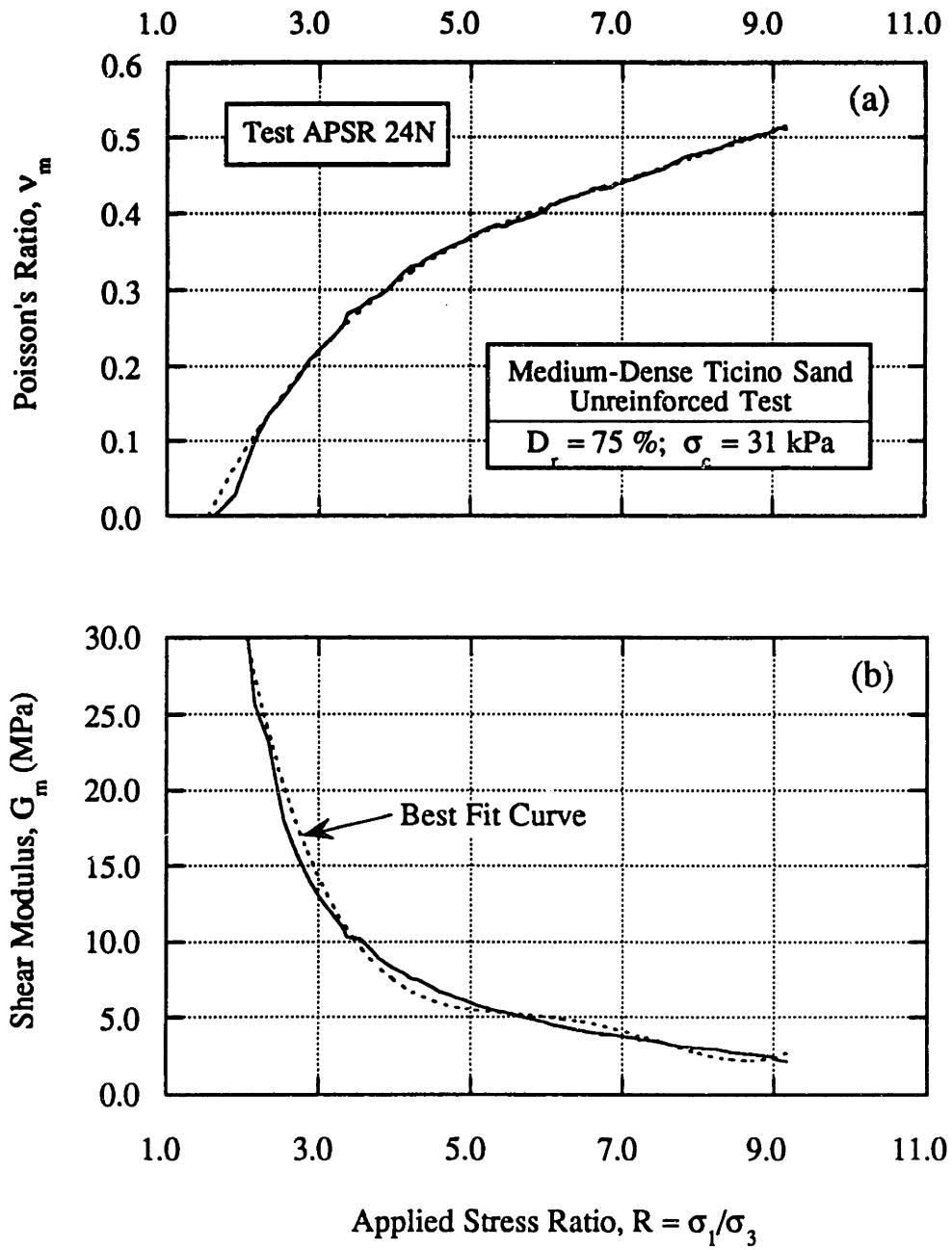


Figure 4.23: Variation of Secant Shear Modulus and Poisson's Ratio for Unreinforced Medium Dense Ticino Sand.

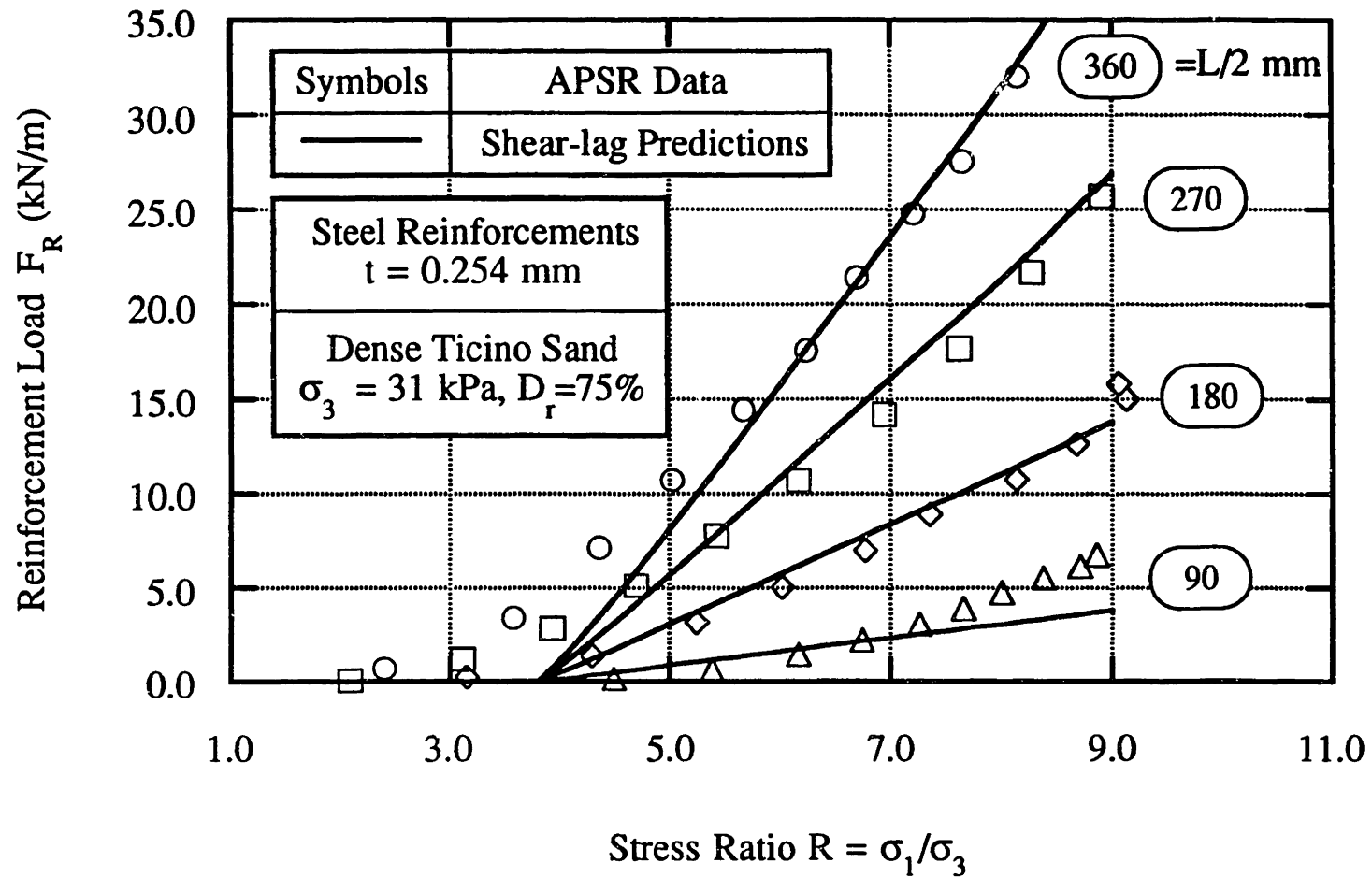


Figure 4.24: Shear-lag Predictions vs. Measured Inclusion Tensile Loads for Steel Sheet Inclusions of Different Lengths.



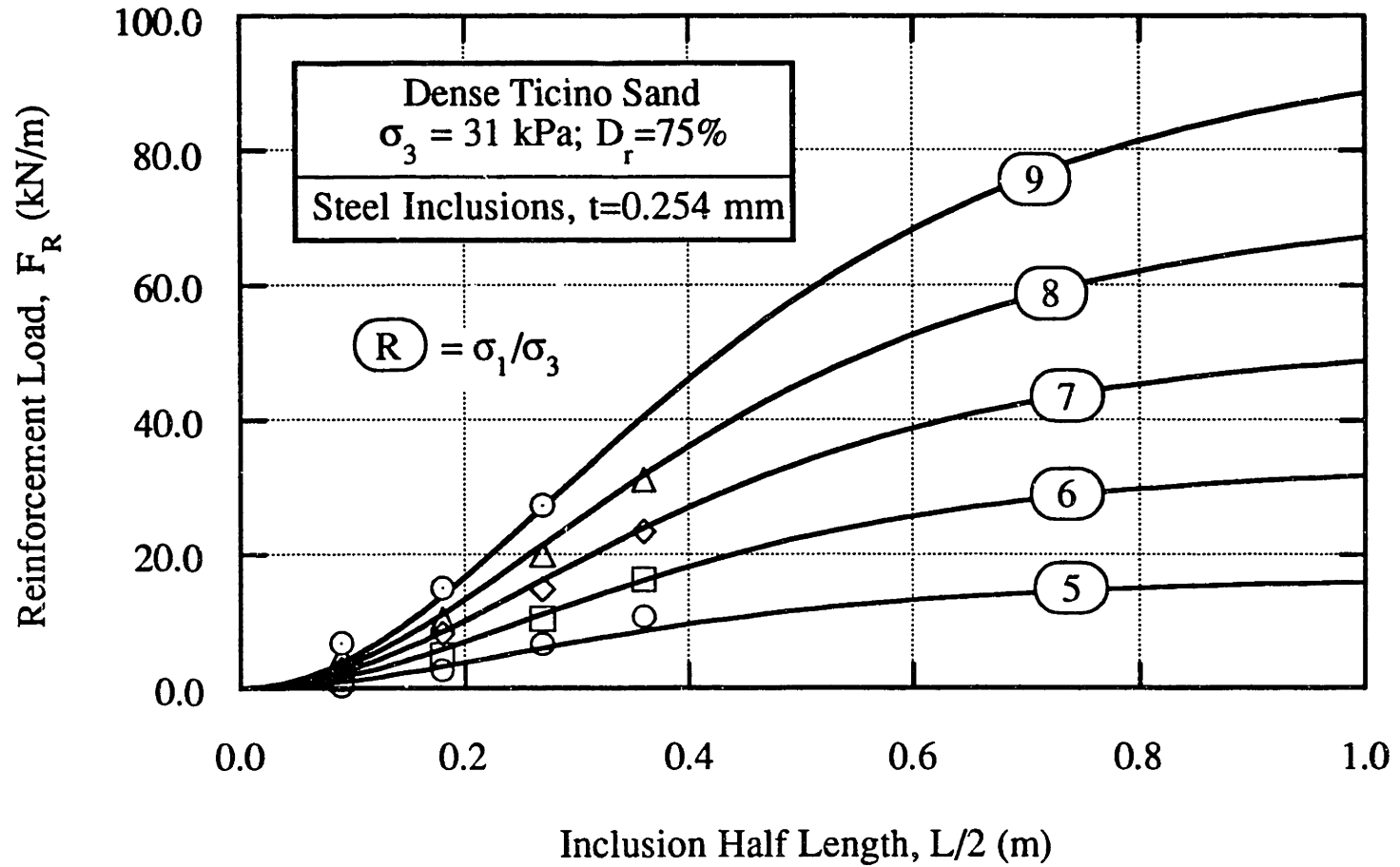


Figure 4.25: Effects of Inclusion Length on Predicted and Measured Load Transfer for Steel Sheet Reinforcements.

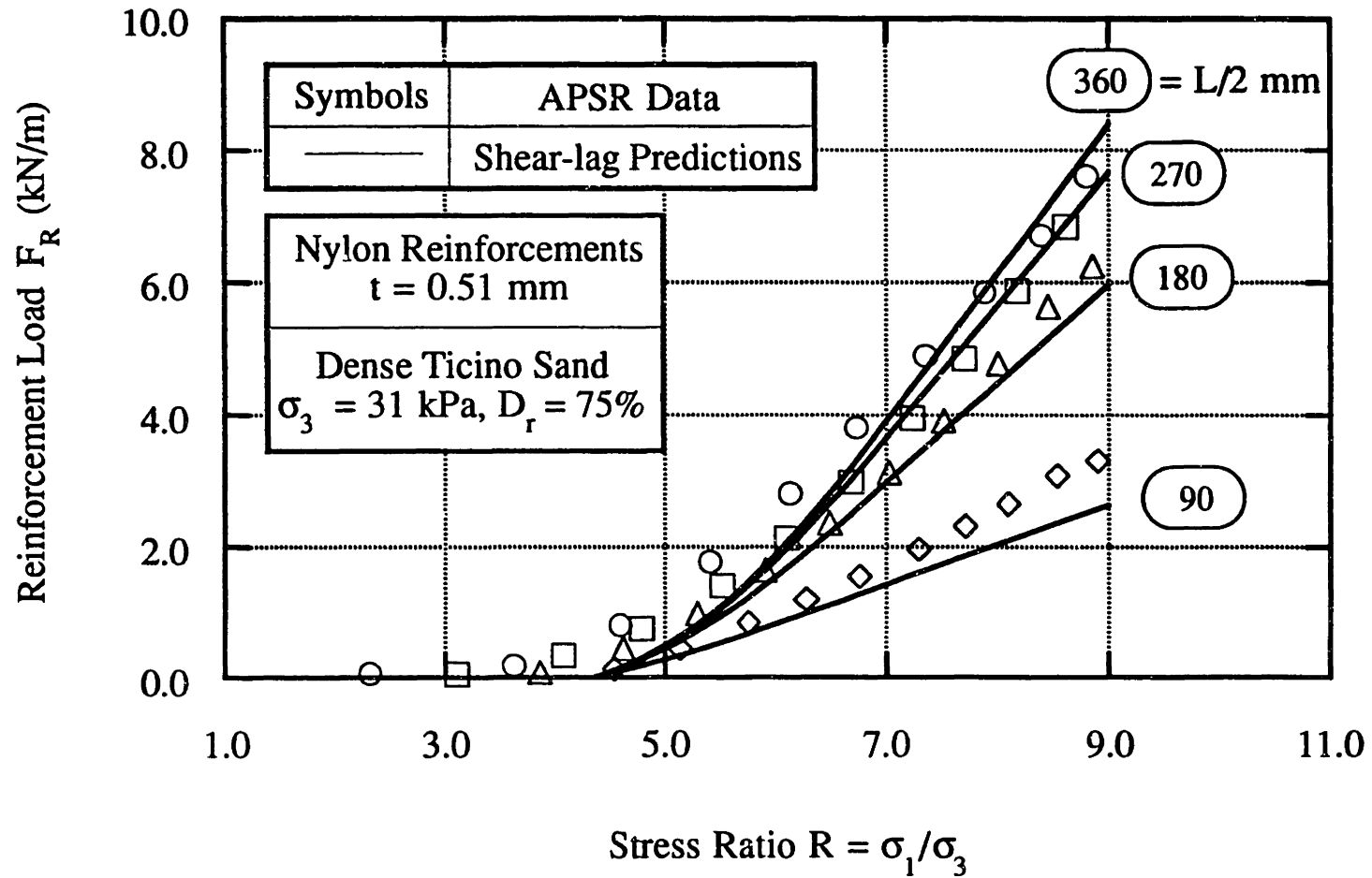


Figure 4.26: Comparison of Shear-lag Predictions and APSR Measurements of Load-Transfer for Nylon 6/6 Sheet Reinforcements.

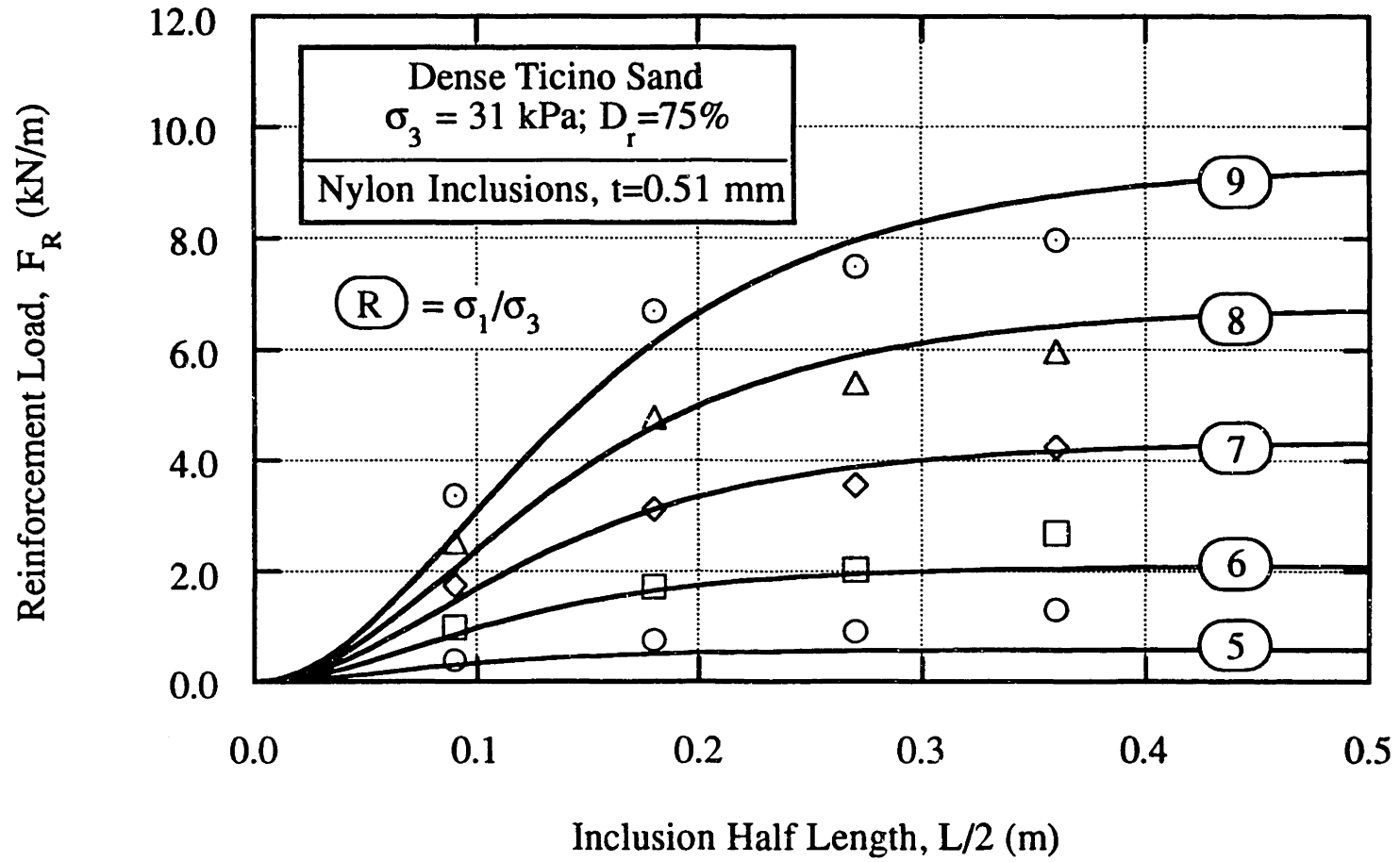


Figure 4.27: Prediction and Measurement of Load-Transfer for Nylon 6/6 Sheet Reinforcements.



# Chapter 5: APSR Tests on Geogrids

## 5.1. Introduction

This chapter presents and discusses the results of APSR tests performed on Fortrac<sup>®</sup> geogrid reinforcements embedded in dense Ticino sand. Fortrac<sup>®</sup> is the trade name for a high-strength, woven, polyester geogrid manufactured and supplied by Huesker Corporation. The present testing program uses two different grades of Fortrac geogrids that differ mainly with regard to their axial stiffness and ultimate tensile strengths. The details on geogrid material properties, specimen preparation, and testing procedures are given in Chapter 3.

Sections 5.2 and 5.3 establish the repeatability and reliability of APSR tests performed on Fortrac geogrids and evaluate the main trends in the measurements of load-transfer. The discussion focuses on the effects of the grid geometry on the measured tensile stresses. Comparisons with results for Nylon 6/6 inclusions (Sections 4.3, 4.4) and shear-lag predictions of load-transfer are used to show similarities and differences between the behavior of geogrids and planar inclusions in the APSR cell.

## 5.2. Presentation of Grid Test Results

### 5.2.1 Overview

A total of thirteen APSR tests were performed using Fortrac geogrid reinforcements. Table 5.1 presents a summary of the experimental program which included two grades of geogrid: Fortrac 80/30-20 and Fortrac 110/30-20.<sup>1</sup> Fortrac 80/30-20 grade was used in the first five APSR tests because its axial stiffness,  $J = 1400$  kN/m at small strain ( $\epsilon_a \leq 0.8\%$ ) is comparable to that of the Nylon 6/6 sheet reinforcement ( $J = 980$  kN/m) used earlier in the APSR research (refer to Sections 3.2, 4.3). The refinement of experimental methods including position control and seating of grid specimen in the APSR required several iterations, such that data for tests 57N-59N (Table 5.1) are not reported in this chapter. Two successful tests (APSR 60N and 61N) were performed using Fortrac 80/30-20 grid specimens prepared from the limited quantity of samples supplied by the manufacturer. A second larger shipment of material included sufficient samples of Fortrac 110/30-20 to perform a complete series of APSR tests ( $L/2 = 100$ -360 mm), but no further samples of Fortrac 80/30-20 grade. Therefore, the remainder of test program was completed using the slightly stiffer Fortrac 110/30-20 geogrid reinforcements ( $J = 2000$  kN/m). The following sections are focused primarily on presentation of load-transfer measurements in the APSR cell using Fortrac 110/30-20 grid reinforcements. Results from the APSR test on Fortrac 80/30-20 grids are provided wherever they compliment the discussion.

---

<sup>1</sup> The first two numbers in the grid designation (i.e., Fortrac 20/13) indicate wide width ultimate strengths (as per ASTM D-4595) in kN/m, measured in the longitudinal and lateral directions respectively.

The axial stiffness,  $J$ , of a grid is partially related to the physical width of a specimen, as this controls the 'solid width ratio',  $\alpha_s = B_s/B$ , where  $B$  is the overall width of the test specimen and  $B_s$  is the width of the 'solid' grid members within the specimen. For infinitely wide Fortrac grids,  $\alpha_{s\infty} = 20.30\%$  (refer to Table 3.5). In contrast, the grid specimens used in APSR tests are much narrower with  $B_s = 120.6$  mm (6 longitudinal members in each specimen), hence  $\alpha_s = 23.95\%$ . Thus,  $\alpha_{s\text{APSR}}/\alpha_{s\infty} = 1.18$ , hence the effective width of the specimen is approximately 136.2 mm. Throughout this chapter, the reported force per unit width,  $F_R$ , (in kN/m) has been computed by dividing the measured loads by the effective specimen width.

### 5.2.2. Tensile Loads in Inclusions

A total of seven high quality APSR tests were performed using Fortrac 110/30-20 grid reinforcements embedded in dry, dense Ticino sand ( $D_r = 75\%$ ) at a confining pressure,  $\sigma_3 = 31$  kPa. The reinforcement specimens have three standard half-lengths,  $L/2 = 360, 193,$  and  $109$  mm which correspond to an integral number of cells within each grid specimen. Figure 5.1 shows the measured tensile load in the grid reinforcements,  $F_R$ , as a function of the applied stress ratio,  $R = \sigma_1/\sigma_3$ , for the grid specimens with half-lengths,  $L/2 = 193$  and  $360$  mm. There is excellent repeatability and consistency in the measurements confirming the high quality of the test procedures used in the APSR cell.

The experimental observations of load-transfer in Figure 5.1 show all the characteristic features reported previously for planar steel and Nylon 6/6 sheet inclusions (Sections 4.3, 4.4). Tensile loads are only transferred to the grid reinforcements when the applied stress ratio is higher than a characteristic value,  $R^* \approx 4.5$ . During the initial phase of the test ( $R < R^*$ ), the reinforcing inclusion is allowed to move freely and the active control of reinforcement position is established only when

the exit point displacements indicate the onset of tensile load-transfer. Thereafter, the tensile loads in the reinforcement increase as the surrounding soil is sheared to a stress ratio,  $R = 9$  ( $\phi' \approx 53^\circ$ ), corresponding to failure conditions in the unreinforced soil. The rate of load-transfer,  $dF_R/dR$ , also increases gradually with the applied stress ratio and approaches a constant value at high values of stress ratio ( $R > 7$ ).

Figure 5.2 and 5.3 show the effects of inclusion length on the magnitude of tensile loads in Fortrac 110/30-20 and 80/30-20 geogrid reinforcements respectively. A comparison between Figure 5.2 and 5.3 shows that the load-transfer characteristics of Fortrac 80/30-20 grid reinforcements are qualitatively very similar to those of Fortrac 110/30-20 grids. In general, at any given value of the applied stress ratio,  $R$ , both the magnitude of the tensile load in the grid reinforcement and the transfer gradient,  $dF_R/dR$  increase with the length of the reinforcement. At an applied stress ratio,  $R = 7$ , the measured centerline loads for the Fortrac 110/30-20 grids are  $F_R = 4.55$  kN/m and 2.15 kN/m for reinforcements of half-lengths  $L/2 = 193$  mm and 109 mm respectively. The tensile loads for Fortrac 80/30-20 are slightly lower; for the same stress ratio and inclusion lengths,  $F_R = 3.5$  kN/m and 1.58 kN/m.

For both grades of geogrid, as the inclusion half-length changes from,  $L/2 = 109$  mm to  $L/2 = 193$  mm, the measured tensile load,  $F_R$ , increases by approximately 100%. However, the data for Fortrac 110/30-20 (Figure 5.2) show that tensile loads are almost identical for inclusions with half-length,  $L/2 = 360$  mm. In fact, for lower values of the applied stress ratio ( $R < 7$ ), the loads in the longer inclusion ( $L/2 = 360$  mm) are slightly lower than those in the shorter inclusion ( $L/2 = 193$  mm). This surprising result is contrary to earlier measurements for planar inclusions (steel and Nylon 6/6) in which the centerline tensile load always increased with the inclusion length for  $L/2 \leq 360$  mm in the APSR cell. Table 5.2 summarizes the APSR



measurements for Fortrac 110/30-20 geogrids. The tensile strains,  $\epsilon_f$ , at the centerline (exit point) location reported in Table 5.2 were computed by dividing the measured reinforcement loads by the small strain axial stiffness of the grid,  $J = 2000 \text{ kN/m}$ . The maximum axial strain in the grid reinforcements,  $\epsilon_f \approx 0.5\%$ , is comparable in magnitude with the strain level assumed for computing the small strain stiffness of the Fortrac 110/30-20 grid from tensile test data (refer to Section 3.2.2).

Figure 5.4 shows the inclusion loads plotted against the average lateral strain in the soil,  $\epsilon_{xx}$ , measured at the specimen boundary. The results show that the centerline loads can be reasonably described as linear functions of the lateral strain in the soil matrix. The tensile load-transfer, however, does not begin until a small amount of offset tensile lateral strain,  $\epsilon_{xx}^* \approx -0.1\%$ , occurs in the sand specimen. The 'apparent stiffness coefficient' defined as,  $E_a = dF_R/d\epsilon_{xx}$ , relates lateral strains in the soil matrix to inclusion loads. Table 5.3 summarizes the results from linear regression analyses which show that the "apparent stiffness" for Fortrac 110/30-20 reinforcements ranges from  $E_a = 977 \text{ kN/m}$  at  $L/2 = 360 \text{ mm}$  to  $E_a = 170 \text{ kN/m}$  at  $L/2 = 109 \text{ mm}$  with the average regression coefficient,  $r^2 = 0.989$ . It should be noted that although an increase in the inclusion length beyond,  $L/2 = 193 \text{ mm}$  does not produce a higher inclusion load,  $F_R$ , at a given shear stress (Figure 5.2), the  $L/2 = 360 \text{ mm}$  inclusion does have a very significant effect on the overall stress-strain behavior of the soil matrix as indicated by approximately 240% increase in the apparent stiffness,  $E_a$ .

### 5.2.3. Shear Behavior of Ticino Sand

Figures 5.5 and 5.6 show the shear behavior of the APSR specimens of dense Ticino sand reinforced with Fortrac 80/30-20 grid. The figures also include results from the base-case test on unreinforced Ticino sand (APSR 24N) for comparison. All stresses and strains reported in these figures were measured at the specimen

boundaries. Figure 5.5 plots the applied stress ratio,  $R$ , against the average lateral boundary strain,  $\epsilon_{xx}$ . For any given value of the applied stress ratio, lateral strains in the soil specimens reinforced with Fortrac 80/30-20 grids are only slightly lower than corresponding strains in the unreinforced specimen. These results are consistent with previous observations for Nylon 6/6 sheet reinforcements (see Section 4.4.4) which showed that the additional confinement of the soil due to reinforcement tensile loads of the order of,  $F_R = 10$  kN/m are not significant enough to modify the overall stress-strain behavior of the sand matrix. The observed stress-strain behavior of APSR specimens reinforced with 105 and 195 mm long Fortrac 80/30-20 geogrids (Figure 5.6) confirm these earlier findings.

The influence of Fortrac 110/30-20 inclusions on the externally measured shear behavior of Ticino sand is presented in Figures 5.7 and 5.8. The figures show that for inclusions with half-length,  $L/2 \leq 195$  mm, the shear behavior of the soil-reinforcement composite is identical to the behavior of the unreinforced soil specimen as would be expected based on the similar results for Fortrac 80/30-20 grids. However, in the test 65N with a specimen of half-length,  $L/2 = 360$  mm, there is a large difference in the externally measured stress-strain behavior of the soil matrix. The Fortrac 110/30-20 grid inclusion with half-length,  $L/2 = 360$  mm is able to reduce the average lateral strain in the soil matrix by approximately 50% (Figure 5.7). Similarly, Figure 5.8a shows that the shear stress ratio,  $R$ , versus axial strain,  $\epsilon_{yy}$ , response of test 65N is approximately 25% stiffer than the unreinforced soil specimen. The presence of the longest inclusion, however, does not affect the externally measured lateral strains,  $\epsilon_{xx}$ , or soil dilatancy as shown in Figures 5.8b and c.

## 5.2.4 Effects of Transverse Members

For shorter inclusions (i.e.,  $L/2 < 200$  mm), the results presented so far indicate that the load-transfer behavior of Fortrac geogrids in the APSR cell is comparable to that of inclusions made from planar (flat sheet) materials such as Nylon 6/6. On the other hand, the longest grid inclusion ( $L/2 = 360$  mm) is far more effective in reducing the lateral strains in the soil matrix and increasing shear stiffness of the soil-reinforcement composite than a planar inclusion of comparable axial stiffness,  $J$ , and length. The differences and similarities between the load-transfer behavior of planar and grid materials were investigated further by altering the grid geometry by removing transverse members (ribs), such that there is no change in the axial stiffness of the specimens. Table 5.4 compares the geometry of intact and modified (Fortrac 110/30-20) grid inclusions used in this phase of the work.

Figure 5.9 shows effects of removing transverse grid elements on the measured tensile load-transfer for grid reinforcements with two different lengths,  $L/2 = 193$  and  $360$  mm. The data show that the transverse ribs have no effect on the magnitude of the load-transfer for the grid inclusion with half-length,  $L/2 = 193$  mm. For the  $L/2 = 360$  mm, however, the centerline tensile load in the modified grid is almost 20% higher than that of the intact grid, at  $R = 9$  (i.e., removing grid elements increases the maximum tensile load). This apparently paradoxical result can be understood by looking at the relationship between  $F_R$  and the lateral strains in the soil,  $\epsilon_{xx}$  (Figure 5.10). The figure shows that this relationship is unique for a grid inclusion of specified stiffness and length, irrespective of rib geometry. A comparison of Figures 5.10b and 5.9b shows that higher loads are induced in the modified grid because larger lateral strains are required in the soil matrix to mobilize a given stress ratio,  $R$ , when ribs are removed from the grid reinforcement. Figure 5.11b shows that,

at a given value of the stress ratio,  $R$ , the average lateral strain,  $\epsilon_{xx}$ , is related to the presence of the transverse ribs (i.e., results for modified grid are intermediate between those of intact grid and unreinforced soil). Since the tensile load in the reinforcement is generally proportional to the lateral strain,  $\epsilon_{xx}$ , in the soil specimen, higher lateral strains for a given value of applied stress ratio means higher tensile loads in the inclusion. It should be noted that the modified grid reinforcement in Figure 5.11b is still able to inhibit lateral strains in the soil by as much as 40%. As reported earlier (Section 5.2.3), the rib geometry for short grid reinforcements ( $L/2 < 200$  mm) have negligible effects on the external stress-strain behavior of the soil (Figure 5.11a).

Figures 5.12 and 5.13 show the boundary measurements of stress-strain behavior of reinforced Ticino sand in a more conventional form and support the above observations which can be summarized as:

1. The longest grid reinforcement affects the stress-strain behavior of the soil matrix while the shorter reinforcements ( $L/2 = 193$  and  $109$  mm) do not have any significant influence on the externally measured sand behavior.
2. The behavior of grid reinforcements without the transverse elements (modified grid) in the APSR cell is intermediate between that of the intact grid and a planar inclusion of an equivalent axial stiffness.

### 5.3. Discussion and Interpretation APSR Data

The previous sections described the results from a total of 9 tests which were performed on dry, medium dense Ticino sand reinforced with two types of Fortrac geogrid inclusions. Overall, the load-transfer behavior of the grid reinforcements is very similar to that of a planar inclusion with same axial stiffness. It is possible to

obtain some insight into the influence of reinforcement geometry (i.e., planar vs. grid) by comparing the load-transfer data from the APSR cell with shear-lag predictions of  $F_R$  versus  $R$ . The shear-lag analysis assumes a planar inclusion geometry and predicts a certain distribution of stresses within the inclusion (refer to Chapter 4). The prediction methodology presented in Section 4.5.2 uses secant elastic soil properties to simulate the non-linear soil behavior and provides a good match between the shear-lag predictions and measured load-transfer for planar (steel and Nylon 6/6) sheet inclusion up to 360 mm in length. Since the elastic properties of Fortrac geogrids are similar to those of the Nylon 6/6 reinforcements, any additional discrepancies between the experimental results and theoretical predictions for Fortrac grid reinforcements can be attributed to geometric factors.

### 5.3.1. Selection of Input Parameters for Shear-lag Analyses

Input parameters for shear-lag analyses include the elastic properties of the unreinforced soil ( $G_m$ ,  $\nu_m$ ) and reinforcement ( $E_f$ ,  $\nu_f$ ), and problem dimensions ( $L$ ,  $f$ ,  $m$ ). Since the shear behavior of unreinforced Ticino sand in the APSR cell is non-linear (refer to Figure 3.18), the most realistic predictions of load-transfer using shear-lag analyses are obtained by using a continuous variation of secant shear modulus,  $G_m$  and Poisson's ratio,  $\nu_m$  in the shear-lag calculations. Figure 4.22 shows the variation of  $G_m$  and  $\nu_m$  with the applied stress ratio,  $R$ , used in the shear-lag predictions of load-transfer in Fortrac grids. The distributions shown in Figure 4.22a and b are implemented in the shear-lag analyses as fourth order polynomials (Equation 4.9) whose coefficients are provided in Table 4.7. In order to compute elastic parameters at a given value of the applied shear stress ratio,  $R$ , using Equation 4.9, the value of external stress ratio is reduced by a factor,  $\beta$ , to account for non-uniformity in the distribution of  $R$  within the soil matrix (Equation 4.10). The APSR tests on steel and

Nylon 6/6 sheet inclusions show that the coefficient  $\beta$  shows some dependence on the axial stiffness of the reinforcement (i.e.,  $\beta = 0.78$  vs.  $\beta = 0.65$  for Nylon 6/6). Since for Fortrac geogrid stiffness per unit width,  $J$ , is of the same order as that of the Nylon 6/6 reinforcement,  $\beta = 0.65$  is used in the shear-lag computations for Fortrac grids.

The axial stiffness of the reinforcement in shear-lag analysis is expressed as the product of thickness,  $f$ , and the elastic modulus,  $E_f$ , (in units of stress) such that any combination of  $f$ , and  $E_f$  that give same value of,  $J = f \times E_f$ , would provide identical results. Therefore, elastic moduli for Fortrac geogrids were obtained from the known values of stiffness per unit width,  $J$ , assuming a unit thickness,  $f = 1$  mm. Very little information exists in the literature regarding the value Poisson's ratio,  $\nu_f$ , for materials with woven, open structures such as Fortrac geogrids. Fortunately, the results of shear-lag analysis are not very sensitive to Poisson's ratio of the reinforcement (Figure 4.21a) and hence,  $\nu_f = 0.2$  was assumed for both Fortrac 110/30-20 and 80/30-20 grids. Table 5.5 summarizes the input parameters used for shear-lag predictions of inclusion stresses in Fortrac geogrid reinforcements.

### 5.3.2. Effects of Inclusion Stiffness and Length

Figures 5.14 and 5.15 compare the shear-lag predictions with APSR measurements of centerline loads for Fortrac 80/30-20 and 110/30-20 geogrid inclusions respectively. The predictions are in excellent overall agreement with the measured loads for Fortrac grid inclusions with half-lengths,  $105 \leq L/2 \leq 195$  mm, and axial stiffness,  $1400 \text{ kPa} \leq J \leq 2000 \text{ kN/m}$ . These results indicate that within these ranges of parameters, the grids essentially behave as planar (sheet) inclusions of equivalent axial stiffness. Figure 5.16 replots the results shown in Figure 5.15 at specified shear stress levels for Fortrac 110/30-20 grids inclusions with half-length,  $L/2 = 109, 193, \text{ and } 360$  mm. The figure clearly demonstrates that the measured load-

transfer for grids as function of the inclusion length follows the shear-lag framework only for inclusions with length,  $L/2 < 200$  mm.

### 5.3.3. Influence of Grid Geometry on Load-Transfer

Figure 5.17 compares the shear-lag predictions of tensile loads with the measured centerline loads for both intact and modified Fortrac 110/30-20 grid inclusions with half-length,  $L/2 = 360$  mm. The figure shows that as most of the transverse rib elements are removed the difference between the shear-lag predictions and experimental measurements of load-transfer narrows down. This indicates that the mechanism of load transfer between the soil and the long grid reinforcement is influenced by the arrangement of the transverse rib elements, when these elements are removed, the load-transfer behavior of grids approaches that of a planar inclusion of the same axial stiffness.

The difference between load-transfer behavior of grid and planar inclusions is related to the differences in the way each of them influences the overall shear behavior of the sand matrix (refer to Section 5.2.3). The reinforcing inclusion affects the shear behavior of the surrounding soil because tensile stress carried by the inclusion increases the confining stress within the zone of reinforcement as illustrated in Figure 5.18. The effect of higher confinement is to decrease the shear stress within the zone of influence compared to externally applied stress ratio,  $R$ , and hence, increase the shear stiffness of the soil matrix. The externally measured shear behavior of the soil matrix is an average behavior and is controlled by the size of the "reinforced zone" relative to the size of the soil specimen. For a given magnitude of the reinforcement tensile load, the size of the reinforced zone depends on the length of the inclusion and the distribution of axial stress within the inclusion (Figure 5.18). Results for planar inclusions (steel and Nylon 6/6 sheets) show that the stiffening of soil is proportional to the magnitude of the

tensile load carried by the inclusion such that when  $F_R < 10$  kN/m, the additional confinement of the soil due to reinforcement tensile loads has negligible effect on the overall stress-strain behavior of sand. Thus, the ability of the longer grid specimens ( $L/2 = 360$  mm) to inhibit lateral strains in the sand and increase the shear stiffness of the soil matrix can be attributed to possible differences in the distribution of tensile stress within the grid and planar inclusions of same length. For a given magnitude of the centerline tensile load, the distribution shown in the lower half of Figure 5.18 will have greater influence on the shear behavior of the composite soil element.

The above discussion suggests that large differences between surface characteristics of reinforcing inclusions can affect their performance under working load conditions. This hypothesis can be readily verified by performing APSR tests using completely roughened Nylon 6/6 sheets and comparing results with the regular Nylon inclusions. Alternately, geometry of a planar inclusion can be altered by punching holes in a regular pattern. The later approach was used in one APSR test (APSR 70N) in which a 1.25 mm thick, two ply Nylon 6/6 sheet (total thickness 2.5 mm) was modified by punching 22.22 mm square holes at 30.63 mm center-to-center spacing. Figure 5.19a shows that the punched Nylon sheet has a greater effect on the shear stress-strain behavior of the soil even though both inclusions carry identical tensile loads as shown in Figure 5.20. The centerline tensile loads are plotted against axial strain in Figure 5.19b which shows that for a given value of inclusion load. A more detailed interpretation of the test is complicated by the fact that the equivalent thickness of the punched sheet (total 'solid' cross sectional area divided by width) is 0.617 mm and is different from the thickness of planar Nylon 6/6 inclusions ( $t = 0.51$  mm).



The APSR tests on grid inclusions suggest possible limitations of all analyses including the finite element method which treat grid reinforcements as planar elements with equivalent axial stiffness. If transverse rib elements affect reinforcing performance of grids at working load levels through bearing or other kinds of mechanisms which are not explicitly modeled in the analysis, the results can be misleading.

## 5.4. Conclusions

This chapter presented results from a series of APSR tests performed using Fortrac geogrids embedded in Ticino sand at a relative density of 75% and sheared at a confining pressure of 31 kPa. The tests provided load-transfer measurements for two grades of grid inclusions, Fortrac 110/30-20 ( $J = 2000$  kN/m) and Fortrac 80/30-20 ( $J = 1400$  kN/m) with three half-lengths,  $L/2 = 105, 195,$  and  $360$  mm which show the following:

1. The tensile loads in the grid inclusions with  $L/2 < 200$  mm are similar in magnitude to loads in planar (sheet) reinforcements of comparable stiffness (e.g. Nylon 6/6). The magnitude of load-transfer for a given value of the externally applied shear increases with the axial stiffness and length of the reinforcement. The results also show that when the inclusion half-length,  $L/2 < 200$  mm, the grid reinforcements do not alter the externally measured shear behavior of the soil-reinforcement composite.
2. There is excellent overall agreement between the measured tensile loads in Fortrac grid inclusions with half-length,  $105 \leq L/2 \leq 195$  mm, and shear-lag prediction obtained assuming coefficient,  $\beta = 0.65$ . The results imply

that within these range of parameters, the grids seem to behave as planar (sheet) inclusions of equivalent axial stiffness.

3. There are significant differences in the load-transfer behavior of grids and planar reinforcements when inclusions are of half-length,  $L/2 = 360$  mm. The results in this chapter show that when the size of the reinforced zone is significant compared to the size of the soil matrix, the grid is much more effective in modifying the external shear behavior of the matrix than a Nylon 6/6 inclusion of the same length and carrying loads of similar magnitude. The ability of the longest grid reinforcement to inhibit lateral strains in the soil and increase its shear stiffness depends on the arrangement of transverse elements in the inclusion. More detailed studies are now required to establish how a grid geometry alters the mechanisms of load-transfer.
4. Although the grid geometry affects the shear behavior of the composite soil element, the relationship between inclusion tensile load,  $F_R$ , and the lateral strain in the soil (Figure 5.10) is unique for a grid inclusion of given axial stiffness and length, irrespective of the arrangement of transverse elements.

Grid Type	Test No.	Half-Length, L/2 (mm)	Comments
Fortrac 80/30-20	APSR 57N	195	First test with Grid. Low vacuum due to air leakage
	APSR 58N	195	Repeat of APSR 57N, still low vacuum
	APSR 59N	100	Normal vacuum. Unresponsive load-transfer, suggests problems in position control
	APSR 60N	195	Repeat of APSR 57N with normal vacuum. New grid specimen
	APSR 61N	105	Repeat of APSR 59N with new grid specimen
Fortrac 110/30-20	APSR 62N	360	Some overloading in $\sigma_3$ direction while sample still under vacuum
	APSR 63N	190	Specimen from test 62N cut into shorter length
	APSR 64N	109	Specimen from test 64N cut into shorter length
	APSR 65N	360	Repeat of APSR 62N
	APSR 66N	193	Repeat of APSR 63N. Computer failure
	APSR 67N	193	Repeat of APSR 63N
	APSR 68N	193	Every fourth rib kept in place
	APSR 69N	360	Only two ribs kept in place

Table 5.1: Summary of APSR Tests with Fortrac Grid Reinforcements.

Stress Ratio, $R = \sigma_1/\sigma_3$	Half-Length, $L/2 = 360$ mm		Half-Length, $L/2 = 193$ mm		Half-Length, $L/2 = 109$ mm	
	Centerline Load $F_R$ (kN/m)	Reinforcement Strain, $\epsilon_f$ (%)	Centerline Load $F_R$ (kN/m)	Reinforcement Strain, $\epsilon_f$ (%)	Centerline Load $F_R$ (kN/m)	Reinforcement Strain, $\epsilon_f$ (%)
5	0.6858	0.0343	0.9904	0.0495	0.3771	0.0188
6	2.2946	0.1147	2.5991	0.1299	1.1157	0.0558
7	4.5759	0.2289	4.5531	0.2276	2.1503	0.1075
8	7.3332	0.3667	6.7607	0.3380	3.4170	0.1708
9	10.020	0.5010	9.0107	0.4505	4.67 <sup>†</sup>	0.2335

<sup>†</sup> Based on extrapolation from  $R = 8.72$ .

Table 5.2: Summary of APSR Measurements of Load-Transfer in Fortrac 110/30-20 Geogrid Reinforcements.

Inclusion Half-Length, L/2 (mm)	Apparent Stiffness, $E_a = dF_R/d\epsilon_{xx}$ (kN/m)	Offset Strain $\epsilon_{xx}^*$ (%)	Least Squares Coefficient, $r^2$
360	980	0.09	0.984
193	414	0.08	0.995
109	170	0.07	0.987

Table 5.3: Linear Regression Analyses of Tensile Loads in Fortrac 110/30-20 Grids as Functions of Average Lateral Strain in Soil.

Grid Length (mm)	Intact Grid		Modified Grid	
	Number of Ribs	Spacing between Ribs (mm)	Number of Ribs	Spacing between Ribs (mm)
193	8	23.8 <sup>†</sup>	2	95.2
360	15	23.8	2	190

<sup>†</sup> All distances are center-to-center and based on actual measurements.

Table 5.4: Geometric Characteristics of Intact and Modified Fortrac 110/30-20 Grid Inclusions.

Property	Sand	Fortrac 80/30-20	Fortrac 110/30-20
Modulus	Distribution shown in Figure 4.22b	$E_f = 1.4 \times 10^6$ kPa	$E_f = 2.0 \times 10^6$ kPa
Poisson's Ratio	Distribution shown in Figure 4.22a	$\nu_f = 0.2$	$\nu_f = 0.2$
Lateral Dimension	$m = 570$ mm	$f = 1$ mm	$f = 1$ mm
Coefficient $\beta$	-	$\beta = 0.65$	$\beta = 0.65$

Table 5.5: Input Parameters for Shear-lag Predictions of Load Pickup in Fortrac Geogrid Reinforcements.

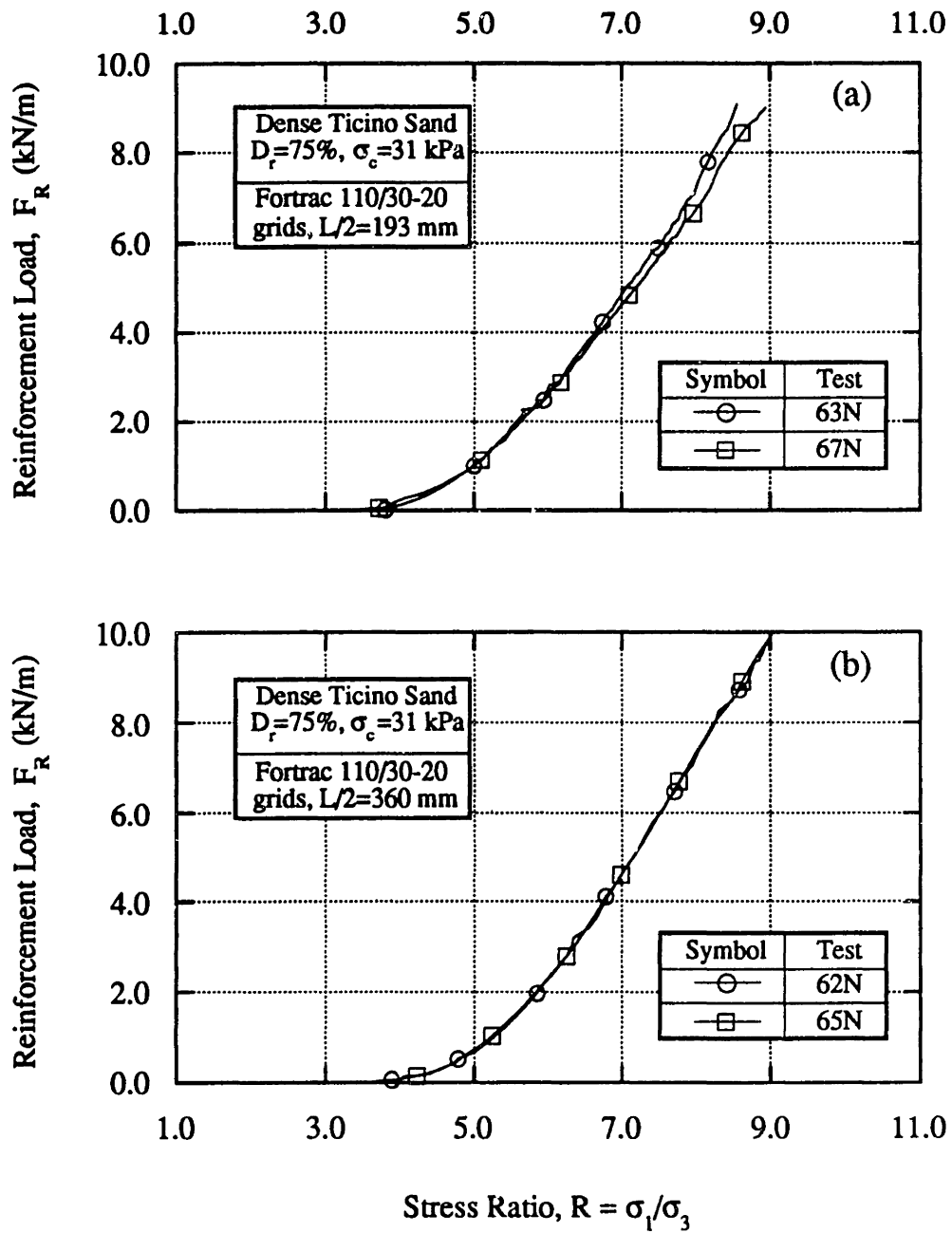


Figure 5.1: Repeatability of Externally Measured Tensile Loads in Fortrac 110/30-20 Grid Inclusions.

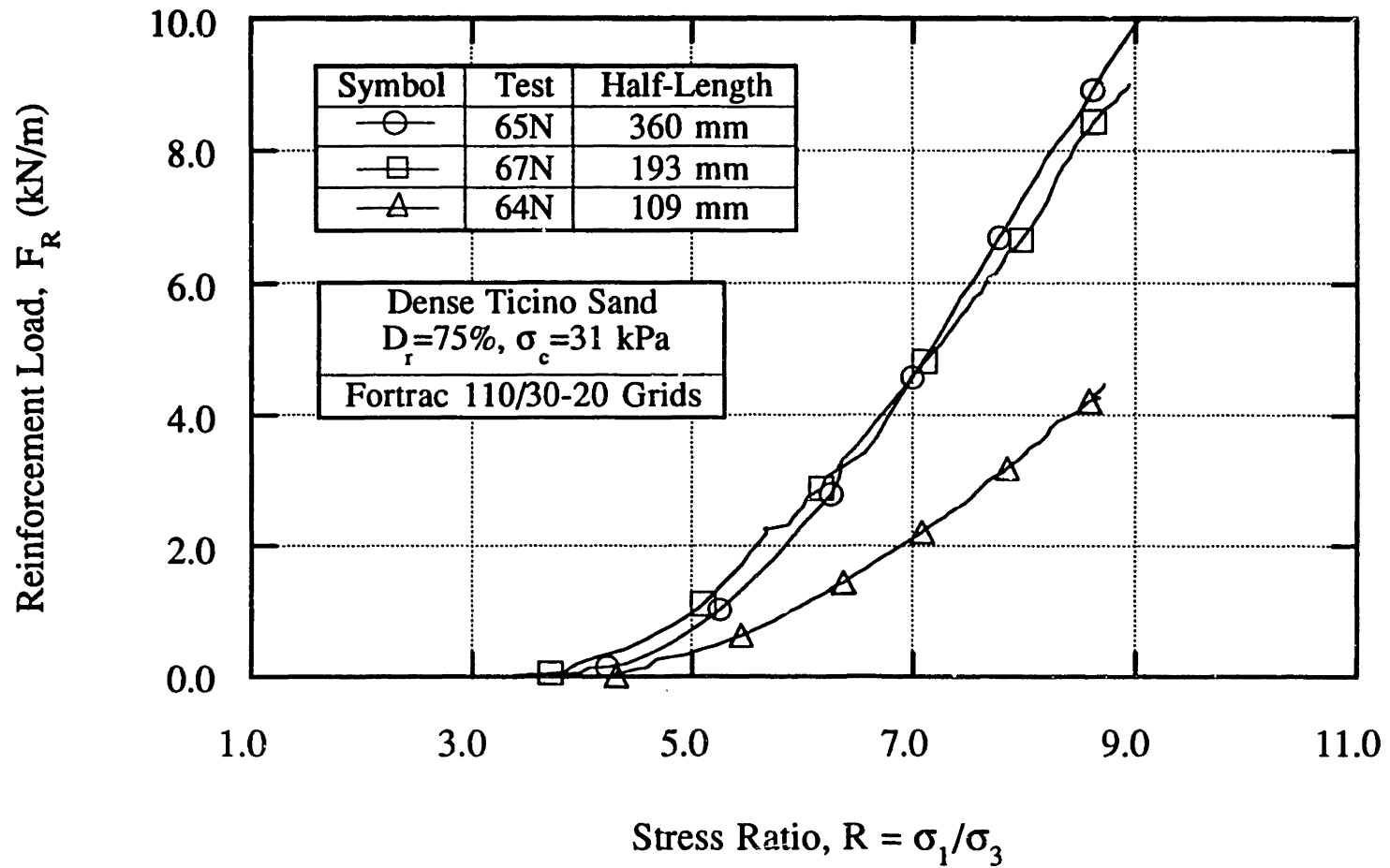


Figure 5.2: Effect of Inclusion Length on Tensile Loads Measured in the APSR Cell for Fortrac 110/30-20 Grid Reinforcements.



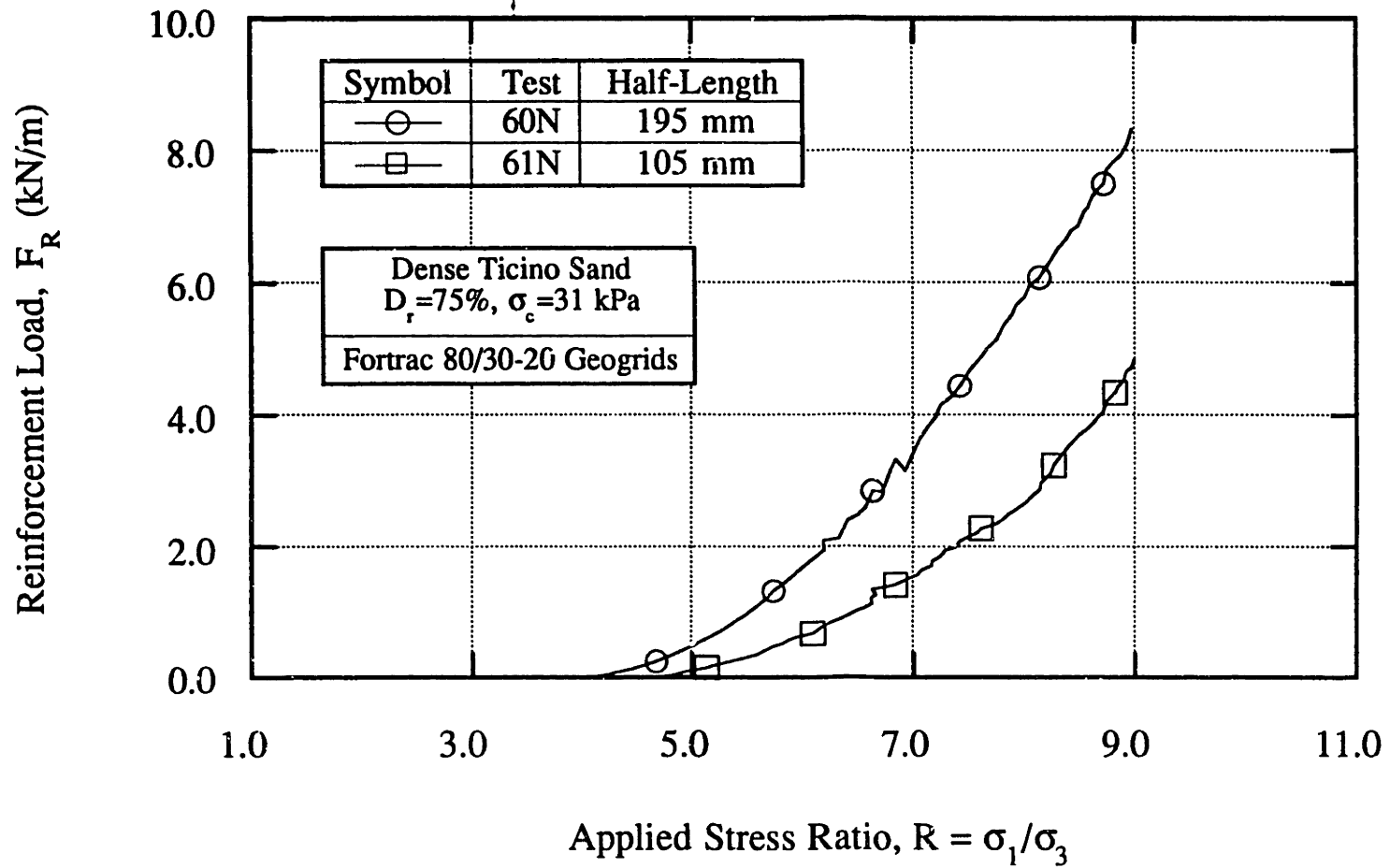


Figure 5.3: APSR Measurements of Load-Transfer for Fortrac 80/30-20 Grid Reinforcements.

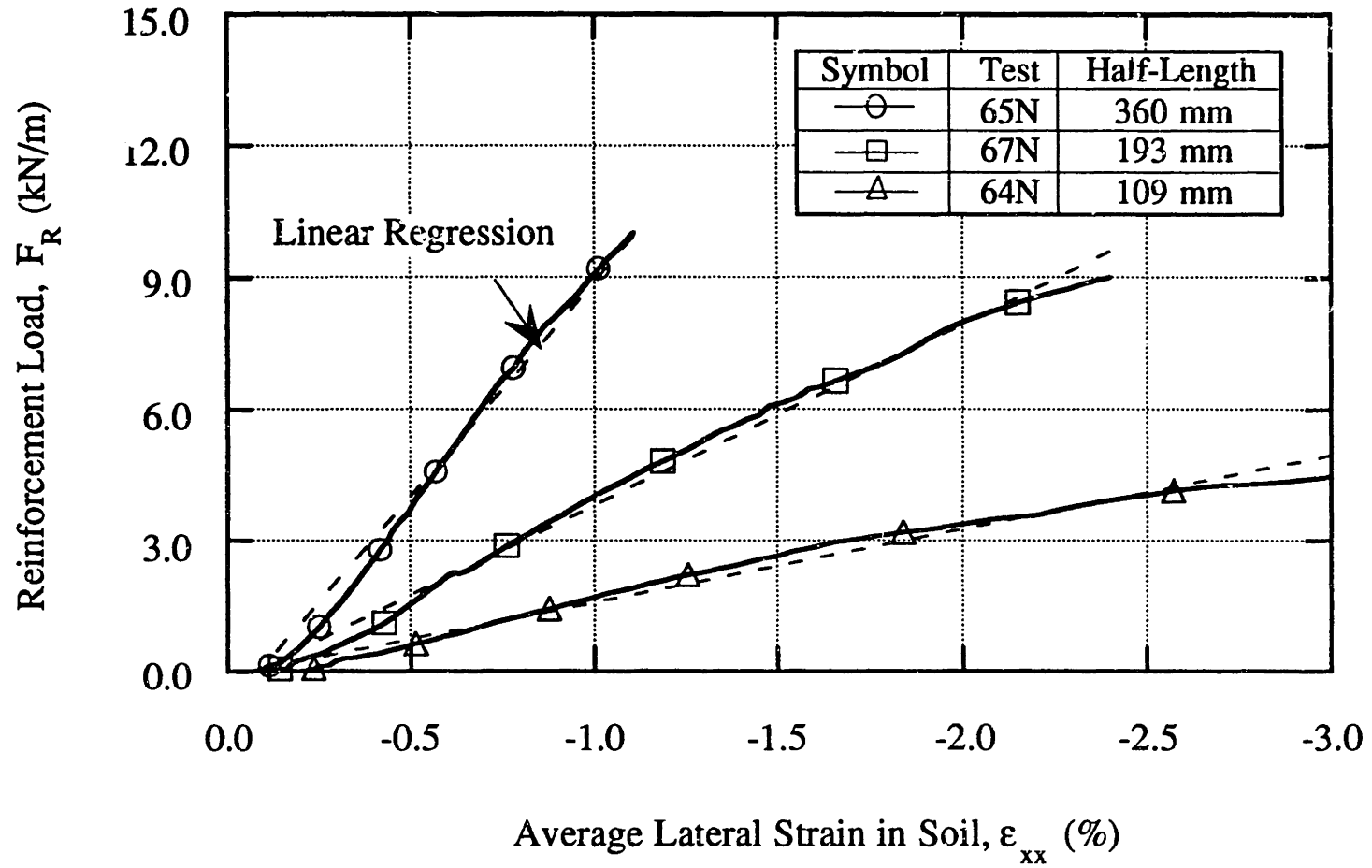


Figure 5.4: Relationship between Exit Point Tensile Loads and Lateral Strains in Soil Matrix for Fortrac 110/30-20 Grid Reinforcements.

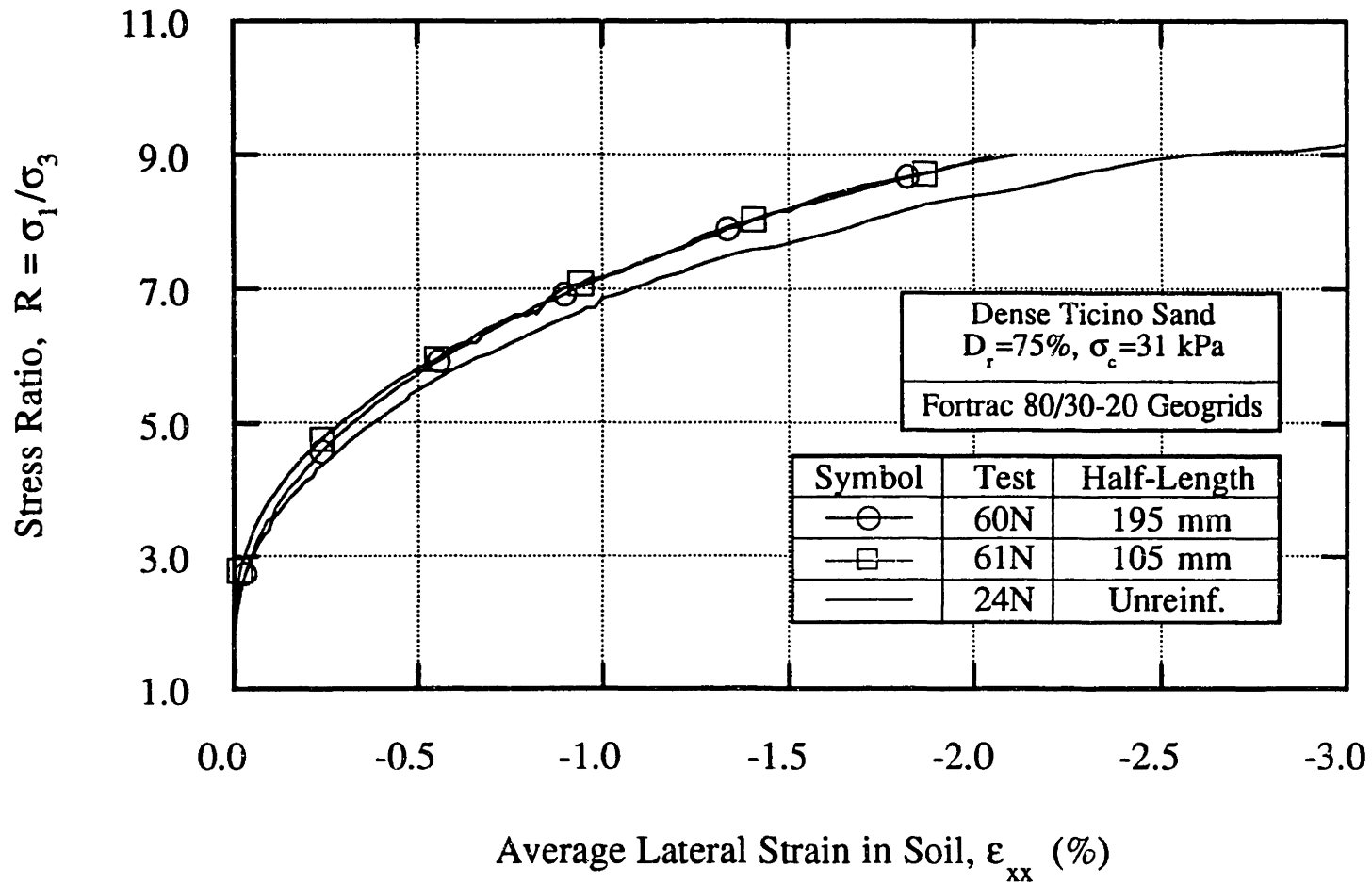


Figure 5.5: Relationship between Applied Stress Ratio and Lateral Strain in Soil for Fortrac 80/30-20 Grid Reinforcements.

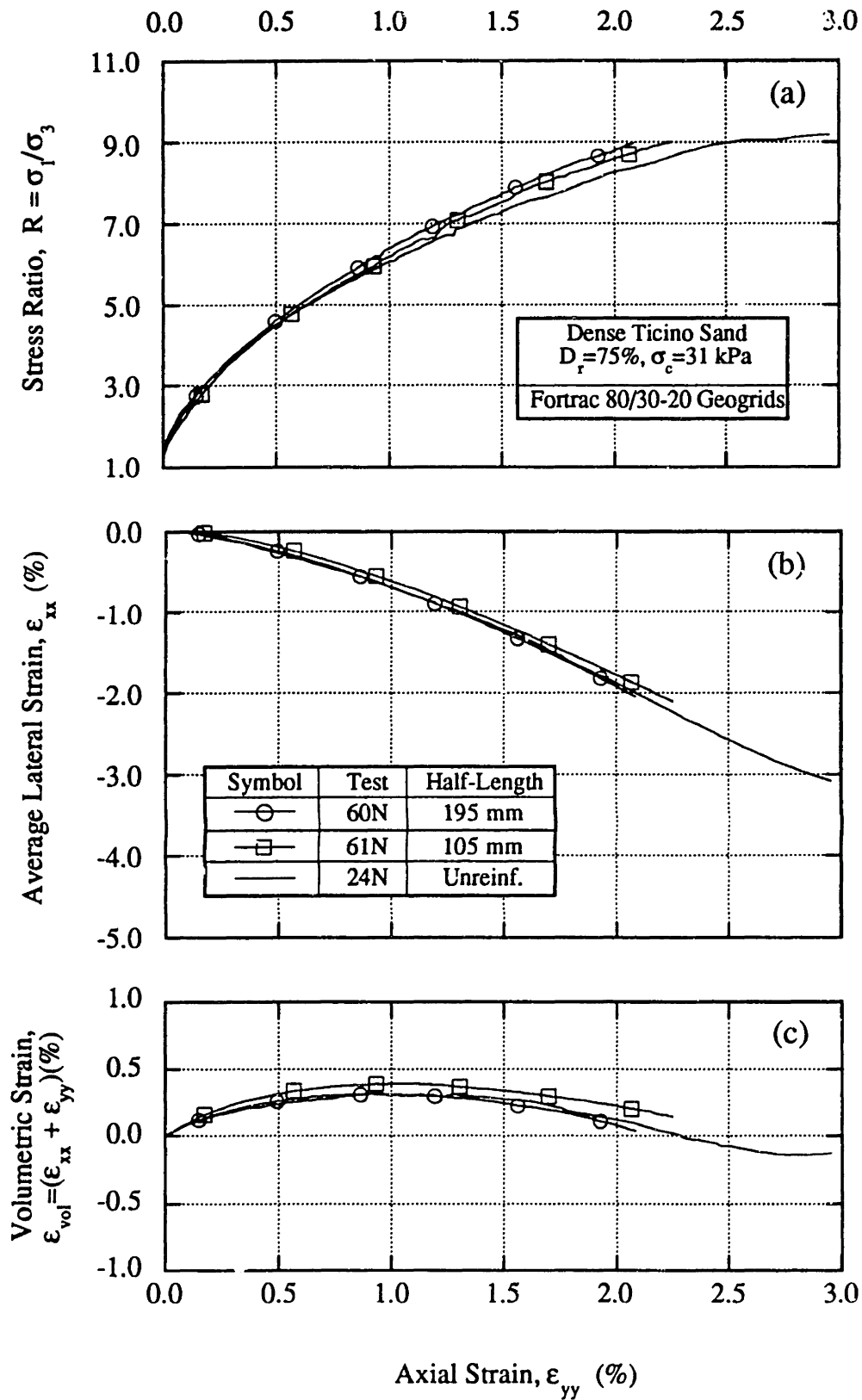


Figure 5.6: Shear Behavior of Medium Dense Ticino Sand Reinforced with Fortrac 80/30-20 Grid Inclusions.

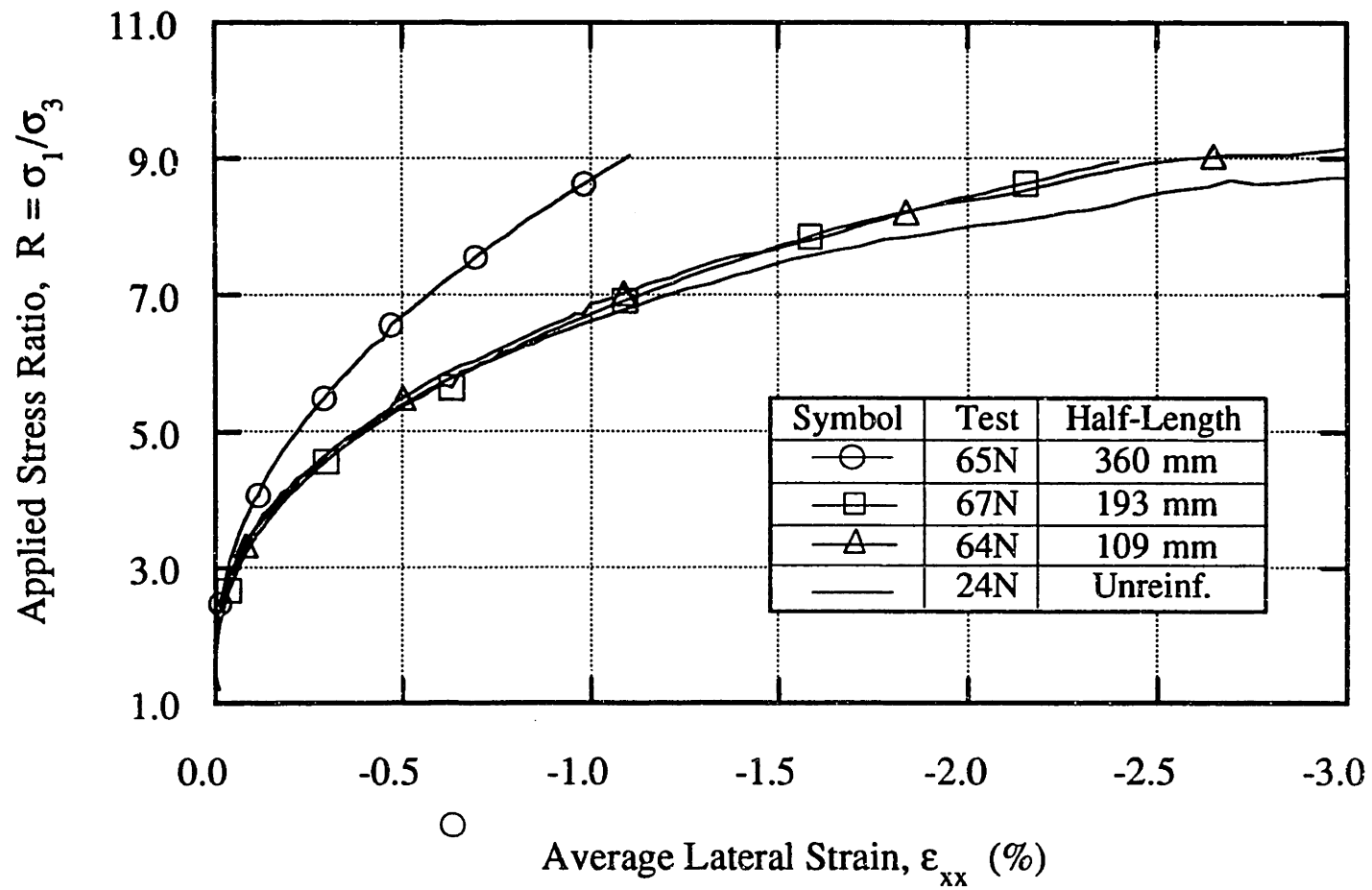


Figure 5.7: Effectiveness of the Longest Fortrac 110/30-20 Grid Inclusion in Reducing Lateral Deformations in the Sand Matrix.

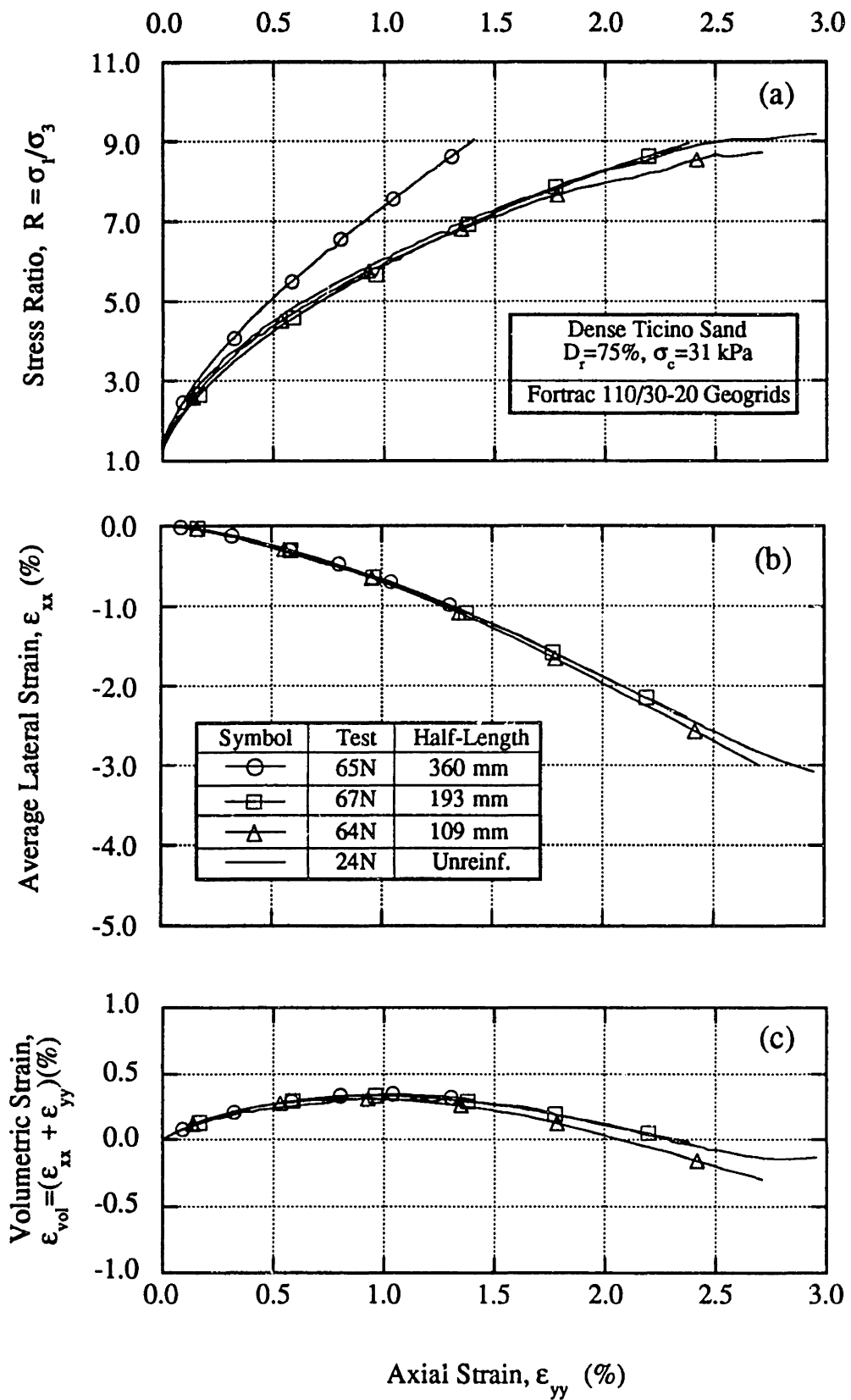


Figure 5.8: Effects of Fortrac 110/30-20 Grid Reinforcements on Boundary Measurements of Shear Behavior of Ticino Sand.

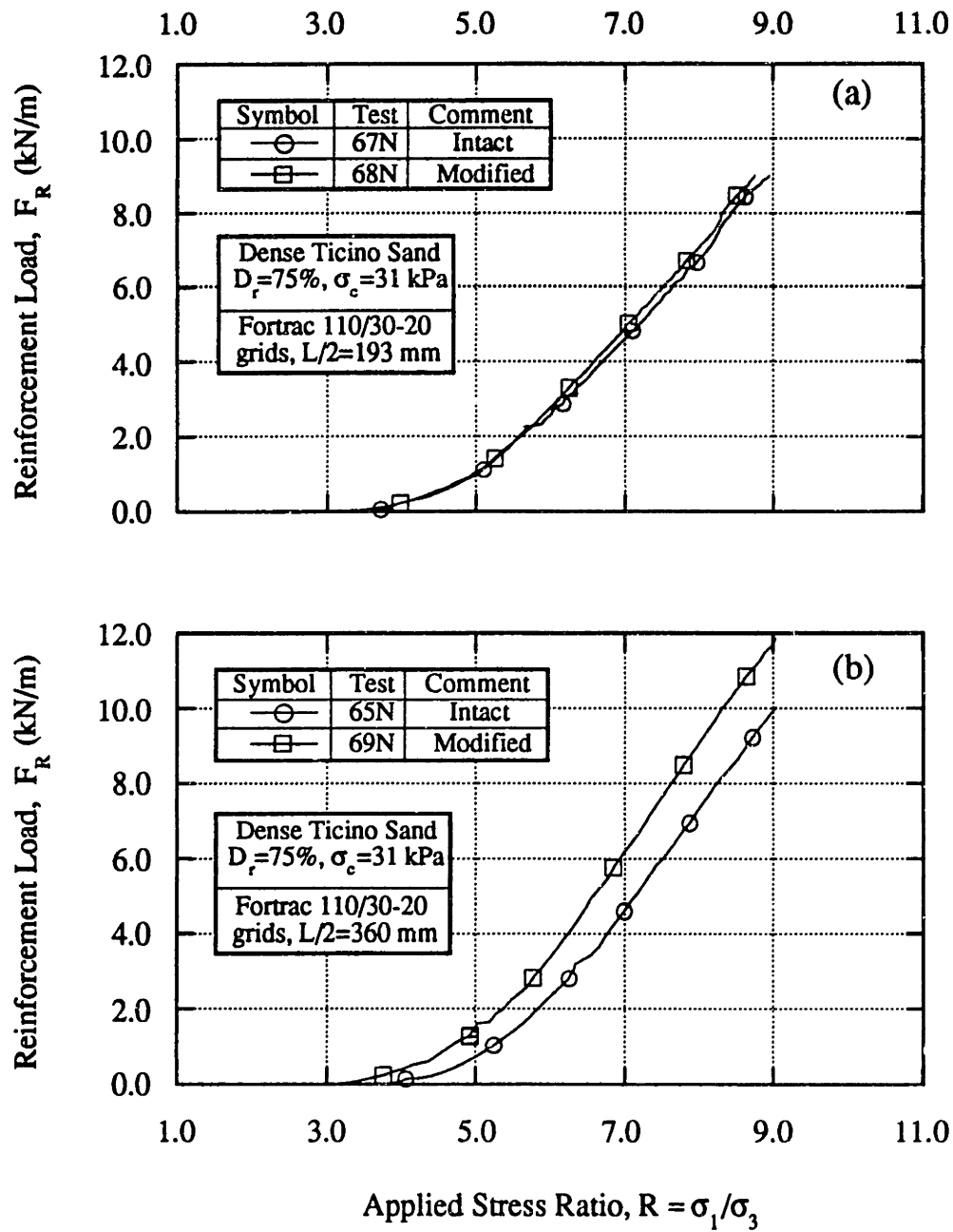
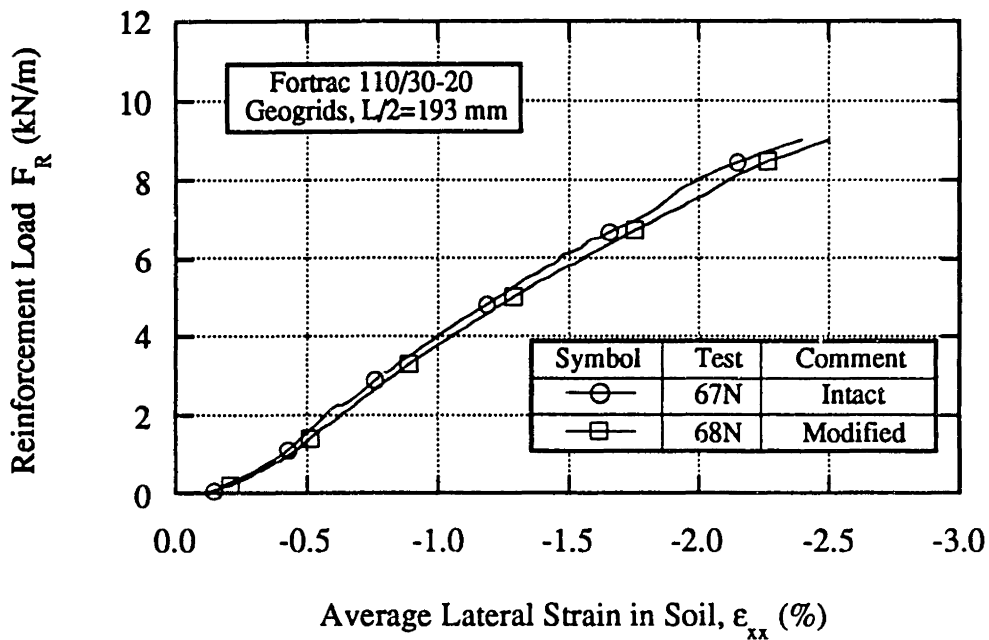
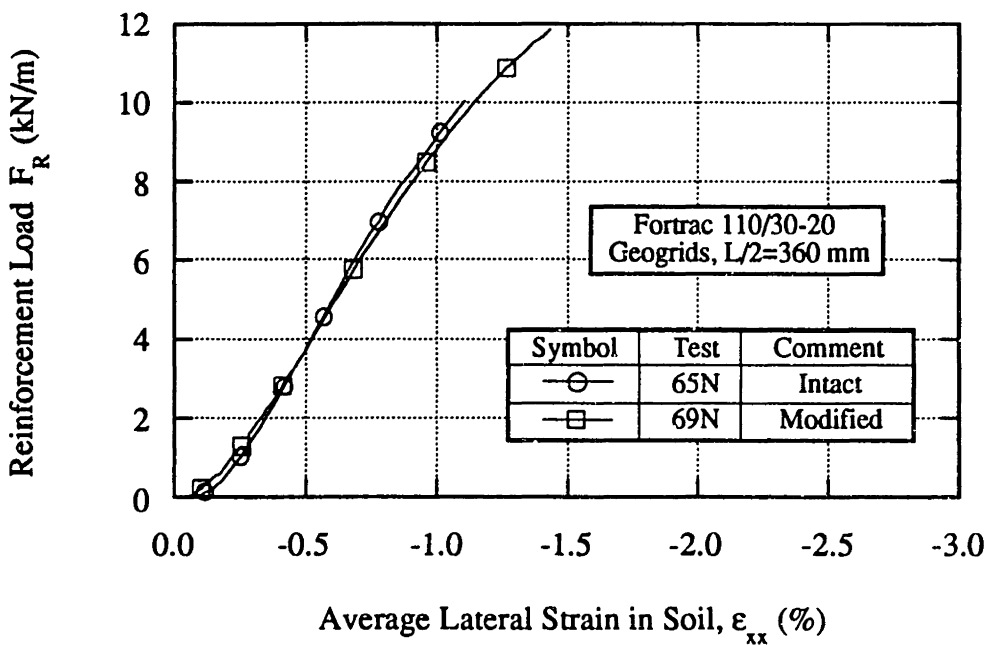


Figure 5.9: Effects of Transverse Members on Load-Transfer Behavior of Two Fortrac 110/30-20 Geogrid Inclusions.



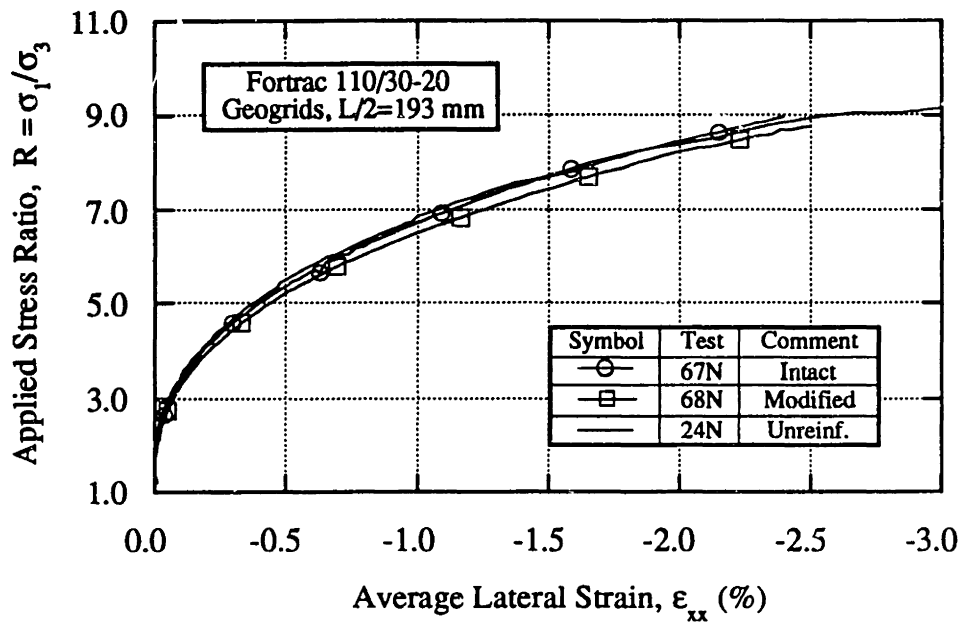
(a) 193 mm Long Grid Inclusions.



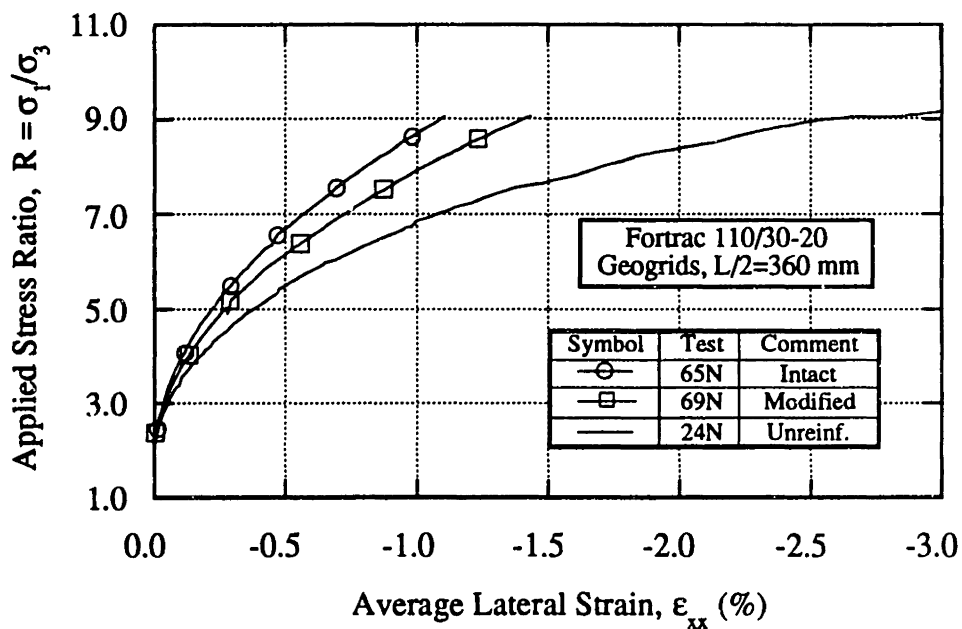
(b) 360 mm Long Grid Inclusions.

Figure 5.10: Relationship between Inclusion Tensile Loads and Average Lateral Strain for Intact and Modified Grid Reinforcements.





(a) 193 mm Long Grid Inclusions.



(b) 360 mm Long Grid Inclusions.

Figure 5.11: Effect of Transverse Grid Elements on the Average Lateral Strains in the Soil Matrix.

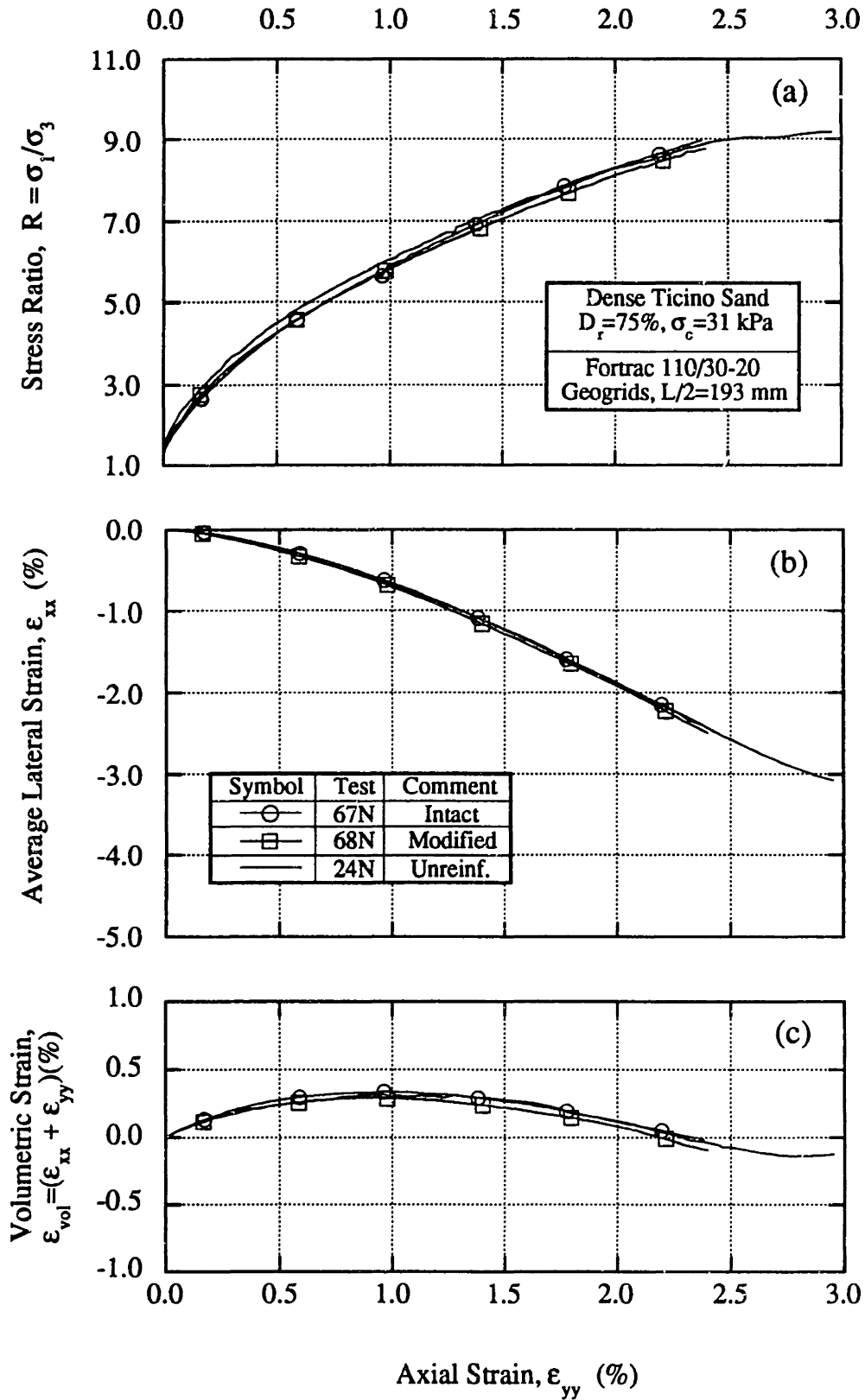


Figure 5.12: Effect of Transverse Ribs on Externally Measured Shear Behavior of Dense Ticino Sand for 193 mm Long Grid Inclusions.

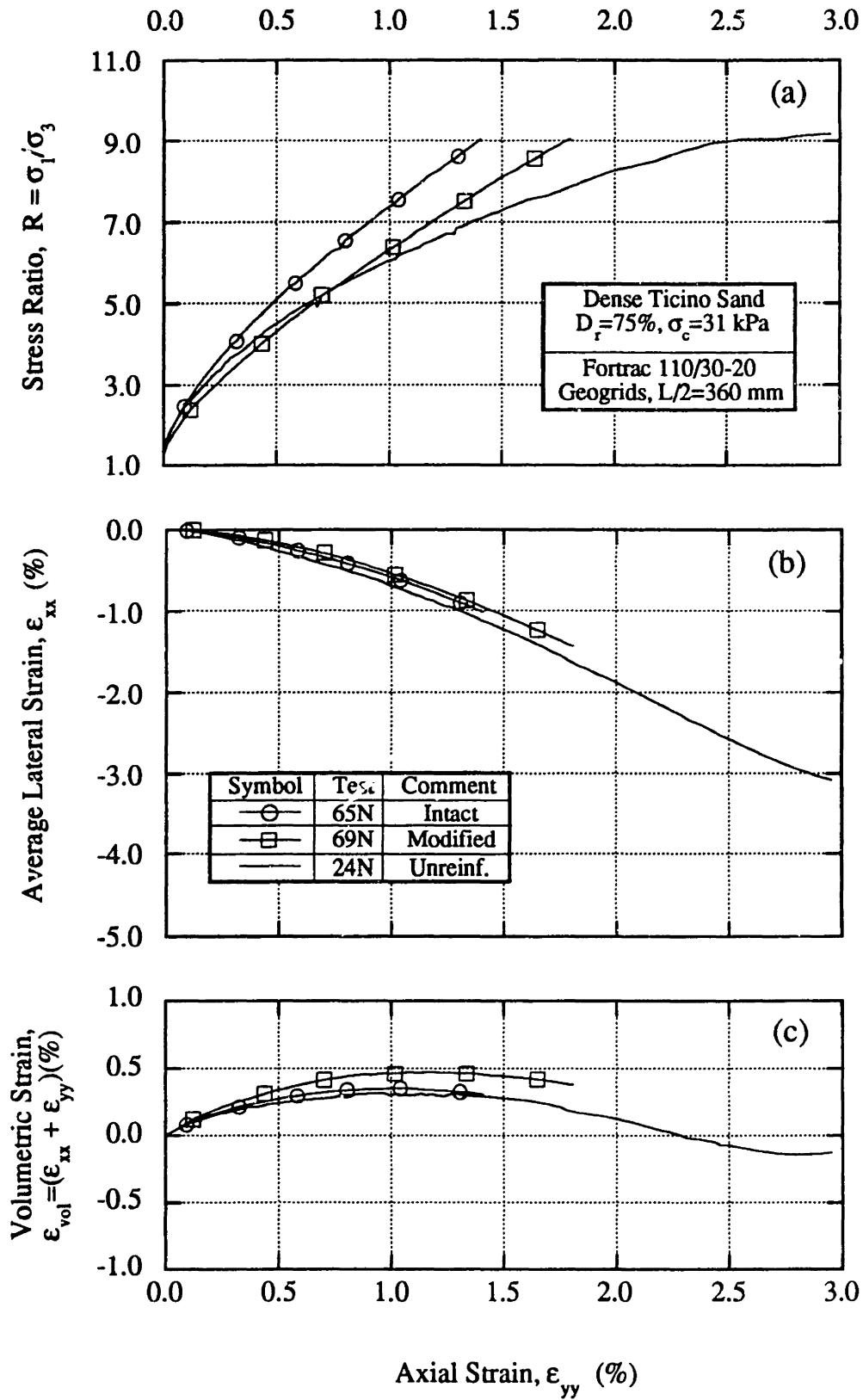


Figure 5.13: Effect of Transverse Ribs on Externally Measured Shear Behavior of Dense Ticino Sand for 360 mm Long Grid Inclusions.

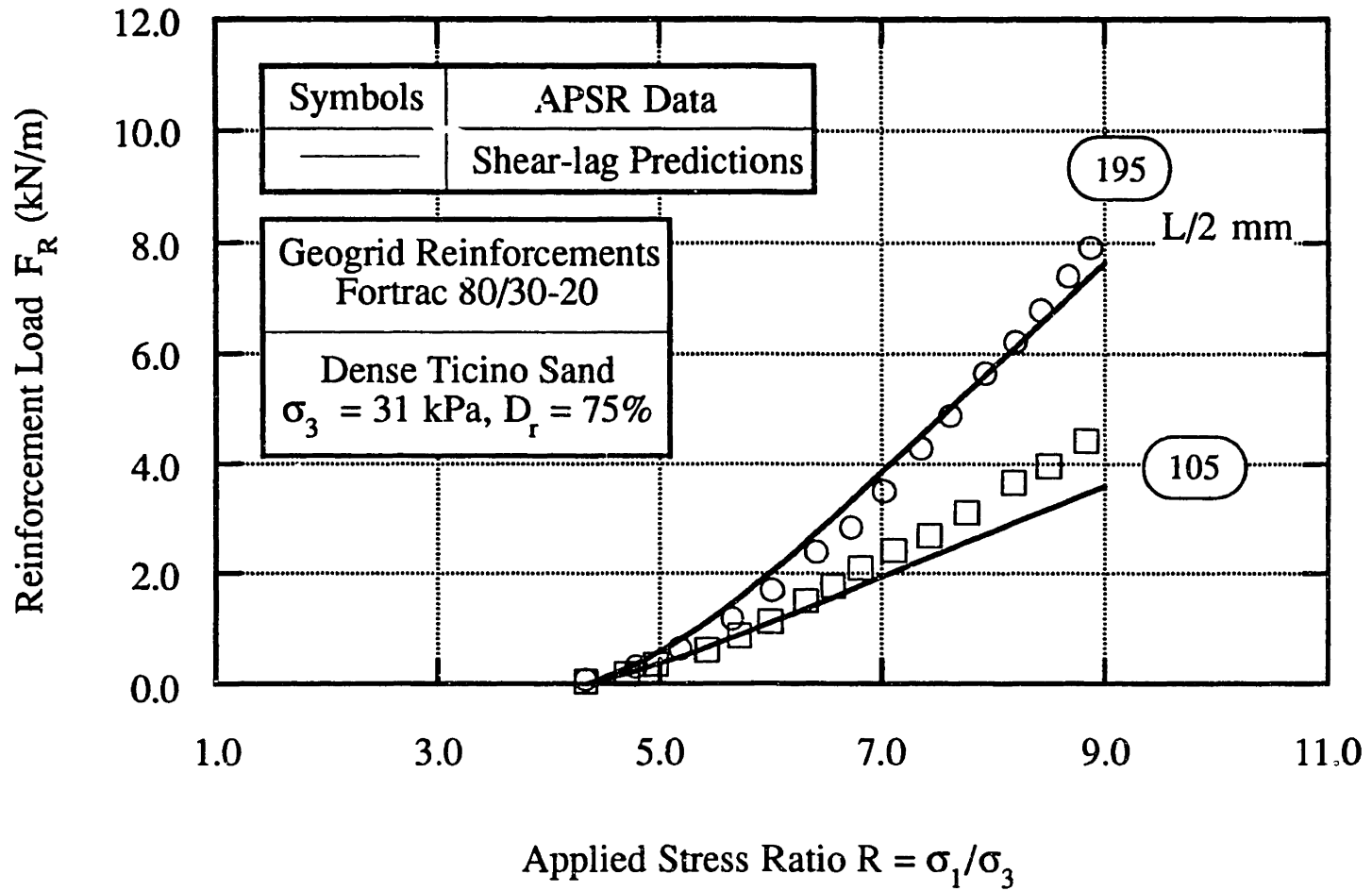


Figure 5.14: Shear-Lag Predictions vs. Measured Inclusion Tensile Loads for Fortrac 80/30-20 Inclusions of Different Lengths.

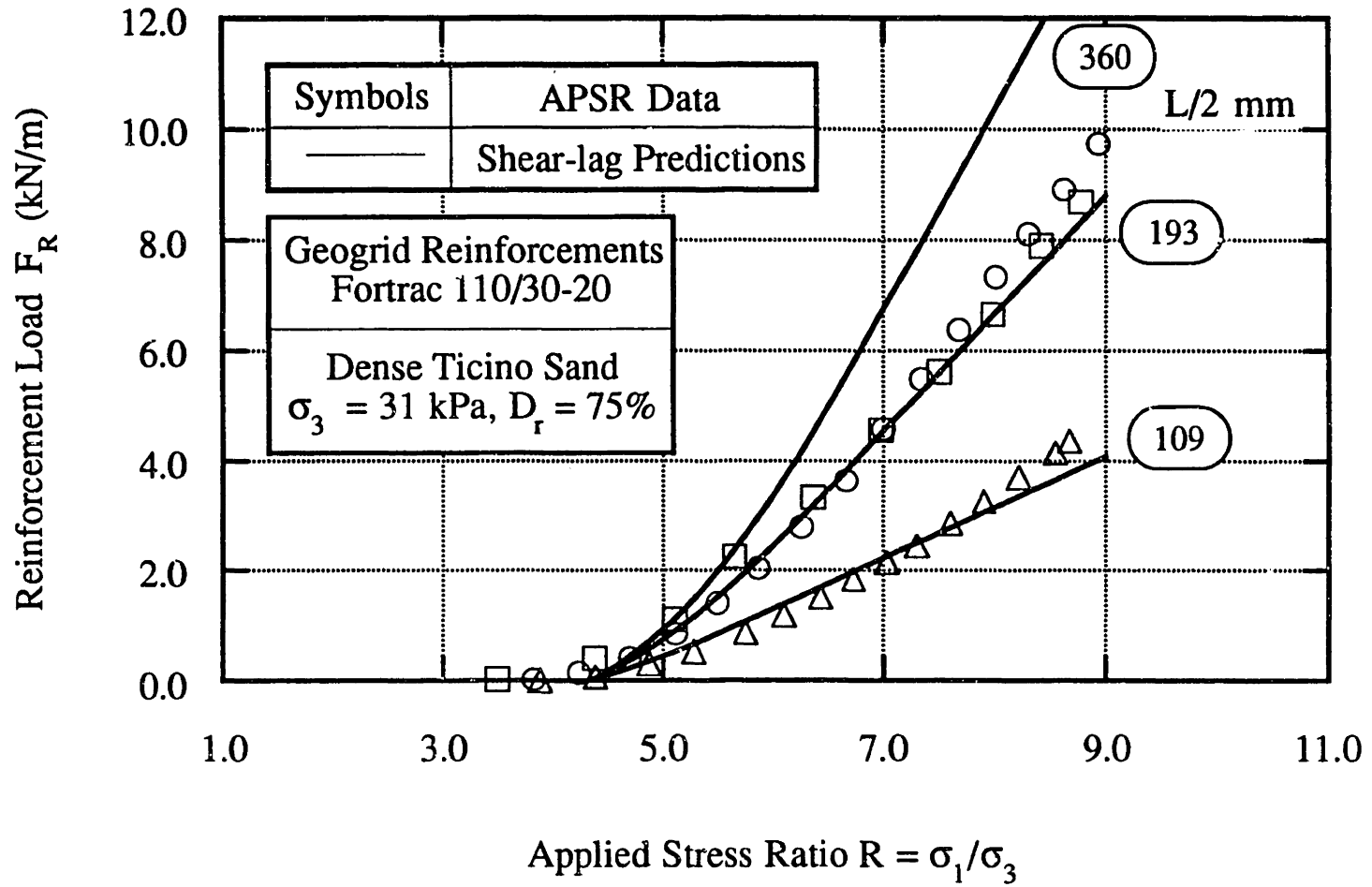


Figure 5.15: Comparison of Shear-lag Predictions and APSR Measurements of Load-Transfer for Fortrac 110/30-20 Grid Reinforcements.

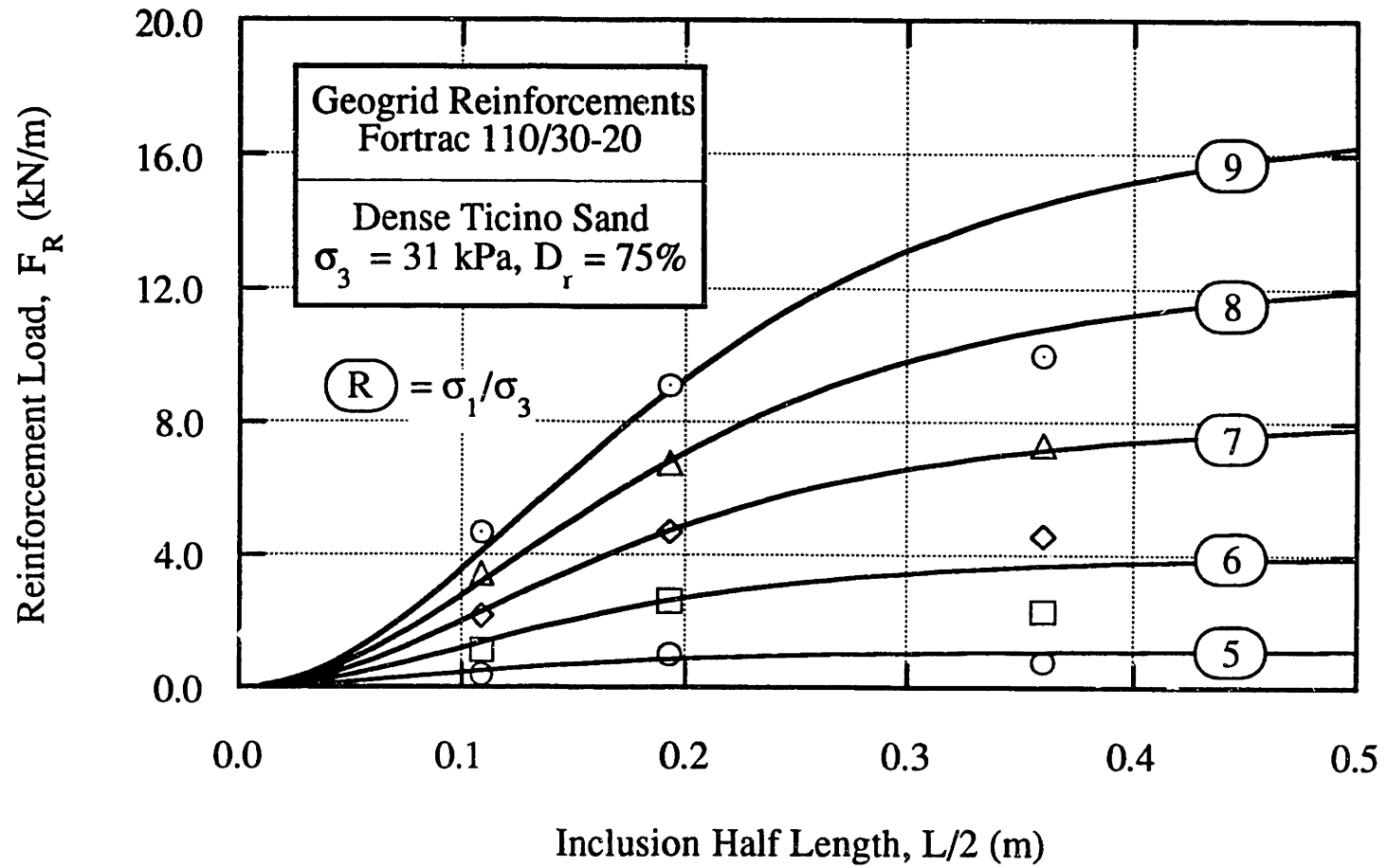


Figure 5.16: Effects of Inclusion Length on Predicted and Measured Load Transfer for Fortrac 110/30-20 Grid Reinforcements.

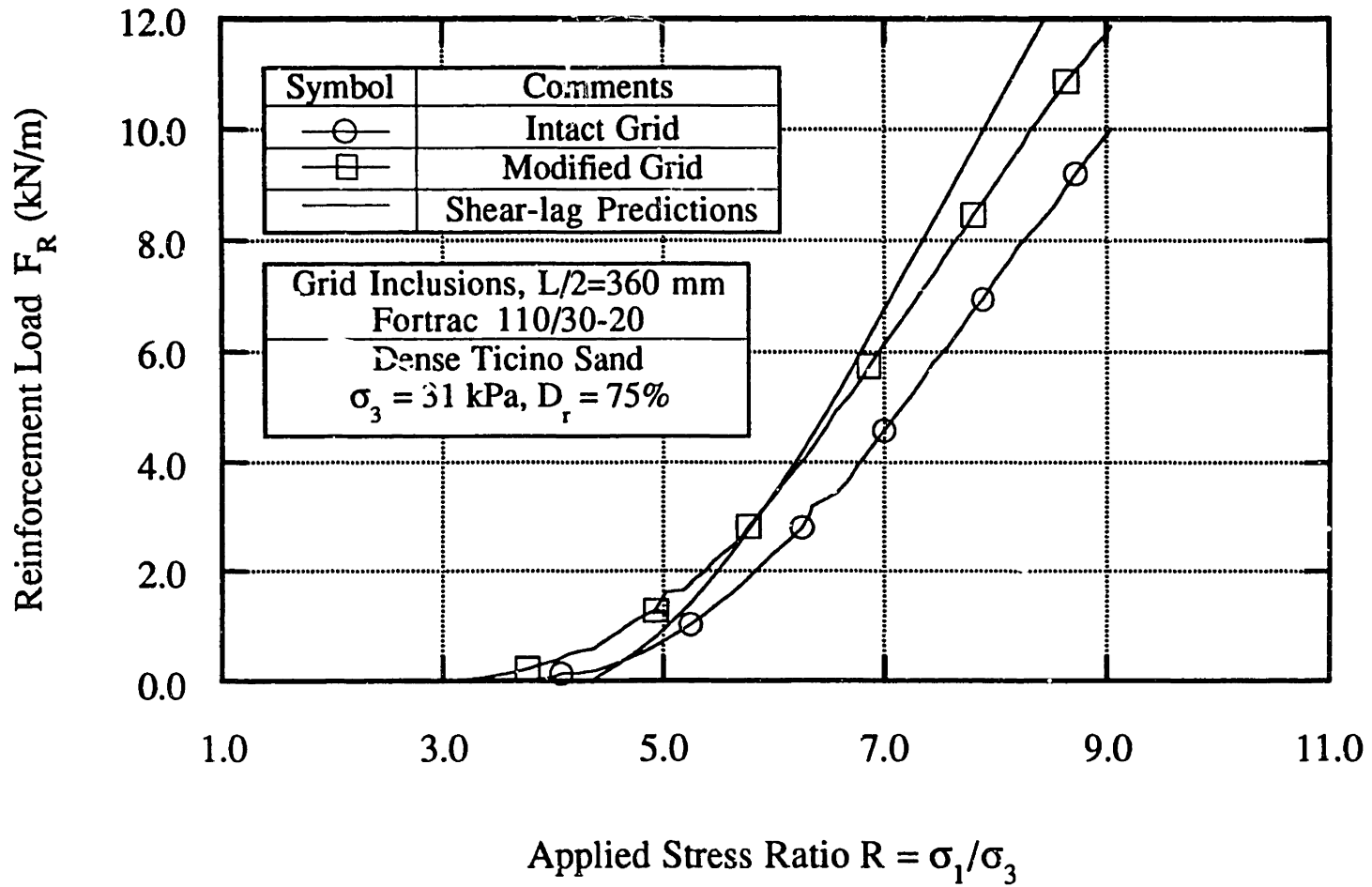


Figure 5.17: Comparison of Shear-lag Predictions and APSR Measurements of Centerline Loads in Modified and Intact Fortrac 110/30-20 Reinforcement.

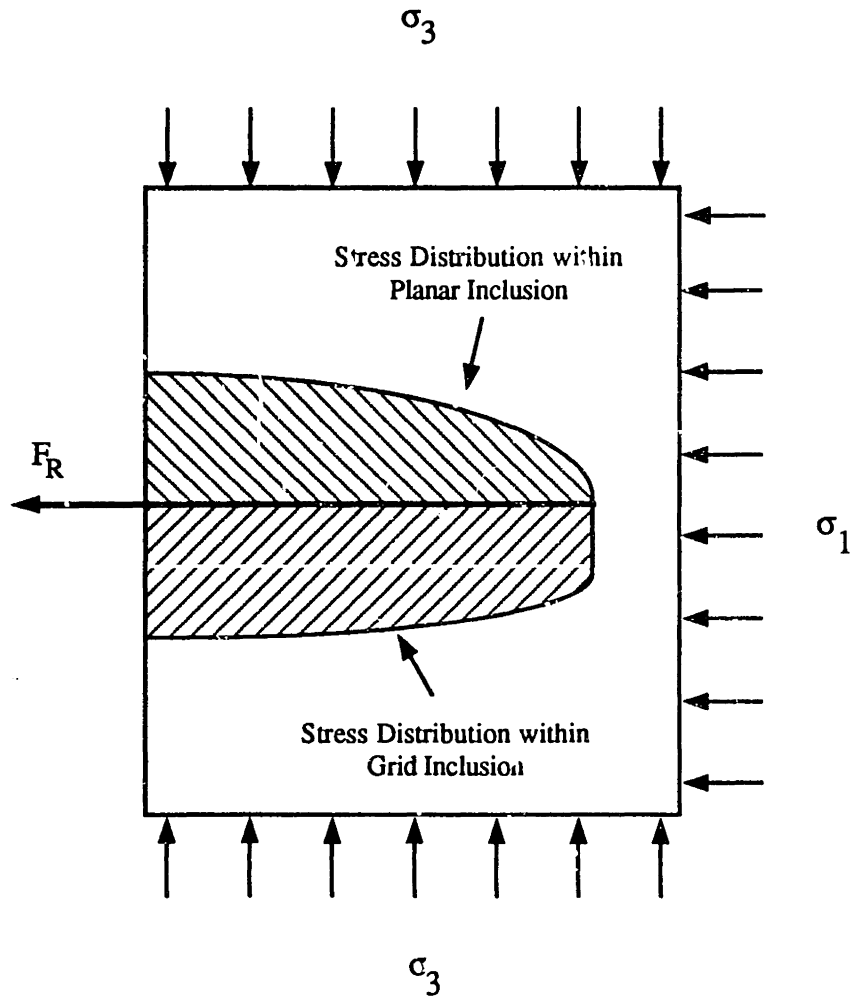


Figure 5.18: Hypothetical Differences between Distributions of Tensile Stress within Planar and Geogrid Inclusions.



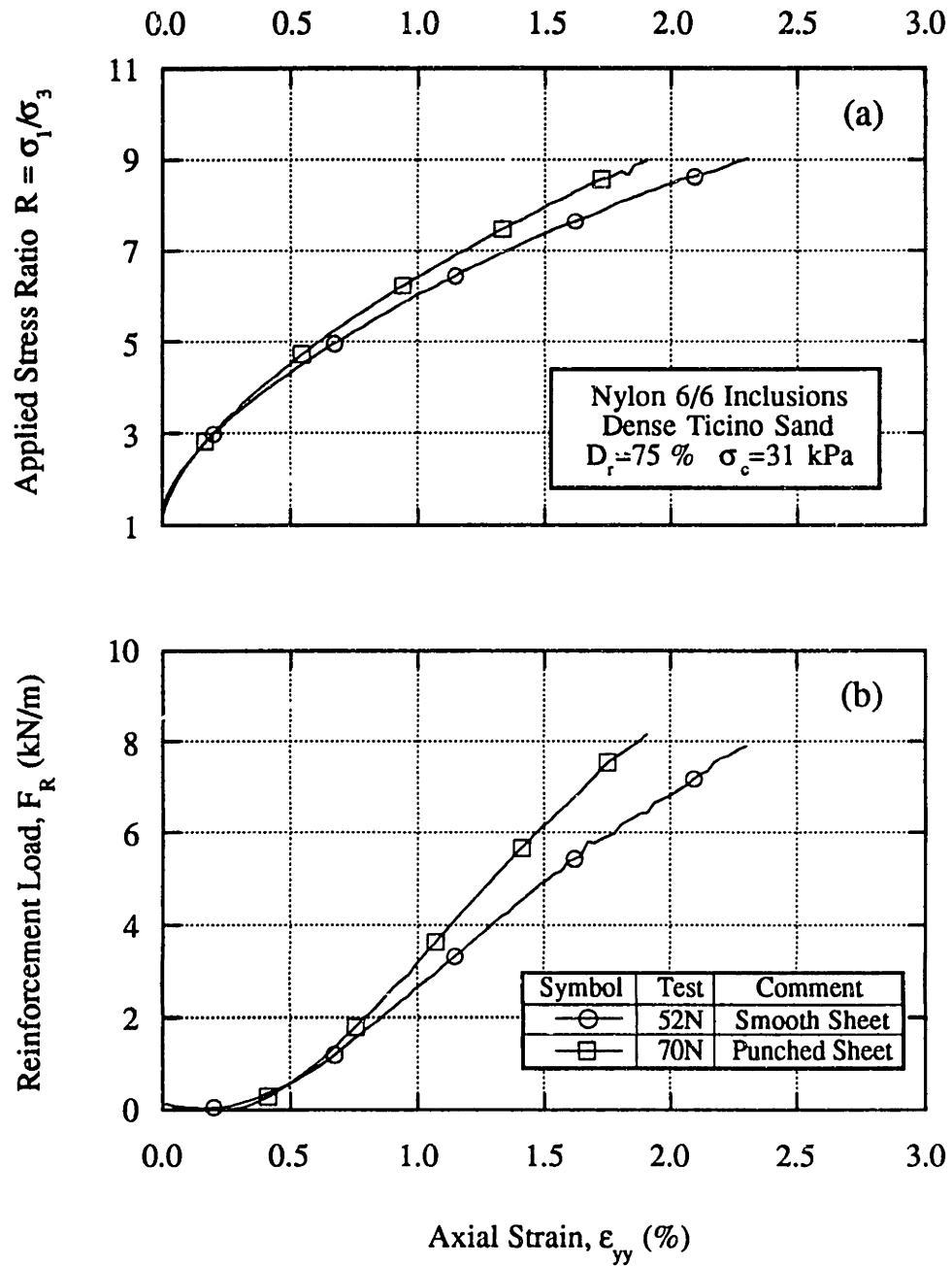


Figure 5.19: Externally Applied Stress Ratio and Reinforcement Tensile Load as Functions of Axial Strain for Smooth and Punched Nylon Sheets.

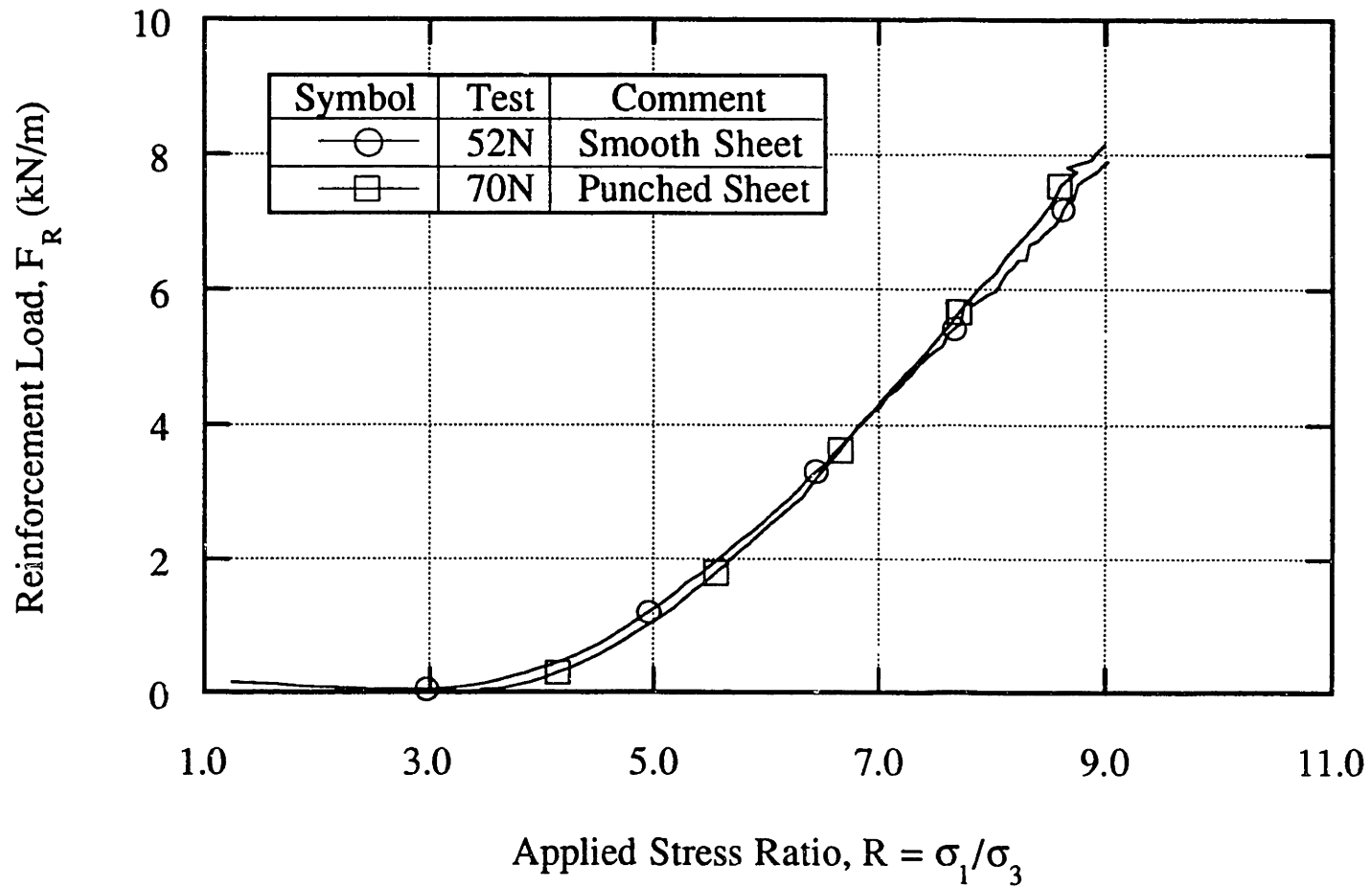


Figure 5.20: Comparison between Load-Transfer Behavior of 360 mm Long Smooth and Punched Nylon 6/6 Sheet Inclusions.

# Chapter 6: A Geogrid Reinforced Wall Case Study

## 6.1. Introduction

The APSR test is an element level test that simulates the soil-reinforcement interaction at working stress levels under interpretable boundary conditions. The previous chapters have described APSR measurements of tensile stresses in planar and grid reinforcements due to shearing of the surrounding soil, and their interpretation using shear-lag analysis. The APSR cell offers a new experimental capability for evaluating the performance of prototype-scale, geosynthetic-reinforced structures under working stress conditions. However, in order to use the results of an element test to interpret and predict behavior of full scale structures, a number of simplifying assumptions are generally required. The research effort needed to formulate and validate these assumptions is outside the scope of this thesis. This chapter illustrates the use of APSR measurements of load-transfer for estimating the maximum reinforcement loads within a reinforced soil wall under working stress conditions, and compares these estimates with measurements from a well documented case study. Further research is required for applying APSR data in predicting distribution of the tensile loads within the reinforcement layers.

Section 6.2 describes the case study which involved the construction of two well instrumented geogrid-reinforced test walls as part of a long-term research project at the Royal Military College of Canada (RMC). Sections 6.3 and 6.4 summarize the magnitudes and distributions of strains within the geogrid reinforcements, and compare

these results with predictions of load-transfer using current design methods. Finally, Section 6.5 compares the backfill and reinforcing inclusions used in the case study with the materials used in the APSR cell, and shows how the maximum reinforcement strains can be estimated from the APSR cell data.

## 6.2. RMC Reinforced Retaining Walls

The Royal Military College (RMC) retaining wall test program (Bathurst et al., 1987) is one of the very few well documented, large scale, geosynthetic reinforced wall case studies reported in the literature. In 1987, a group of prominent researchers in the field of geosynthetics undertook a "prediction exercise" under the auspices of the NATO Scientific Affairs Division. The goal of this exercise was to assess critically the state-of-the-art in the design and analysis of polymeric reinforced soil retaining structures. Two large scale geogrid reinforced soil walls were built, instrumented, and tested in the laboratory at the Royal Military College of Canada. Prior to construction, details of the proposed tests and material properties were sent to a group of predictors who were asked to make Class A (Lambe, 1973) predictions of the performance of the reinforced walls at various surcharge loads and elapsed times after initial construction. The results of the prediction exercise were the center of discussion at a workshop organized by the sponsors (Jarrett and McGown, 1988). The RMC wall case study was selected for the present comparison for the following reasons:

1. The RMC reinforced retaining walls were designed according to the code guidelines established for working stress designs of such structures. Post construction assessments suggested that the walls would have remained

stable for many years under the highest surcharge used in the tests (Bathurst et al., 1987).

3. The backfill and reinforcement materials used to construct the wall were similar to some of the materials used in the APSR research.
2. Several of the measurements were made by more than one independent means: for example, the reinforcement strains were obtained from extensometers and also measured directly using strain gages.

### 6.2.1. Description of Geogrid Reinforced Walls

Figure 6.1 and 6.2 show the general layout of the RMC model walls. Each wall had a total height of 3.0 m and was constructed with 4 layers of geogrids, 3.0 m in length, within a uniform sand backfill. The test walls were built in the laboratory within a large structural box comprised of six heavily reinforced concrete, counterfort cantilever wall modules (Figure 6.1). The reinforced mass had a total depth of 6.0 m and width of 2.4 m and was loaded by a 200 mm thick sand surcharge layer and pressurized airbags that are confined between the concrete modules and the structural steel sections located at the top of the structure. The surcharging arrangement was capable of applying a vertical pressure equivalent to 3.0 m of fill. The inside walls of the structure were covered with a composite plywood/clear Plexiglas/polyethylene sheeting which acts to reduce sidewall friction. Shear box tests showed that the sand/sidewall friction interface had a fully-mobilized friction angle of approximately 15° (Bathurst et al., 1988).

Two large scale model retaining walls were constructed, each using four layers of high density polyethylene geogrid reinforcement. Reinforcement layers were spaced 0.75 m apart and attached to the plywood facing panels. Figure 6.2 shows the general arrangement for Test 1 which used twelve, 0.75 m high, articulated facing panels to

achieve 3.0 m height. The second test wall was constructed using the same general arrangement with exception that the wall facings comprised three panels, each 3.0 m in height. The reinforcement location for both tests were standardized by designing the walls using the U.K. Department of Transportation's technical memorandum *Reinforced Earth Retaining Walls and Bridge Abutments for Embankments, BE 3/78* (Department of Transportation, 1973). The length of all four reinforcement layers was limited to 3.0 m which was sufficient to prevent any pull out type of failure. The design is based on working stress conditions at a surcharge pressure of 12 kPa.

### 6.2.2. Material Properties

The soil used was a uniformly graded, clean (less than 0.3% of particles less than 75  $\mu\text{m}$ ) sand with angular to subangular quartz and feldspar particles, and a mean particle size,  $D_{50} = 1.2 \text{ mm}$ . The measured maximum and minimum dry densities were 19.2 and 15.9  $\text{kN/m}^3$ . The average dry density of the sand backfill was typically 17.6  $\text{kN/m}^3$ , giving a relative density,  $D_r = 52\%$ . The average moisture content at the time of placement and compaction was between 2 to 3%. The shear resistance of the backfill material was determined from standard direct shear box tests ( $3.6 \times 10^3 \text{ mm}^2$  in plan area) performed at RMC and large direct shear box tests ( $3.93 \times 10^4 \text{ mm}^2$  in plan area) carried out at Oxford University. At densities comparable to those measured 'insitu', the tests gave peak (secant) friction angles that varied from  $\phi' = 53^\circ$  at a vertical confining stress of 12 kPa, to  $\phi' = 40^\circ$  at  $\sigma_v \approx 120 \text{ kPa}$  (Bathurst et al., 1989). Standard consolidated-drained triaxial tests at dry density,  $\gamma_d = 16.8 \text{ kN/m}^3$  ( $D_r = 27\%$ ) gave a peak friction angle,  $\phi' = 41^\circ$  with a linear failure envelope between confining pressures  $30 < \sigma_3 < 60 \text{ kPa}$ . The value of the elastic shear modulus,  $G_m$ , was estimated to vary between 5000 to 2300 kPa from the direct shear test data using a

procedure described in a companion paper published in the conference proceedings (Jewell, 1987) Table 6.1 summarizes the important engineering properties of the soil.

Four layers of reinforcement were used in the test, each comprising 3.0 m long strips of high density polyethylene Tensar SR2 geogrid. Figure 6.3 shows the isochronous load-extension curves at 20° C for this material. The figure gives very little data at low loads (< 15 kN/m, say), but is the only information readily available for this particular material. The low strain ( $\epsilon < 1\%$ ) reinforcement stiffness estimated from these curves ranges from  $J_f = 625$  kN/m, at 100 hrs sustained loading, to  $J_f = 550$  kN/m at 1000 hrs loading (Jewell, 1987).

### 6.2.3. Test Configurations and Procedures

The model walls were constructed with a central 1.0 m wide instrumented section and two 0.7 m wide edge sections. This construction was used to minimize edge-effects on the performance of the central monitored section (Bathurst et al., 1987). The construction sequence adopted for both walls was similar with the exception of the temporary facing support and release sequence:

1. Test 1 used a four stage incremental construction procedure with each panel externally supported only until the sand fill behind the panel was placed and compacted.
2. Test 2 used full support for the facing panels throughout the construction. The supports were released only after the full height of the sand fill (3 m) had been placed behind the facing units.

A foam rubber void filler was placed along all panel edges in order to prevent the panels from binding during outward movements. Prior to construction, a 250 mm thick blinding layer was placed and compacted behind the concrete floor leveling pad

shown in Figure 6.2. The plywood facing panels were seated on the concrete leveling pad and initially restrained by a wooden waling located in front of the wall at the same elevation. Subsequently, all sand backfill was placed and compacted in 125 mm lifts, covering the full length of the test facility. A vibrating plate tamper was the principal means of compaction although a hand-held hammer with tamper attachment was used to compact sand in the corners. A light pre-tensioning load of about 0.4 kN/m was applied to the geogrids prior to fill placement to ensure that the reinforcement was free of warps. The pre-tensioning was released after the grid was covered by 0.5 m of compacted sand.

#### 6.2.3.1. Test 1: Incremental Panel Facing

In this method of construction, a total of twelve 0.75 m high plywood facing panels were used to construct a 3 m high reinforced soil wall (Figure 6.2). Temporary lateral support was provided to each row of panels by a pair of timber wall beams which were bolted to the front of the test facility. Each panel row support was released after the sand backfill was placed and compacted to the top of the facing panel. The purpose of this form of construction was to mobilize progressively the inherent tensile capacity of the geogrid reinforcement as the height of the composite structure was increased (Bathurst et al., 1988). Following the construction, the incremental panel facing wall was subjected to a series of surcharge loads. The design surcharge of 12 kPa was sustained for a period of about 1000 hrs in order to observe creep behavior under the design loads. Later, the surcharge was increased to 10, 30, 40, and 50 kPa and each increment was sustained for at least 100 hrs. The maximum surcharge load of 50 kPa was maintained for 500 hrs to observe further creep deformations.



### 6.2.3.2. Test 2: Propped Panel Facing

This construction method used three timber panel units, each 3.0 m in height. The facing panels were temporarily supported at the wall base, and at 1.0 m, 1.75 m, and 2.5 m above the base of the wall by the same system of horizontal timber waling described in Section 6.2.3.1. The wall beams were released simultaneously only after the sand backfill had been placed and compacted to the full panel height of 3.0 m. In a manner similar to the incremental panel wall, the propped panel wall was surcharged using the airbag system installed at the top of the RMC test facility. A surcharge load of 12 kPa was initially applied for a period of about 500 hrs and later a maximum surcharge of 50 kPa was maintained for a 1000 hrs period.

### 6.2.4. Instrumentation

The extensive instrumentation for each reinforced wall included measurements of the following parameters throughout the construction and loading: 1) horizontal movement of the facing panels; 2) longitudinal displacements and strains in the geogrid reinforcement; 3) distribution of vertical earth pressure below the reinforced soil block; 4) vertical settlements at the top of the surcharge; and 5) temperature in the fill.

The horizontal movements of the central facing panels were monitored by an array of potentiometer type displacement transducers. The displacements of the reinforcement in the longitudinal direction were measured by attaching tensioned steel wires to selected locations on the central portion of the grids. Movement of the thin steel wires was recorded by displacement transducers located at the back of the test facility. The wires were protected from the granular fill by passing them through stiff plastic tubing. Strain gages were attached at selected mid-rib locations along the length of each central reinforcement strip. The gages used were high-strain, foil type gages

manufactured by Showa Measuring Instruments Co., Ltd. and had been found to perform satisfactorily in geogrid applications (Bathurst et al., 1987). Experience with strain-gaging Tensar geogrids indicates that in general, the overall (average) grid strain slightly differs from rib strains measured by strain gages due to grid geometry and variable material moduli. In isolation tests, conducted prior to the wall tests, however, indicated that the grid strains and rib strains were essentially equivalent up to about 1.8% grid strain (Bathurst et al., 1987).

## 6.3. Presentation of Test Results

A large amount of data were collected both during the construction and surcharge loading of the walls. Since the scope of this chapter is limited to a discussion of reinforcement loads under working stress conditions, only two types of measurements are presented here: i) horizontal panel movements, and ii) geogrid strains.

### 6.3.1. Test 1: Incremental Panel Facing Wall

Figure 6.4 summarizes the measurements of facing panel movements from Test No. 1. At the end of the surcharging load sequence, the top of the wall is displaced about 40 mm outwards from the initial alignment of the bottom panel. However, 14 mm of this movement were accumulated prior to placement of the top panel. The figure also shows that significant incremental movements were recorded at each surcharge load level. It is also evident from Figure 6.4 that there is a general mode of outward rotation occurring about the toe.

Bathurst et al. (1987) presented reinforcement strains corresponding to various stages of construction, surcharge loading, and dismantling operations. However, in order to limit the amount of data, strain measurements corresponding to only two events are presented here: a) 12 kPa surcharge sustained for 100 hrs (Event A) followed by, b) 50 kPa surcharge sustained for 1000 hrs (Event B). Event A corresponds to the design load condition for these walls while Event B is representative of the conditions prevailing at the end of the testing program. For the incremental panel wall, the test was terminated 500 hrs after the application of 50 kPa surcharge increment. However, the deformation rates at this time were found to be essentially zero and no difference was anticipated in the measurements at 500 and 1000 hrs (Bathurst and Koerner, 1987). Figure 6.5 shows the distribution of grid strains recorded along the length of each reinforcement layer. It can be seen that strain levels in the upper two layers (3 and 4, refer to Figure 6.2) of reinforcements are in the range  $\epsilon_a = 0.2 - 0.4\%$  following the construction of the wall and application of 12 kPa design surcharge (Figure 6.5a), while strains are generally smaller in levels 1 and 2. As additional surcharge is added (Figure 6.5b), there is a significant increase in the maximum axial strains,  $\epsilon_a \leq 1\%$ , which occur in layers 3 and 4 at about 450 mm behind the facing panels. There is no strong correlation between the magnitude of peak strain and reinforcement depth.

Figure 6.6 shows a comparison of grid strains calculated from the steel wire gages (extensometers) with grid strains obtained from rib mounted strain gages. The data show that the values from extensometers are never less than 70% of the average strain gage readings taken over the same length. The lower values were attributed to compliance of the extensometers, which were considered to give lower-bound estimates of the true grid strains (Bathurst et al., 1987).

### 6.3.2. Test 2: Propped Panel Facing Wall

Outward panel movements recorded at selected times during surcharging of the propped panel wall are shown in Figure 6.7. The data shows a progressive rotation of the full height wall facings about the toe. The total outward movement at the top of the wall is about 13 mm at the end of the test. Figure 6.8 shows the distribution of grid strains recorded at two selected times (events A and B) described previously.

Reinforcement strains following the construction and design loading (12 kPa, event A) of the propped panel walls (Figure 6.8a) are less than 0.1%. The magnitude of peak strain is very similar for all four layers irrespective of the depth. Upon subsequent application of 50 kPa surcharge (Figure 6.8b), there is a large increase in axial strains in all layers with peak strains in the range,  $\epsilon_a = 0.4 - 0.6\%$ . Much larger strains recorded in layer 4 close to the wall facing (Figure 6.8b), were attributed to differential settlement between the facing panels and sand backfill (Bathurst et al., 1987). This relative movement was reduced in the incremental wall test by the compressible foam rubber void filler placed between the facing units (refer Section 6.2.3).

## 6.4. Results of Prediction Exercise

Several researchers attempted to predict the performance of RMC test walls at various elapsed times after initial construction and after surcharge loads had been applied to the top of the walls. The participants were asked to predict a variety of parameters including: a) horizontal movements of panels, b) strain (or load) in the reinforcement, and c) vertical pressures along the base of the reinforced soil mass. A total of 10 predictors (working individually or in teams) submitted estimates of wall performance. Some predictors attempted only one of the walls while others only

provided a portion of the requested items. Since the primary subject of this discussion is working stress magnitudes of reinforcement load, only the predictions of reinforcement strains are presented here.

#### 6.4.1 Methods of Analysis

The original paper (Bathurst and Koerner, 1987) provides brief descriptions of the methods of analysis employed by the predictors. Predictor A<sup>1</sup> was the only one to use the finite element method to simulate wall construction. This analysis used a non-linear hyperbolic, total stress-strain model for the soil (Duncan and Chang, 1970), and treated the geogrid reinforcement as a linearly elastic material. All of the other predictors used modifications of limit equilibrium methods to predict working stress performance of the walls. These methods generally fall into two main categories: methods in the first category (sometimes known as 'tie back analysis') rely on the concept of equivalence in horizontal forces in the soil and in the reinforcement. This approach implicitly assumes that only local horizontal force equilibrium and local interaction between the soil and reinforcement need be considered. The horizontal stress at a given depth is generally computed by multiplying the vertical effective stress,  $\sigma'_v$ , (due to overburden and surcharge loading) by a coefficient of lateral earth pressure,  $K$ . Thus, the maximum reinforcement force,  $F_R$ , is given by:

$$F_R = S_v S_h C K \sigma'_v \quad (6.1)$$

---

<sup>1</sup> In order to ensure openness of discussion, predictions were not attributed to their authors and individual predictors were identified by letters A through J.

where,  $S_v$  and  $S_h$  are vertical and horizontal reinforcement spacing respectively,<sup>2</sup> and  $C$  is an empirical constant derived from observations of actual reinforced walls. Various hypotheses exist regarding the variation of  $K$  with depth in a reinforced wall; it is common to assume that  $K$  varies between the 'at-rest' state,  $K_o$ , and the active condition,  $K_a$ . Table 6.2 summarizes some of the assumptions made by predictors who used tie-back type analyses to predict reinforcement loads for the RMC walls.

The second approach is based on conventional slope stability analyses (which compute stabilizing versus destabilizing forces and/or moments along an assumed failure surface), modified to account for the presence of tensile reinforcements (Claybourn and Wu, 1992). Bathurst and Koerner (1987) do not provide any description of the analytical methods employed by predictors G through J, except an indication that the methods correspond to limiting equilibrium calculations. It is reasonable to assume that some of these predictors may have used slope stability type of calculations to predict the reinforcement loads.

#### 6.4.2 Comparison with Measured Strains

In general, the comparisons have been restricted to predictions corresponding to the 12 kPa surcharge applied for 100 hrs (Event A) and 50 kPa surcharge sustained for 1000 hrs (Event B). The predictions of geogrid strains at selected times for the incremental panel wall test are given in Figures 6.9 and 6.10. With the exception of Predictor A and Predictor G, all analytical methods predicted values of the strains which are significantly larger than the measured strains. Most of the methods not only failed to predict the correct magnitude of strain in the reinforcement but also predicted

---

<sup>2</sup> For continuous grid type reinforcements,  $S_h = 1$ .

an increase in the strain with depth which is not evident from the measured data.

Figures 6.11 and 6.12 compare the predictions of geogrid strains in propped facing panel wall at selected times. Again, with the exception of Predictor A, all participants significantly overestimated measured strains in the reinforcement. The discrepancies between measured values and predictions of reinforcement strains are largest for the propped panel wall surcharged with 12 kPa load for 100 hrs (Figure 6.11).

Results of the prediction exercise demonstrate that the finite element method is able to account for the differences in the propped and incremental panel construction techniques and provides very reasonable predictions of load-transfer in both the cases. However, it is difficult to interpret the underlying physical basis of the model from complex numerical analyses. Limit equilibrium methods, on the other hand, failed to compute the reinforcement strains even within one order of magnitude. Large scatter between methods using same basic calculation procedure makes it difficult to evaluate and apply these methods in practice. The success of the finite element method shows that for predicting reinforcement loads under working stress conditions, elastic equilibrium analyses are more appropriate than modified limit equilibrium approaches.

## 6.5. APSR Estimates of Reinforcement Strains

This section illustrates the application of the APSR experimental data for estimating the maximum tensile strains and forces in the geosynthetic reinforcements.

Table 6.1 compares the engineering properties of the backfill sand used in the RMC instrumented walls with those of Ticino sand used in the APSR experiments. The RMC sand has larger average grain size and hence, its range of formation densities

( $\gamma_{min}$ ,  $\gamma_{max}$ ) is higher than Ticino sand. The RMC tests have a higher placement density than the APSR experiments ( $\gamma = 17.6$  vs.  $15.9 \text{ kN/m}^3$ ) but this condition represents a lower relative density ( $D_r = 52$  vs.  $75\%$ ). The effective friction angle,  $\phi'_{ps} = 50^\circ$ , in plane strain failure mode is slightly lower compared to  $\phi'_{ps} = 53^\circ$  for Ticino sand. However, the estimated value of the average secant shear modulus,  $G_{50}$  (at 50% of the shear strength) for the RMC sand is about 40% lower. This is probably due to the lower relative density of the RMC sand. The Tensor SR2 geogrid reinforcement used in the RMC walls has a small strain ( $\epsilon_a < 1\%$ ) response which is approximately linear with average axial stiffness,  $J = 600 \text{ kN/m}$ . The material with the closest value of axial stiffness used in the APSR cell is Nylon 6/6 ( $J = 980 \text{ kN/m}$ ). APSR tests on Nylon 6/6 are, therefore, used to predict load-transfer in the RMC trial walls. The following findings of the APSR and shear-lag research further support such a comparison:

1. The maximum tensile stress in a long inclusion is primarily controlled by relative stiffness of the inclusion and soil,  $E_f/G_m$ . Although the axial stiffness of the reinforcement used in the RMC trial walls is about 40% lower than that of the Nylon 6/6 inclusion, the ratio,  $E_f/G_m$  is comparable in both cases.
2. For a reinforcing material with axial stiffness per unit width,  $J < 1000 \text{ kN/m}$ , shear-lag analyses (Abramanto and Whittle, 1993) show that the maximum reinforcement load, corresponding to an infinitely long inclusion, is achieved in an inclusion with half-length,  $L/2 < 400 \text{ mm}$ . Therefore, the APSR load-transfer measurements for the longest Nylon 6/6 inclusion ( $L/2 = 360 \text{ mm}$ ) are representative of loads in a long, prototype-scale reinforcement. The RMC grids have length,  $L = 3 \text{ m}$  and hence, will mobilize similar levels of tensile stress under the same loading conditions.



3. The presence of lateral members (ribs) in a grid causes approximately 20% reduction in the magnitude of loads (for a given applied stress ratio,  $R$ ) in an infinitely long planar inclusion with equivalent stiffness (refer to Section 5.3). Therefore, the measurements for Nylon 6/6 inclusion are expected to provide an upper bound estimate of the tensile loads for the geogrid reinforcement used in the RMC walls.
4. For the APSR cell, the lateral dimension of the soil matrix,  $m = 0.57m$ , compared to an average reinforcement spacing  $m = 0.75m$  for the RMC walls. Shear-lag calculations, however, show that for the range of material properties ( $E_f/G_m, \nu_m, \nu_f$ ) representative of the RMC walls, this difference in reinforcement spacing has no significant effect on the predicted maximum inclusion loads.

Figure 6.13 shows the direct application of the APSR cell geometry to the RMC model wall tests. Measurements of load-transfer in the APSR cell can be equated with the stress conditions locally acting around the distant end of a geogrid reinforcing layer bounded by two planes A-A' and B-B' (Figure 6.13b). The plane A-A' is located within the unreinforced soil mass; while B-B' is located at a distance  $X_1$  ( $X_1 > 360$  mm; from above) which is sufficient to mobilize the maximum tensile stress in the reinforcement. The analyses assume that gradients due to self weight stresses can be ignored in preliminary calculations of load-transfer hence the stresses on the free body are given as the average vertical stress  $\sigma_v$  and horizontal stress  $\sigma_h$ . The value of  $\sigma_h$  is linked to  $\sigma_v$  through a coefficient of earth pressure  $K$ , while horizontal equilibrium in the element is maintained by the tensile force,  $F_R$ , and reaction stress in the soil matrix,  $K_B \sigma_v$  (Figure 6.13b).

In the APSR experiments, the plane B-B' represents the 'rigid' rear wall of the cell,  $\sigma_h$  is maintained at a constant value and the specimen is sheared by increasing  $\sigma_v$  (and hence  $R = \sigma_v/\sigma_h$  increases during shearing). In the RMC model wall tests, deformations of the facing panels (C-C', Figure 13a) are constrained during construction but are free to deform when surcharge loads are applied at the ground surface. Deformations along planes B-B' and A-A' are inevitable, but are not measured directly in the tests. If the wall face is rigidly braced then there are no nominal lateral deformations within the sand and  $\sigma_h = K_o \sigma_v$  (i.e.,  $R = 1/K_o$  throughout loading). However, in reality lateral deformations do occur as surcharge loads are applied, such that  $K_a \sigma_v < \sigma_h < K_o \sigma_v$ , where  $K_a$  is the 'active' earth pressure coefficient, and corresponds to the minimum pressure the soil can exert along a vertical plane.

There have been extensive measurements of lateral earth pressure acting on retaining walls with dry granular backfill (e.g., Tschebotarioff, 1943; Terzaghi, 1936). Handy (1985) has reported an extensive study of the development of lateral earth pressures from various controlled modes of wall deformations. One particular case of interest involves rigid rotation of the wall about a fixed base. For this case, the experimental data show that the lateral earth pressure coefficient,  $K$ , is approximately constant over the full height of the wall. The limiting earth pressure corresponding to  $K = K_a$  is mobilized at rotation angles,  $\delta_H/H = 0.1$  to  $0.5\%$ , where  $K_a = (1 - \sin \phi')/(1 + \sin \phi')$  is the Rankine earth pressure coefficient, and  $\phi'$  is the peak friction angle of the granular backfill.

The measured lateral deformations for both the incremental and propped RMC model walls (Figures 6.4 and 6.7), can be represented approximately as a rigid body rotations. For incremental construction there is also a net translation of the wall as shown in Figure 6.4. From the above discussion, it is clear that the most difficult

aspect of relating APSR data directly to the prototype reinforced soil wall lies in the estimation of  $K$ . This thesis proposes the following simplifying assumptions:

1. Deformations of the plane A-A' can be equated with those measured at the front face of the wall, and can be approximated by a rigid body rotation about the base.
2. The results given by Handy (1985) apply directly to the RMC geogrid reinforced walls. Hence, i)  $K$  is constant over the depth of the soil; and ii)  $K$  reduces from an initial  $K_0$  value to active conditions, occurring at  $\delta_H/H = 0.1 - 0.5\%$ .
3. The initial  $K_0$  stress in the soil is affected by the confining pressure, compaction procedures, etc. For most dry granular soils,  $K_0 = 0.4 \pm 0.1$  (Ladd et al., 1977)<sup>3</sup> hence,  $R = 2.0$  to  $3.3$  is the stress ratio operative when there is no lateral strain in the reinforced soil mass.
4. The active earth pressure coefficient is computed from Rankine assumption (i.e., with no shear stress acting on vertical planes),  $K_a = (1 - \sin \phi') / (1 + \sin \phi')$ , where  $\phi'$  is the peak friction angle which is a function of confining pressure in the sand. Figure 6.14 presents the non-linear strength envelope for the RMC sand based on the secant friction angles quoted by Bathurst (1989) and the failure criterion developed by Baligh (1975). The failure envelope (based on reported data) is highly non-linear at low confining pressures. For both  $\Delta q_s = 12$  kPa and  $50$  kPa, the maximum vertical

---

<sup>3</sup> Widely used empirical expressions such as Jaky (1944):  $K_0 = 1 - \sin \phi'$ , produce unrealistically low values of  $K_0 = 0.23$  for  $\phi'_{ps} = 50^\circ$  measured for the RMC sand. Similarly, for high compaction pressures,  $K_0 = 1$  is theoretically possible (Seed and Duncan, 1987).

stress,  $\sigma_v = 50 - 100$  kPa depending on the depth of the reinforcement layer. These results imply that the peak friction angle,  $\phi' = 53^\circ - 45^\circ$  (decreasing with depth) and hence,  $K_a = 0.17 - 0.11$ , and  $R = 5.9 - 9.0$ .

For the propped wall (Test 2) there are no wall rotations during construction, hence,  $R = 2.0 - 3.3$  and  $F_R \approx 0$ , (Figure 6.15a). For a surcharge load,  $\Delta q_s = 12$  kPa (Event A), the average rotation  $\delta_H/H$  equals 0.03%, which increases to  $\delta_H/H = 0.4\%$  at  $\Delta q_s = 50$  kPa (Event B). In contrast, for the incremental test, initial construction causes,  $\delta_H/H = 0.5\%$ , and further rotation occurs during surcharge loading ( $\delta_H/H = 0.7 - 1\%$  for  $\Delta q_s = 12$  kPa and 50 kPa, respectively). Figure 6.15a shows the expected maximum tensile force and strain in the geogrid based on the APSR data (for Nylon 6/6 with  $L/2 = 360$  mm), as a function of the principal stress ratio,  $R = \sigma_1/\sigma_3$ . From these results, it is clear that: a) there is negligible tensile load/strain in the reinforcement during construction of the propped wall; and b) assuming that the wall rotation at Event B ( $\Delta q_s = 50$  kPa) is sufficient to mobilize active earth pressures at all depths in the backfill, the maximum tensile strains expected in the reinforcements,  $\epsilon_f = 0.48$  to 1.3% corresponding to  $R = 5.9$  to 9.0. However, uncertainties in the value of  $R$  as a function of wall rotation (change in  $\delta_H/H$ ) preclude more definite conclusions by this approach.

In the APSR experiments, tensile loads develop in the reinforcement due to (tensile) lateral strains,  $\epsilon_{xx}$ , within the surrounding soil matrix. For Nylon 6/6 inclusions, the magnitudes of the reinforcements loads are relatively small ( $F_R < 10$  kN/m) and hence, the inclusions have minimal effects on the externally measured stress-strain properties of the soil matrix (Figure 6.16). As a result, it is possible to relate the maximum tensile strain in the reinforcement directly to the lateral strain in the soil matrix as shown in Figure 6.15b. The APSR data show that the maximum tensile

force/strain is almost a linear function of the lateral strain for  $\epsilon_{xx} < 1\%$  and hence, provide a much more consistent means for interpreting the performance of the RMC model walls.

The first step is to compute the maximum tensile lateral strain in the backfill soil at each reinforcement layer based on the observed movements of the facing panel. Figure 6.17a shows a typical soil layer with total length,  $L = 6$  m and height,  $H = 0.75$  m. Since there is a net rotation of the wall facing about toe, in addition to tensile strain,  $\epsilon_{xx}$ , each layer is also subjected to shear strain,  $\tau_{xy}$ , due to the difference in the outward movement at the top and bottom of the soil layer. If the differential between the facing displacement of the top and bottom of the soil layer is  $\delta_x$ , then horizontal displacements within the soil can be expressed as:

$$u_1 = \delta_x \left( \frac{X_1}{H} \right) \left( \frac{X_2}{L} \right) \quad (6.2)$$

where  $X_1$  and  $X_2$  are coordinates in horizontal and vertical directions respectively (Figure 6.17a). Differentiating Equation 6.2 with respect to  $X_1$  and substituting  $X_2/H = 0.5$ , the average lateral strain at mid-height is obtained as:

$$\bar{\epsilon}_{xx} = \frac{1}{2} \frac{\delta_x}{L} \quad (6.3)$$

Similarly, the average shear strain at the center of the layer:

$$\bar{\tau}_{xy} = \frac{1}{2} \left( \frac{\partial u_1}{\partial X_2} \right) = \frac{1}{2} \left( \frac{\delta_x}{H} \frac{X_1}{L} \right) = \frac{1}{4} \frac{\delta_x}{H} \quad (6.4)$$

Substituting  $L = 6$  m and  $H = 0.75$  m in Equations 6.3 and 6.4, shows that:

$$\bar{\epsilon}_{xx} = \frac{\delta_x}{12}; \quad \bar{\tau}_{xy} = \frac{\delta_x}{3} \quad (6.5)$$

Equation 6.5 shows that on an average, the shear strain in the soil layer is four times the horizontal tensile strain. Figure 6.17 shows the Mohr circle of strain assuming  $\epsilon_{xx} = 4 \tau_{xy}$  and vertical strain,  $\epsilon_{yy} = 0$ . The diagram shows that the maximum tensile strain,  $\epsilon_{max} = 0.377 \delta_x$ . From the above discussion, it is clear that shear deformations substantially increase the maximum tensile strain in a relatively thin, reinforced soil layer. If the total outward displacements of the wall facing at the bottom and top of a layer are  $\delta_{xb}$  and  $\delta_{xt}$  respectively, then the maximum tensile strain in soil is computed as:

$$\epsilon_{max} = \frac{\delta_{xb}}{6} + 0.377 (\delta_{xt} - \delta_{xb}) \quad (6.6)$$

The reinforcement strain is obtained from tensile strain in the soil matrix,  $\epsilon_{max}$  using the relationship shown in Figure 6.16b. A linear regression through the initial portion of the plot in Figure 6.15b ( $\epsilon_{xx} \leq 1\%$ ) shows that:

$$\epsilon_r = 0.6875 \epsilon_{max} \quad (6.7)$$

Tables 6.3 and 6.4 summarize the computed values of maximum tensile strains in the soil and 'predictions' of geogrid strains for the incremental and propped panel wall experiments.

Tables 6.5 summarizes the geogrid strains measured at selected locations for the propped panel wall and incremental panel tests respectively. In general, the maximum strains within a particular layer were recorded at either 445 mm or 625 mm behind the facing panels. In some cases, the highest strains were recorded by the strain gages located 220 mm behind the facing panels. However, this was ignored in the present comparison because there is evidence that geogrid strains in the vicinity of facing panels were influenced by relative vertical compressibility of the facing units (see Section 6.3.2). The APSR predictions summarized in Tables 6.3 and 6.4 compare favorably

with the data presented in Table 6.5, although the strains measured in upper three layers (Layer 2 to 4) do not decrease with the depth as would be expected from the computations presented above.

## 6.6. Discussion and Conclusions

The chapter presented a well documented case study involving two 3 m high reinforced soil model walls constructed with a sand backfill and Tensar SR2 geogrid reinforcement. The first model used the conventional articulated incremental panel wall construction and the second full height wall panels and external bracing (propped) system which was removed only after construction. Each wall was subjected to sustained surcharge loading up to 50 kPa following construction.

The APSR measurements of load-transfer for Nylon 6/6 inclusion with half-length,  $L/2 = 360$  mm were used to establish a correlation between the maximum lateral strain in the backfill soil and expected tensile strain in the reinforcement. Strains in the soil mass were computed from the observations of horizontal movements of the facing panels. In general, the APSR predictions of maximum reinforcement strains are of the same order of magnitude as the measured data.

The APSR estimates of reinforcement strains are very reasonable when compared to results the prediction exercise (Section 6.4). With the exception of Predictor A, the methodologies employed were essentially limiting equilibrium analyses. Limit equilibrium methods are known for over-predicting reinforcement loads under working stress conditions. These methods may be sensitive of choice of friction angle, and other factors such as the side friction and base restraint that are

relevant to trial walls. The discrepancy between the predictions and measured data for the RMC walls, however, is simply too large to be accounted for by these factors.

The assumption of local force equilibrium between the reinforcement stresses and the horizontal soil stresses (implicit in Equation 6.1) does not adequately describe equilibrium conditions in a reinforced soil mass. This can be illustrated by comparing reinforcement strains in the propped panel wall and incremental panel wall. Any working stress design method that relies on the concept of local horizontal equilibrium, will fail to predict the observed relative magnitudes of reinforcement strains in the RMC trial walls. The conventional strategy of assuming the  $K_0$  condition when very little wall yielding has occurred (as in the propped panel wall) would predict higher reinforcement loads and hence, strains for the propped panel wall than the incremental panel wall. The APSR cell, along with the shear-lag analysis can be used to develop models for interpretation and prediction of load-transfer in reinforced soil structures. Such models will have to take into the account influence of factors such as compaction, base restraint, and connection between reinforcement and face panel.



Property	Symbol	Ticino Sand	RMC Sand
Mean Particle Diameter	$D_{50}$	0.5 mm	1.2 mm
Maximum and Minimum Densities	$\gamma_{max}$ $\gamma_{min}$	16.7 kN/m <sup>3</sup> 13.5 kN/m <sup>3</sup>	19.2 kN/m <sup>3</sup> 15.9 kN/m <sup>3</sup>
Placement Dry Density	$\gamma_d$	15.9 kN/m <sup>3</sup>	17.6 kN/m <sup>3</sup>
Relative Density	$D_r$	75%	52%
Peak Friction Angle from Consolidated Drained Triaxial Tests ( $\sigma_3 \approx 30$ kPa)	$\phi'_{tx}$	45° *	41° **
Failure Strain in Triaxial Test ( $\sigma_3 \approx 30$ kPa)	$\epsilon_f$	3.0 - 3.5%	4.0%
Peak Friction Angle in Plane Strain Failure Mode	$\phi'_{ps}$	53°	50° †
Secant Shear Modulus at 50% of Peak Shear Strength	$G_{50}$	6000 kPa	3700 kPa ‡

\* Tests performed at relative density  $D_r = 93\%$ .

\*\* Test performed at relative density  $D_r \approx 27\%$ .

† Obtained from Direct Shear Tests. Interpolated for the average value of 'insitu' confining stress.

‡ Rough estimate from large box shear tests (Jewell, 1987).

Table 6.1: Comparison of Ticino Sand with the Sand Used in RMC Trial Walls.

Prediction Method	Coefficient, C	Distribution of Coefficient of Lateral Earth Pressure, K	
		Incremental Panel Wall	Propped Panel Wall
B	1	$K_o : 0 \leq Z \leq 2H/3$ $K_a : 2H/3 \leq Z \leq H$	No distinction
D	0.98	$K_a : 0 \leq z \leq H$	$0.5 (K_o + K_a)$
E	1	$K_o : 0 \leq Z \leq 3H/4$ $K_o/2 : H/4 \leq Z \leq H$	No distinction
F	1	$K_a : 0 \leq z \leq H$	No distinction

Table 6.2: Comparison of Some of the Design Methods Used by the RMC Prediction Exercise Participants.

Layer Number	Elevation above Base (mm)	Facing Panel Movement (mm)		Strain in Soil Matrix (%)	Geogrid Strain (%)
		Top	Bottom		
1	250	0.25	0	0.0094	0.0065
2	1000	0.5	0.25	0.014	0.01
3	1750	2.5	0.5	0.084	0.058
4	2500	4.4	2.5	0.113	0.078

(a) 12 kPa Surcharge for 100 Hours.

Layer Number	Elevation above Base (mm)	Facing Panel Movement (mm)		Strain in Soil Matrix (%)	Geogrid Strain (%)
		Top	Bottom		
1	250	3.5	1	0.11	0.076
2	1000	6.0	3.5	0.153	0.105
3	1750	9.5	6.0	0.232	0.159
4	2500	13.0	9.5	0.290	0.20

(b) 50 kPa Surcharge for 1000 Hours.

Table 6.3: Summary of Predicted Reinforcement Strains for Propped Panel Wall.

Layer Number	Elevation above Base (mm)	Facing Panel Movement (mm)		Strain in Soil Matrix (%)	Geogrid Strain (%)
		Top	Bottom		
1	250	11.0	0	0.415	0.285
2	1000	12.0	7.7	0.290	0.199
3	1750	14.0	11.0	0.296	0.204
4	2500	23.0	15.0	0.552	0.380

(a) 12 kPa Surcharge for 100 Hours.

Layer Number	Elevation above Base (mm)	Facing Panel Movement (mm)		Strain in Soil Matrix (%)	Geogrid Strain (%)
		Top	Bottom		
1	250	15.0	0	0.566	0.389
2	1000	19.0	13.0	0.443	0.305
3	1750	21.0	20.0	0.371	0.255
4	2500	38.0	21.0	0.991	0.626

(b) 50 kPa Surcharge for 500 Hours.

Table 6.4: Summary of Predicted Reinforcement Strains for Incremental Panel Wall.

Layer Elevation above Base (mm)	12 kPa Surcharge for 100 hrs.		50 kPa Surcharge for 1000 hrs.	
	Strain (%) Distance from Wall Face		Strain (%) Distance from Wall Face	
	445 mm	625 mm	445 mm	625 mm
250	0.04	0.01	0.36	0.06
1000	0.06	0.04	0.35	0.36
1750	0.05	0.07	0.35	0.67
2500	0.05	0.05	0.55	0.40

(a) Propped Panel Wall Test.

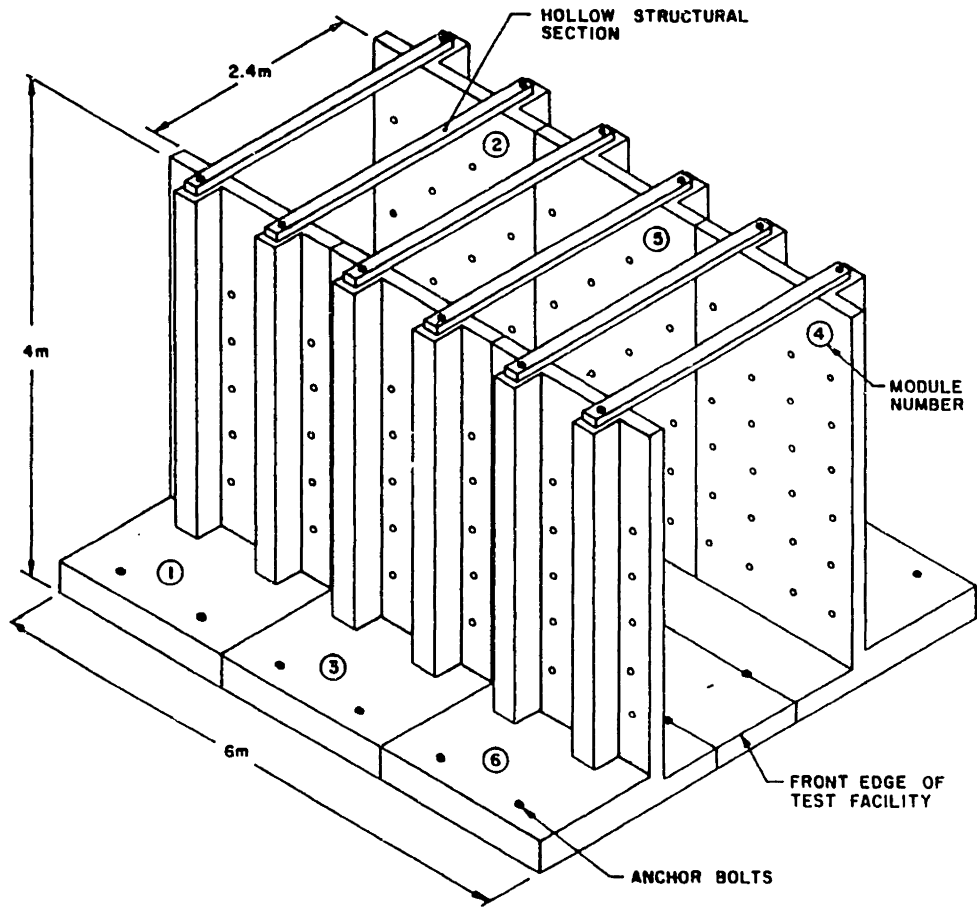
Layer Elevation above Base (mm)	12 kPa Surcharge for 100 hrs.		50 kPa Surcharge for 500 hrs.	
	Strain (%) Distance from Wall Face		Strain (%) Distance from Wall Face	
	445 mm	625 mm	445 mm	625 mm
250	0.07	0.03	0.07	0.01
1000	0.22	0.16	-	-
1750	0.37	0.29	0.79	0.70
2500	0.22	0.18	0.89	0.58

Note: '-' Denotes gage failure.

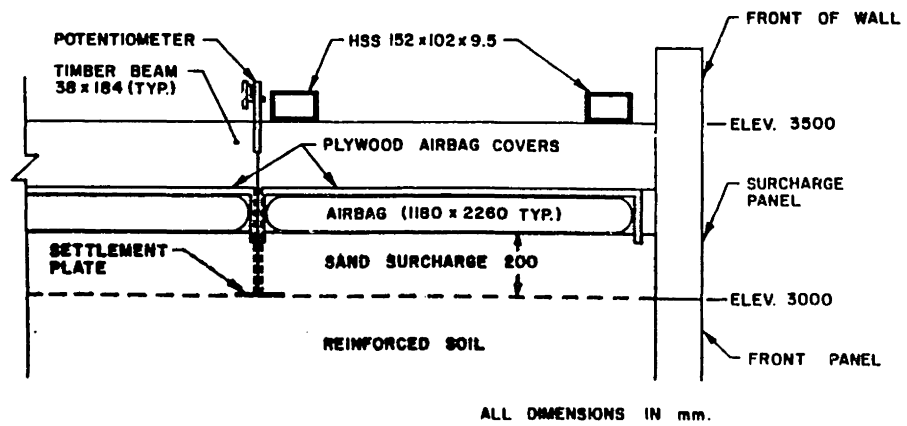
(b) Incremental Panel Wall Test.

Table 6.5: Summary of Measured Reinforcement Strains





(a) Structural Box Built from Concrete Wall Segments.



ALL DIMENSIONS IN mm.

(b) Airbag Surcharging System.

Figure 6.1: Principal Components of RMC Retaining Wall Test Facility (after Bathurst et al., 1987).

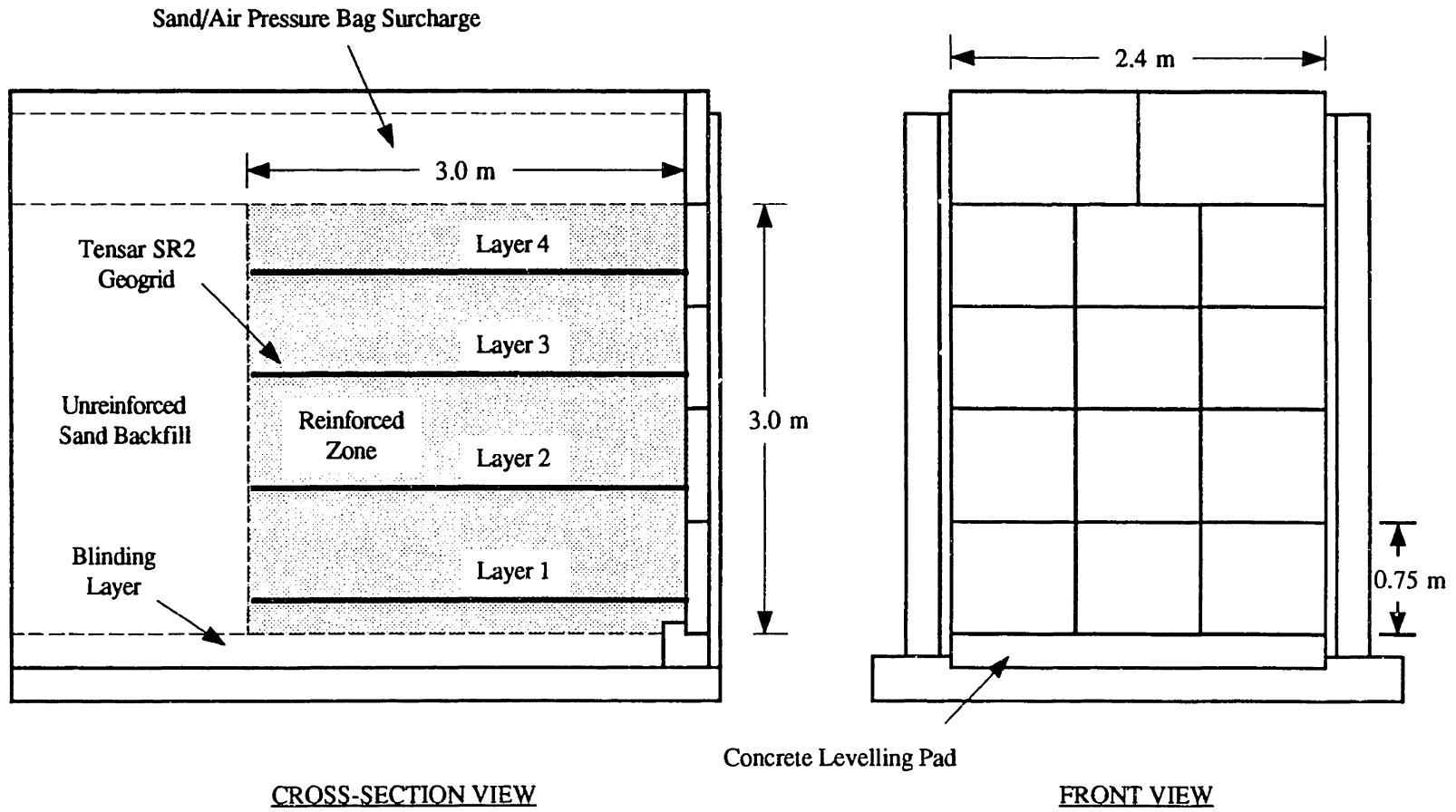


Figure 6.2: General Arrangement for Incremental Panel Wall Construction.



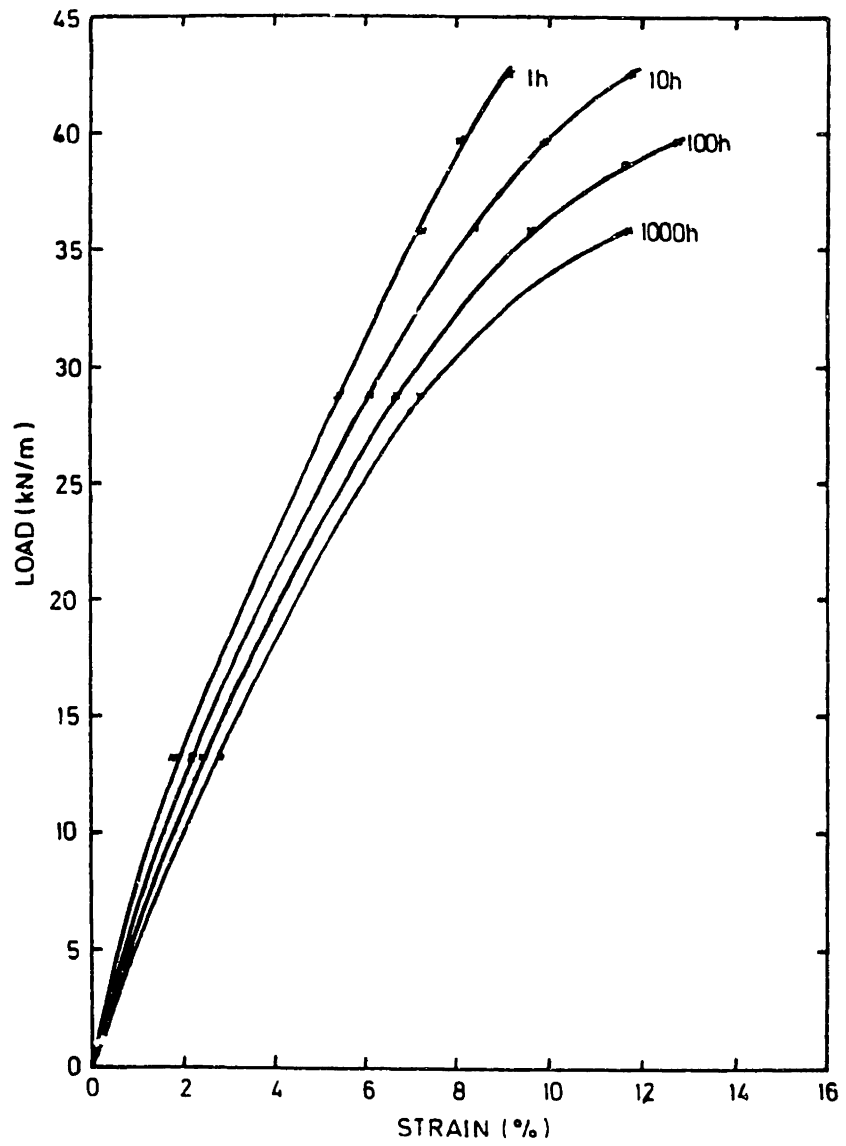


Figure 6.3: Isochronous Load-Strain Curves for Tensar SR2 Geogrids at 20° C (after Bathurst et al., 1987).

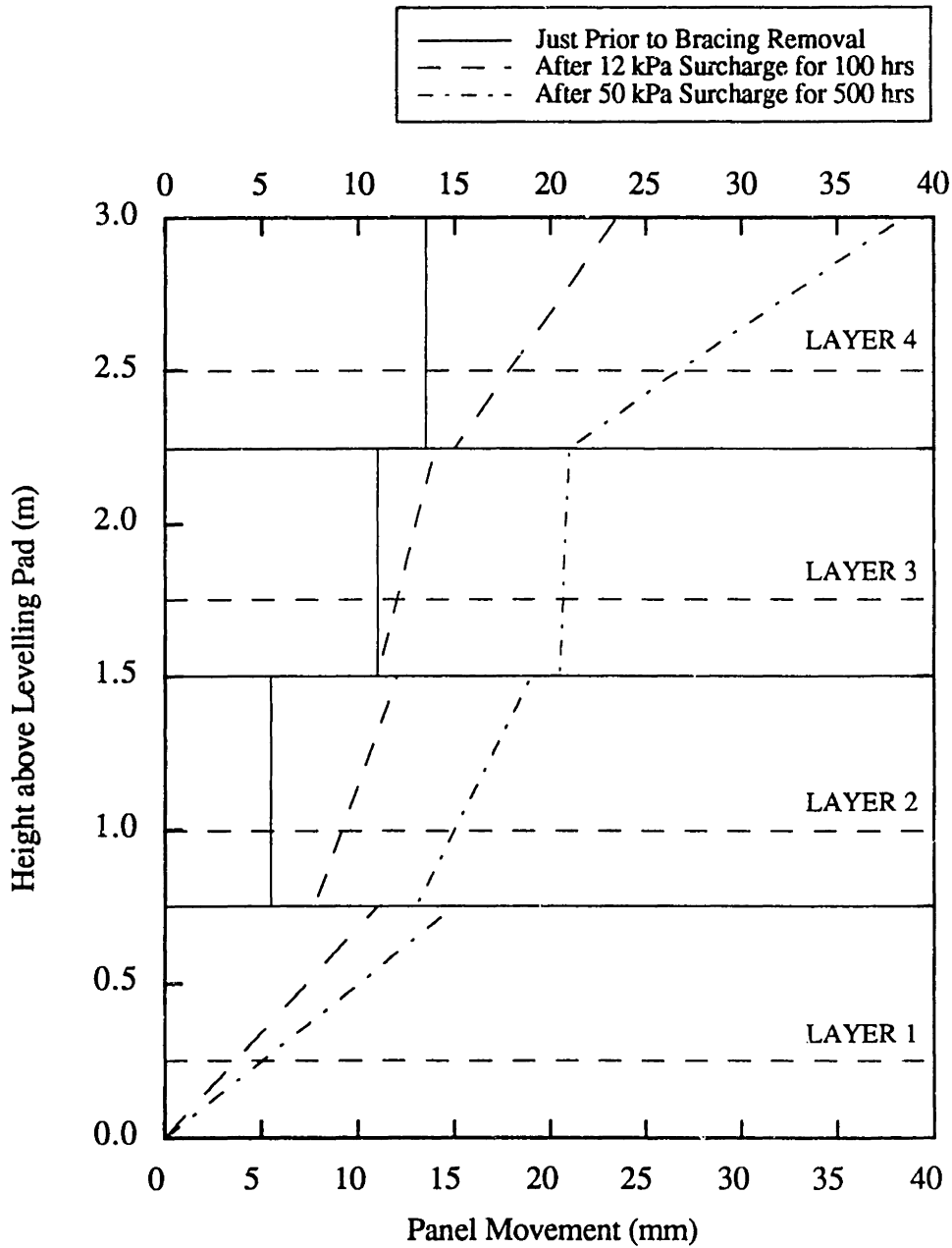
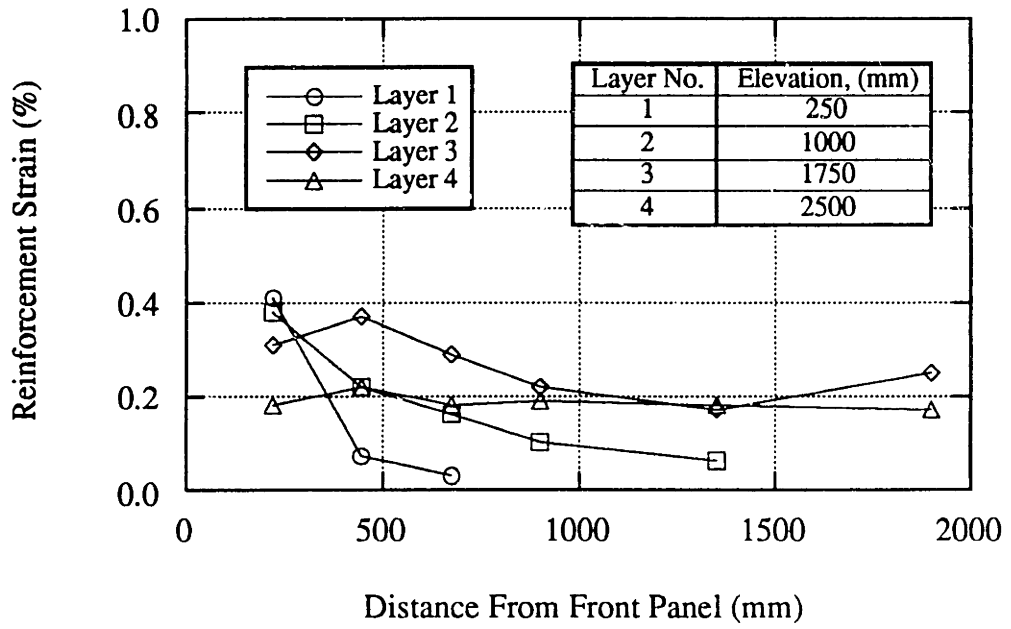
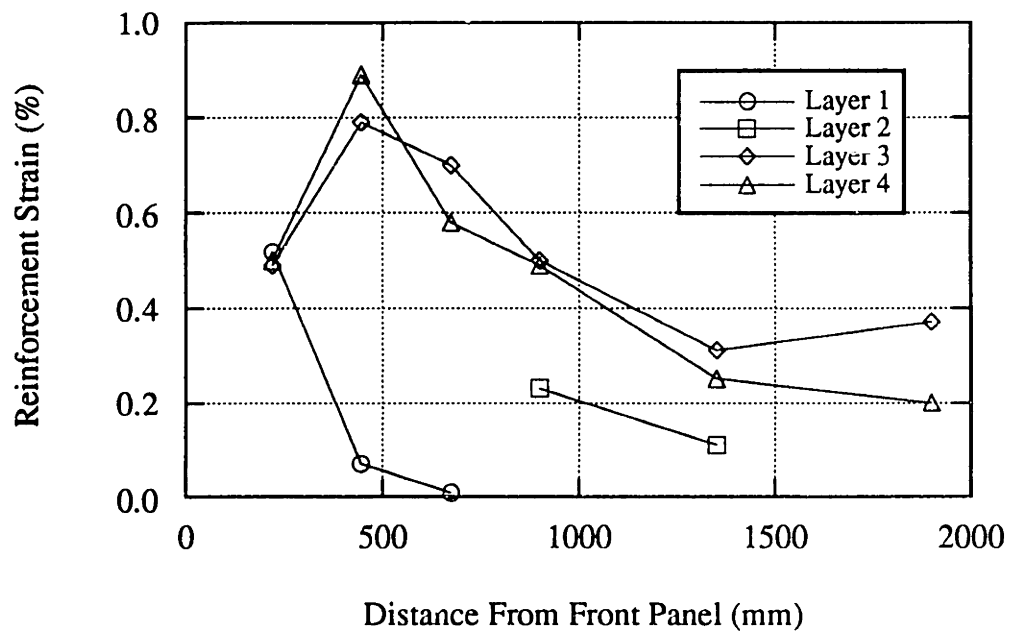


Figure 6.4: Summary of Panel Positions for Incremental Panel Wall Test (after Bathurst et al., 1987).



(a) 12 kPa Surcharge Sustained for 100 Hours.



(b) 50 kPa Surcharge Sustained for 500 Hours.

Figure 6.5: Distribution of Geogrid Strains for Incremental Panel Wall Test.

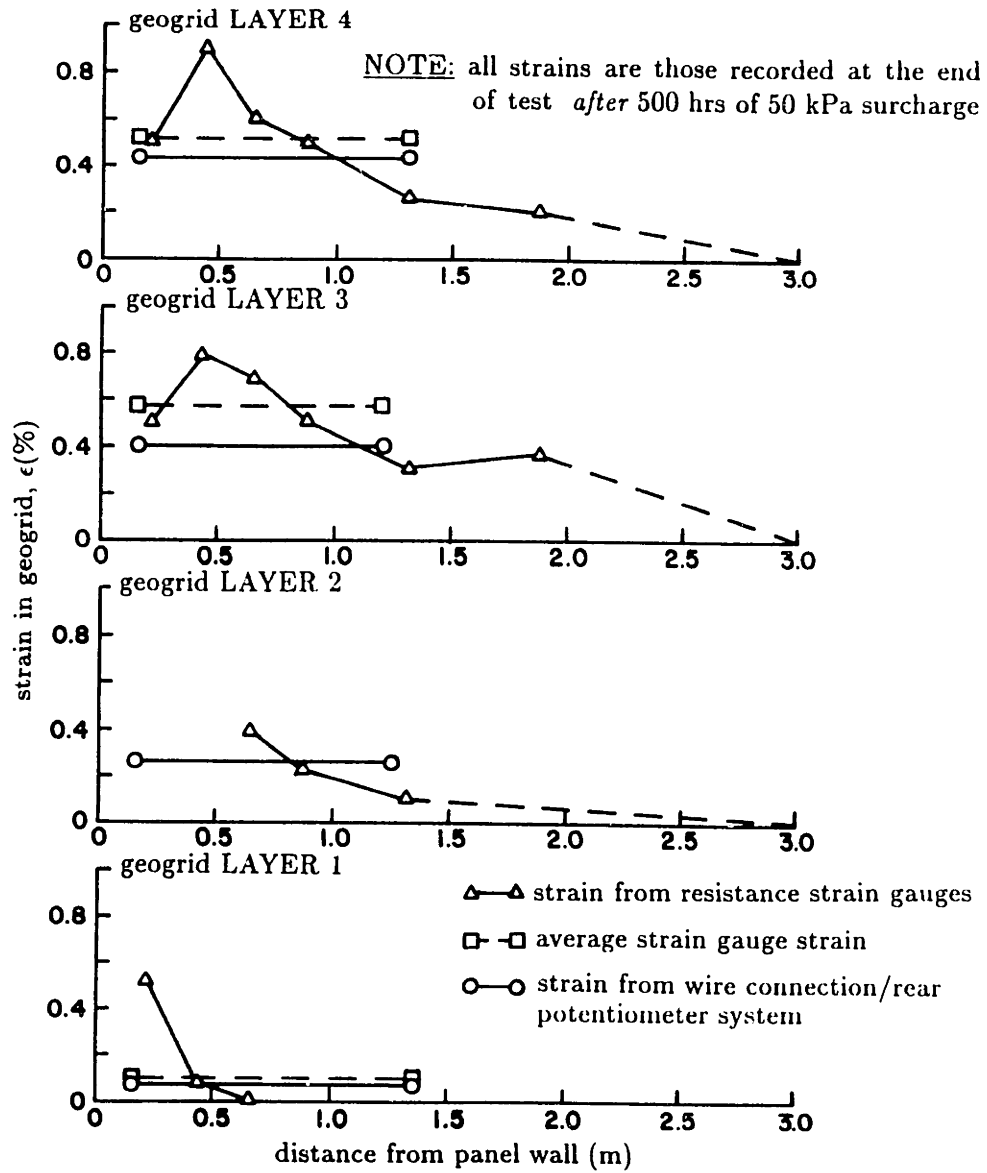


Figure 6.6: Comparison of Grid Strains from Strain Gage Readings and Extensometer Data for Incremental Panel Wall Test (after Bathurst et al., 1987).

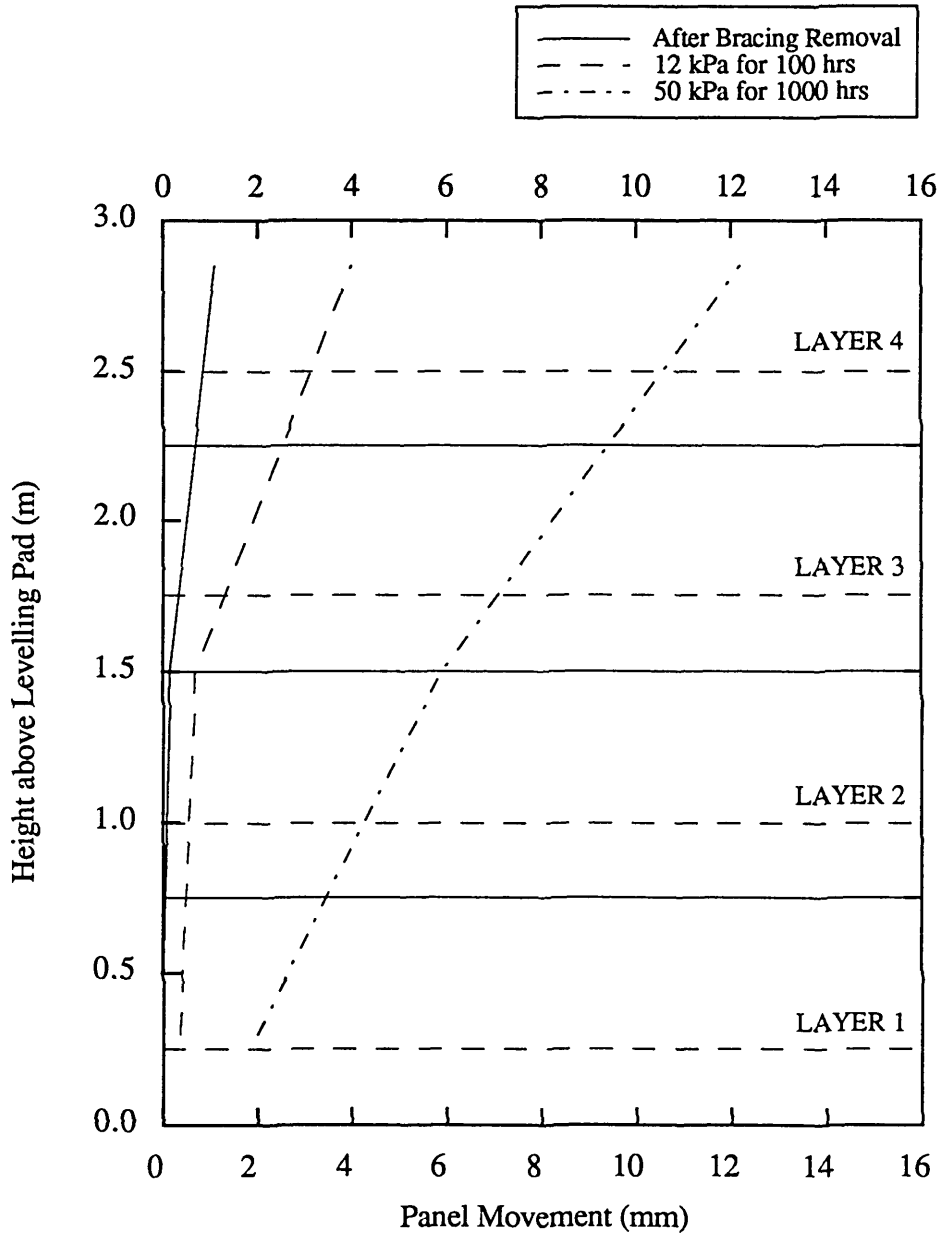
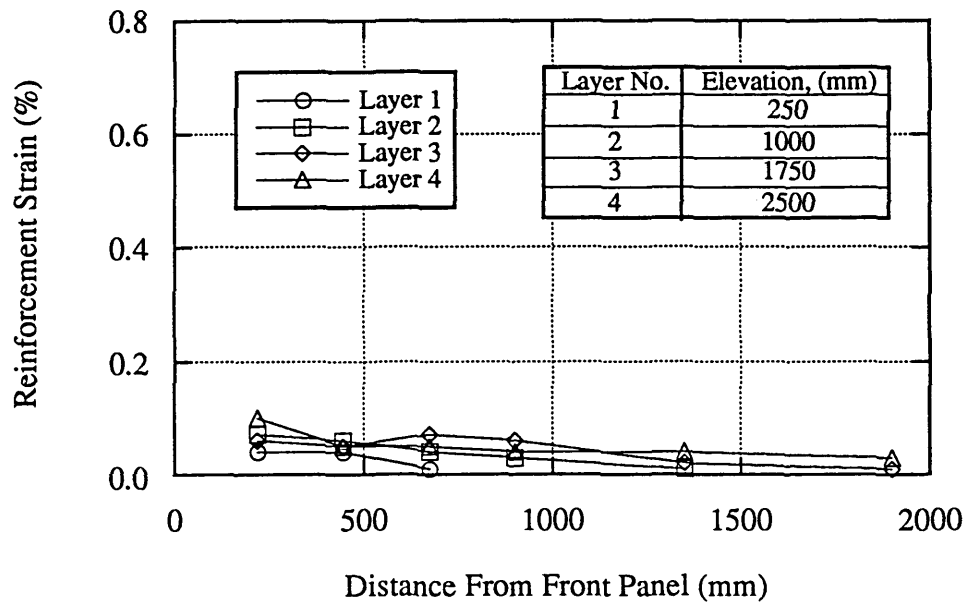
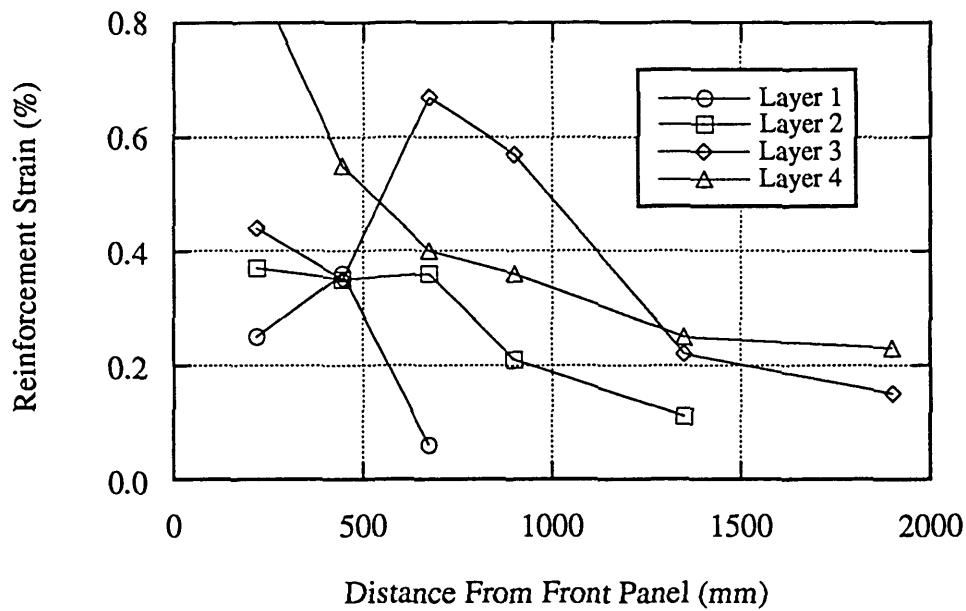


Figure 6.7: Summary of Panel Movements for Propped Panel Wall Test (after Bathurst et al., 1987).



(a) 12 kPa Surcharge Sustained for 100 Hours.



(b) 50 kPa Surcharge Sustained for 1000 Hours.

Figure 6.8: Summary of Strain Gage Readings for Propped Panel Wall Test.

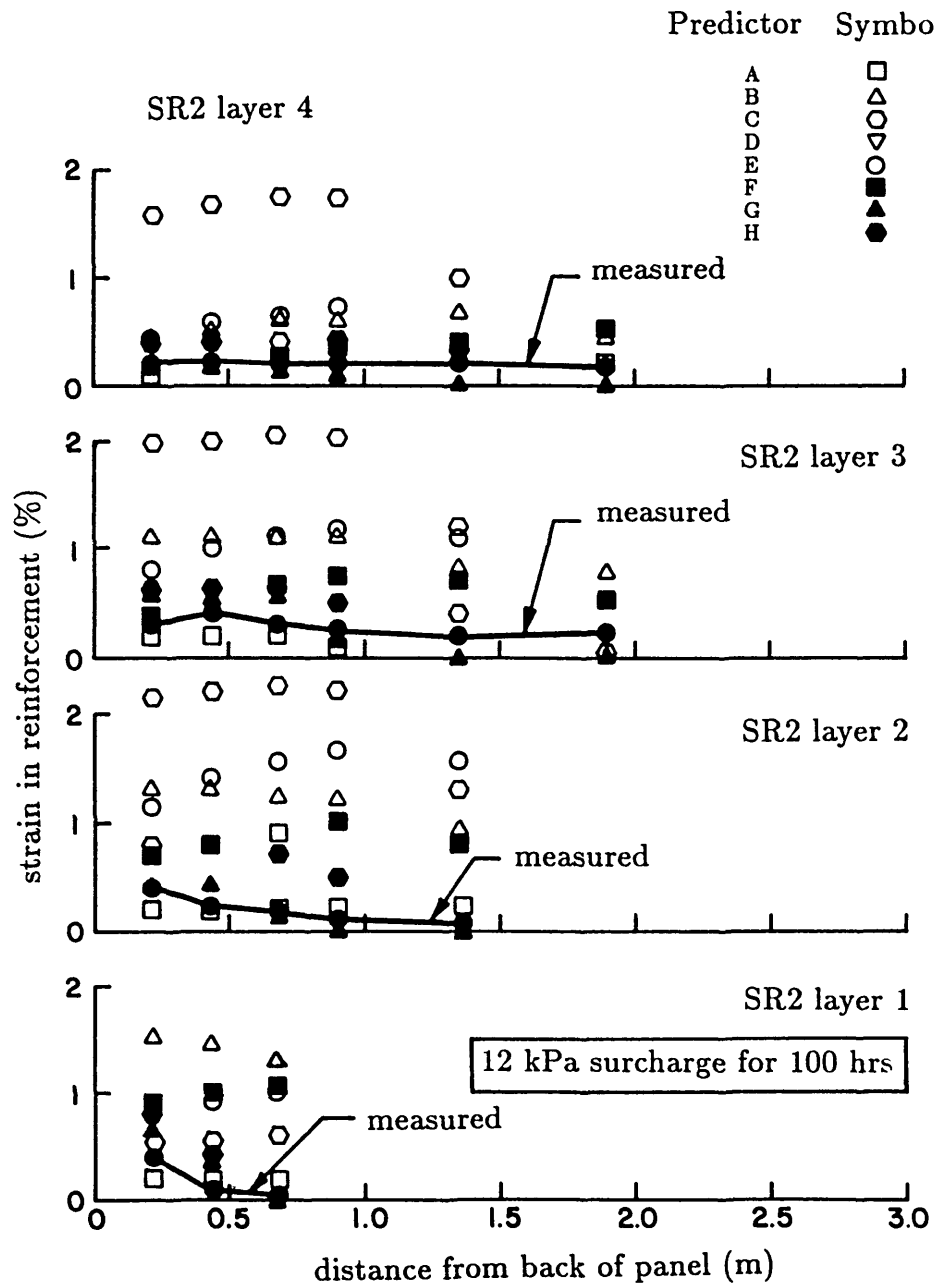


Figure 6.9: Reinforcement Strains in Incremental Panel Wall after 12 kPa Surcharge for 100 hours (after Bathurst and Koerner, 1987).

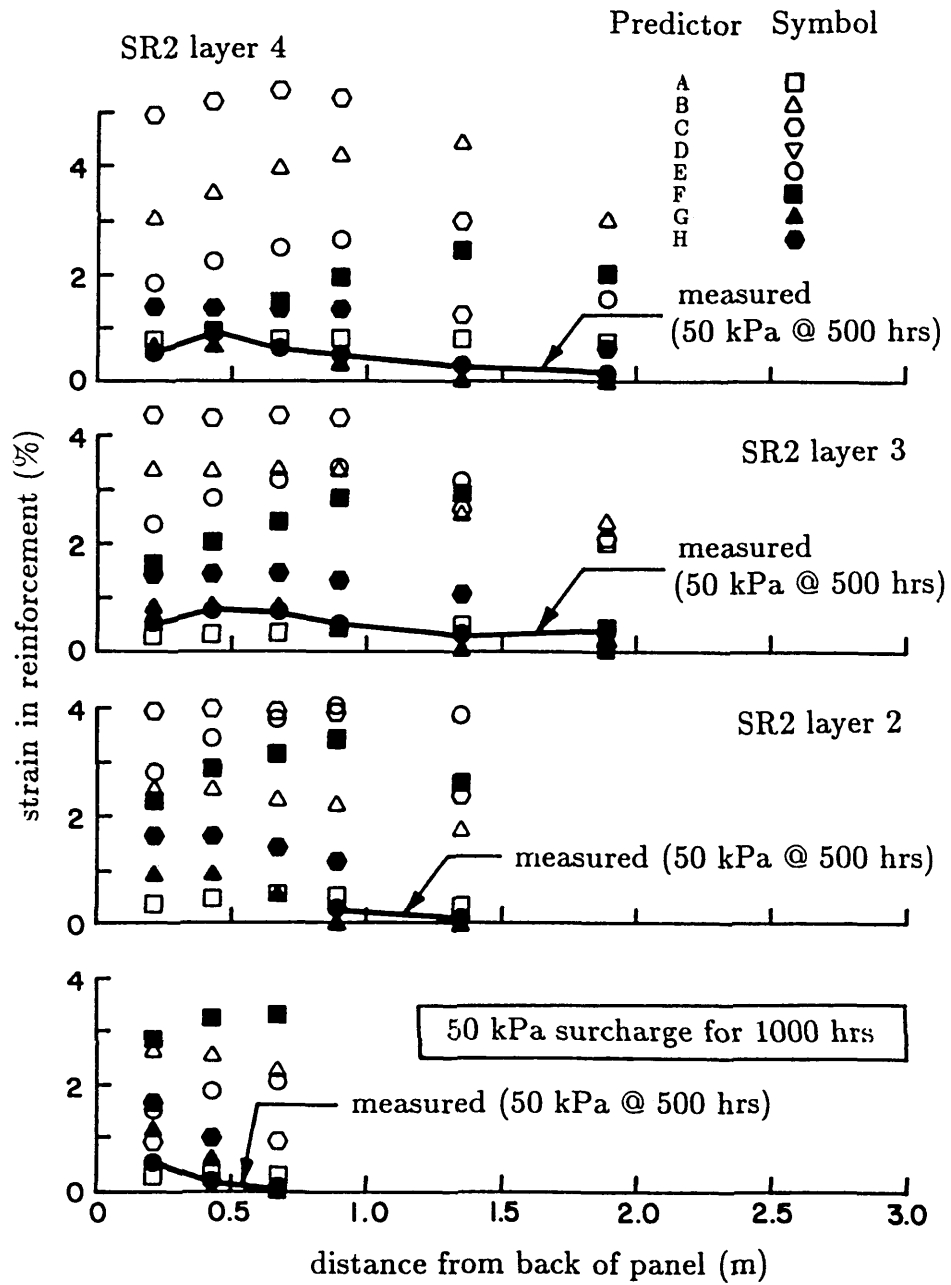


Figure 6.10: Reinforcement Strains in Incremental Panel Wall after 50 kPa Surcharge for 1000 hours (after Bathurst and Koerner, 1987).



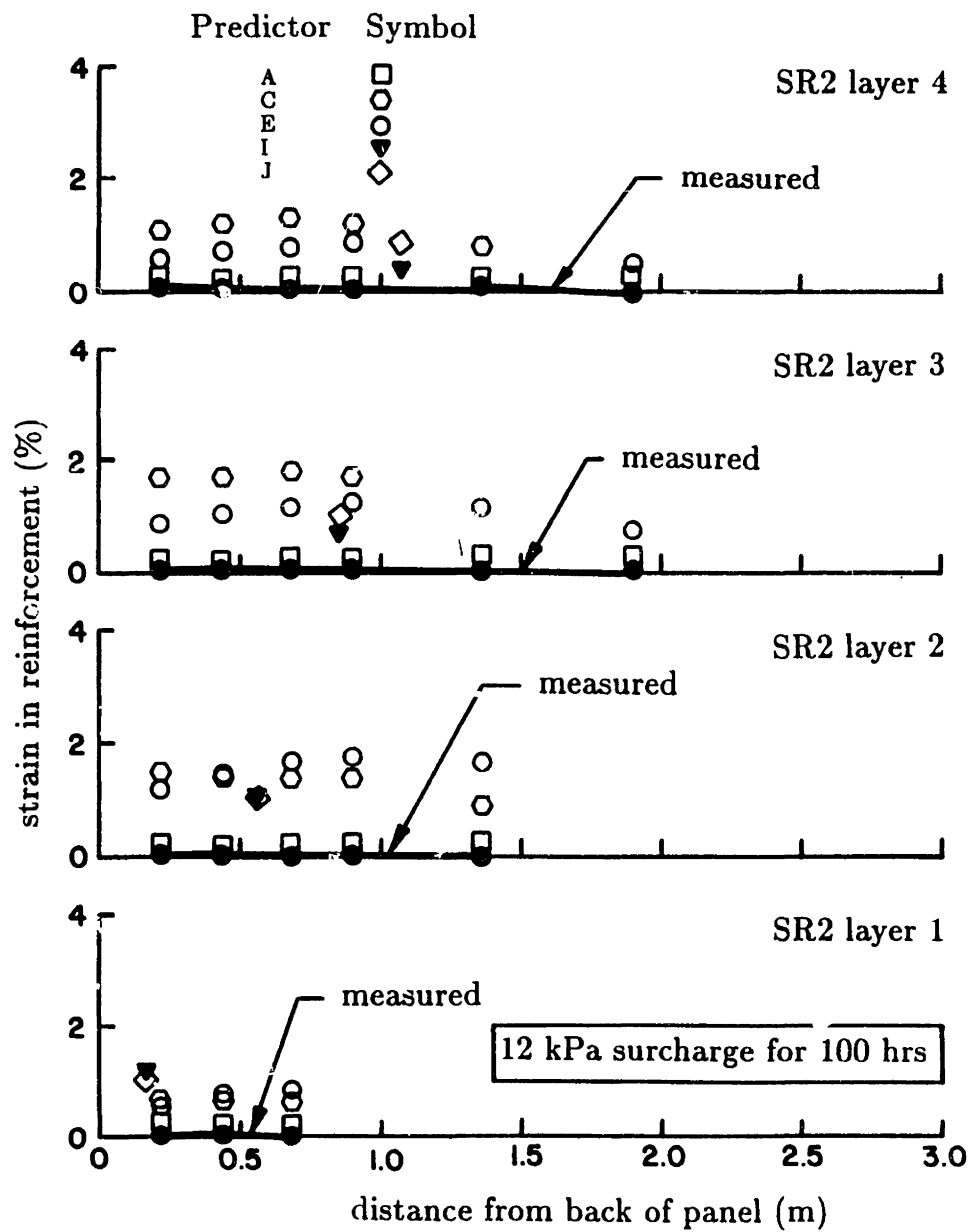


Figure 6.11: Reinforcement Strains in Propped Panel Wall after 12 kPa Surcharge for 100 hours (after Bathurst and Koerner, 1987).

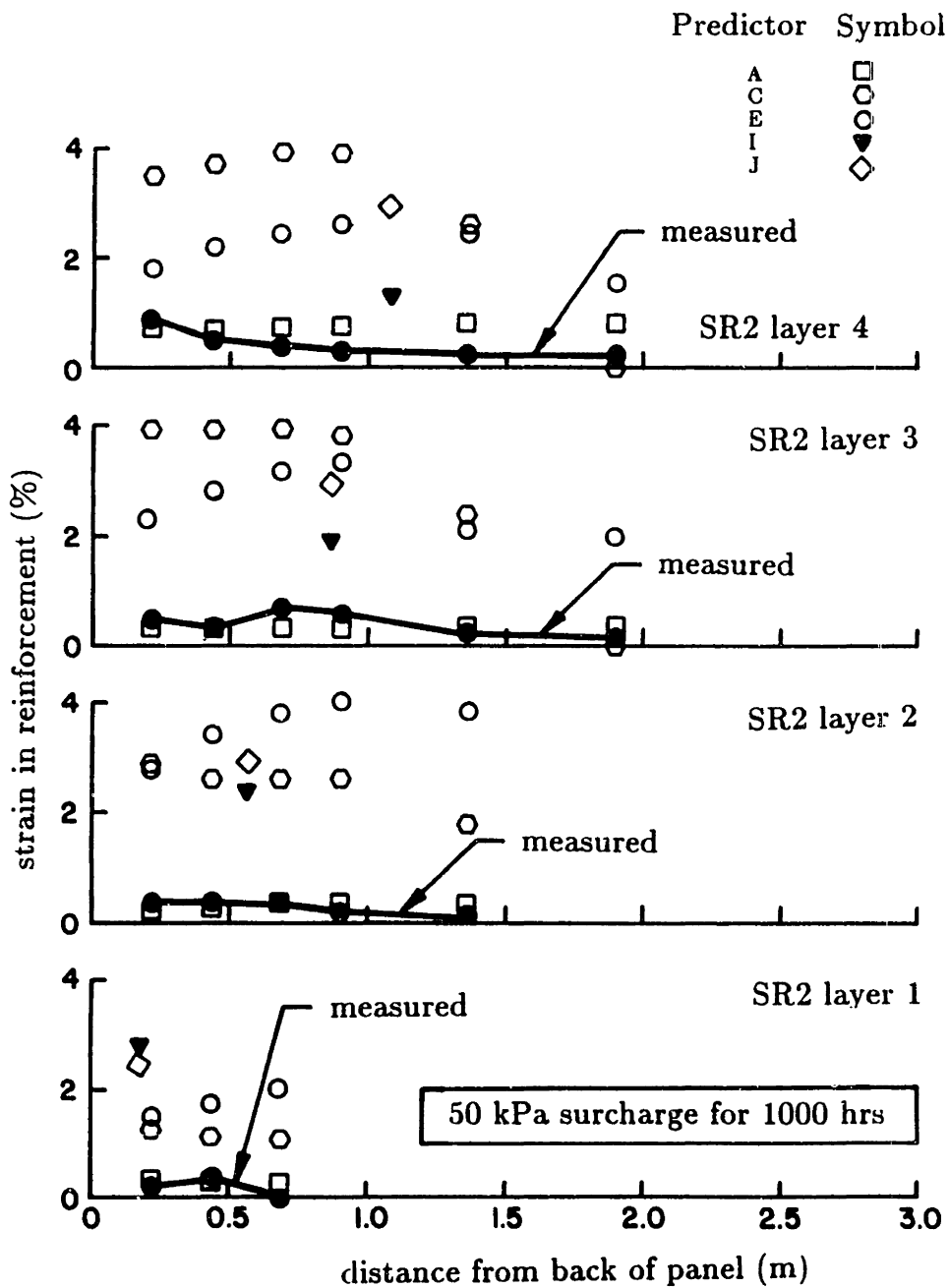
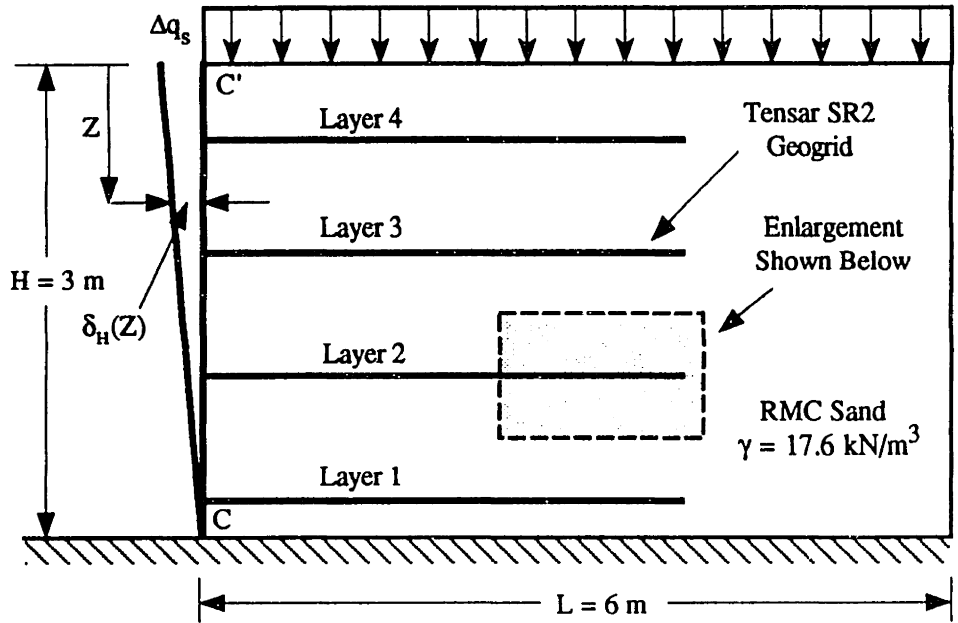
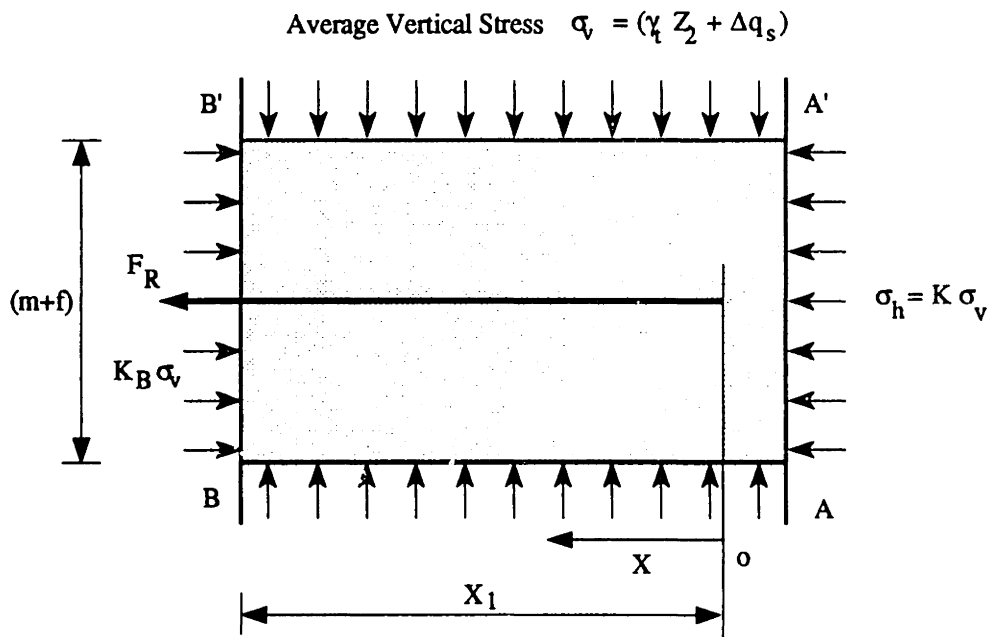


Figure 6.12: Reinforcement Strains in Propped Panel Wall after 50 kPa Surcharge for 1000 hours (after Bathurst and Koerner, 1987).



(a) General Scheme.



(b) Local Stress Equilibrium.

Figure 6.13: Application of APSR Cell for Estimating Tensile Loads in RMC Geogrid Reinforcements.

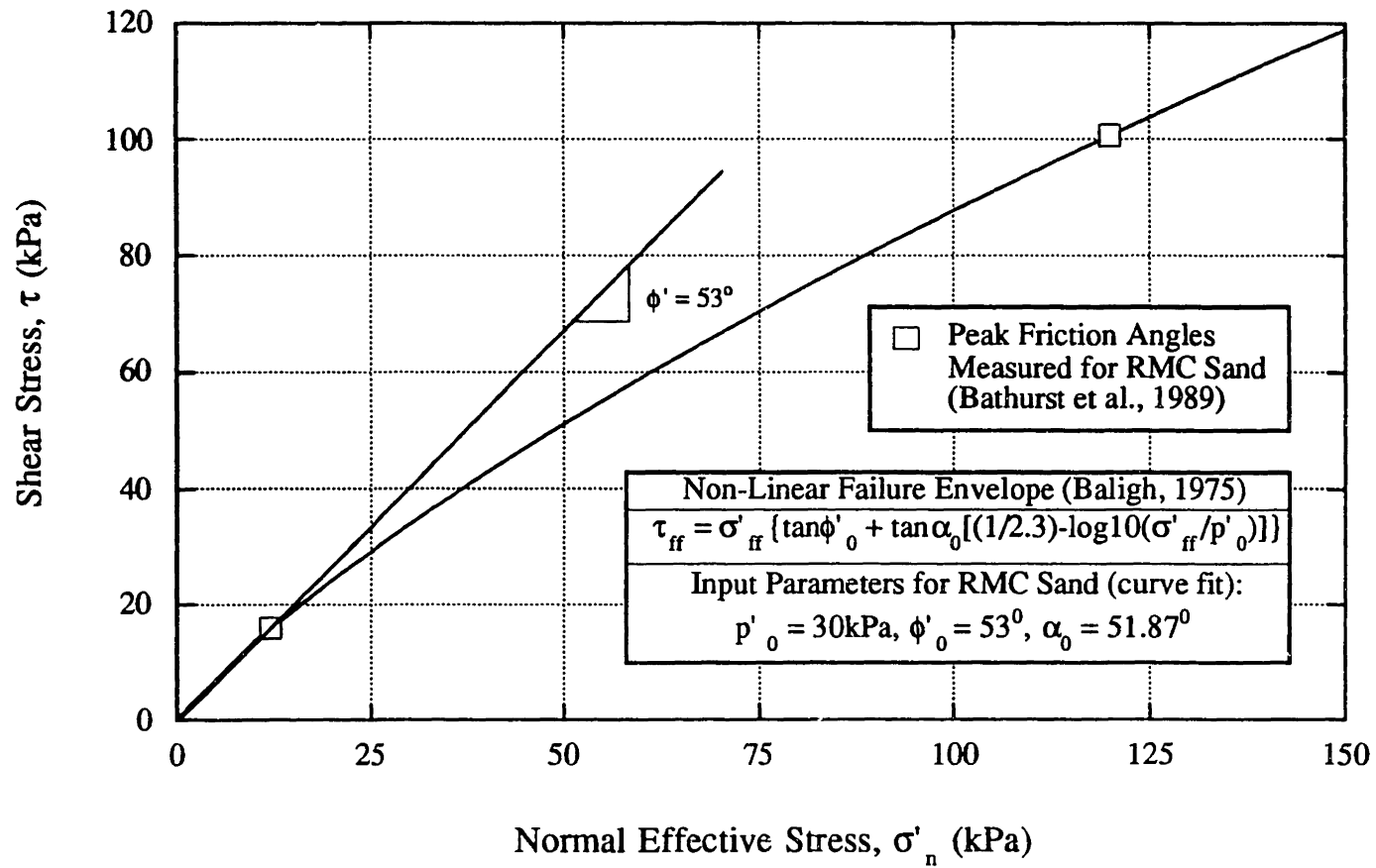
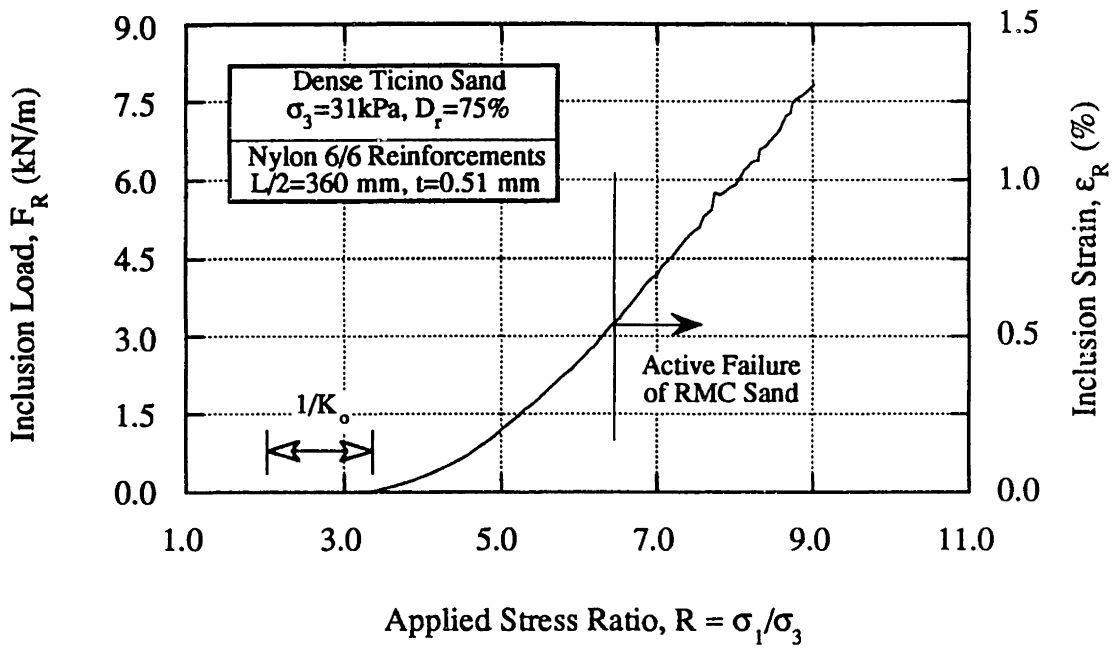
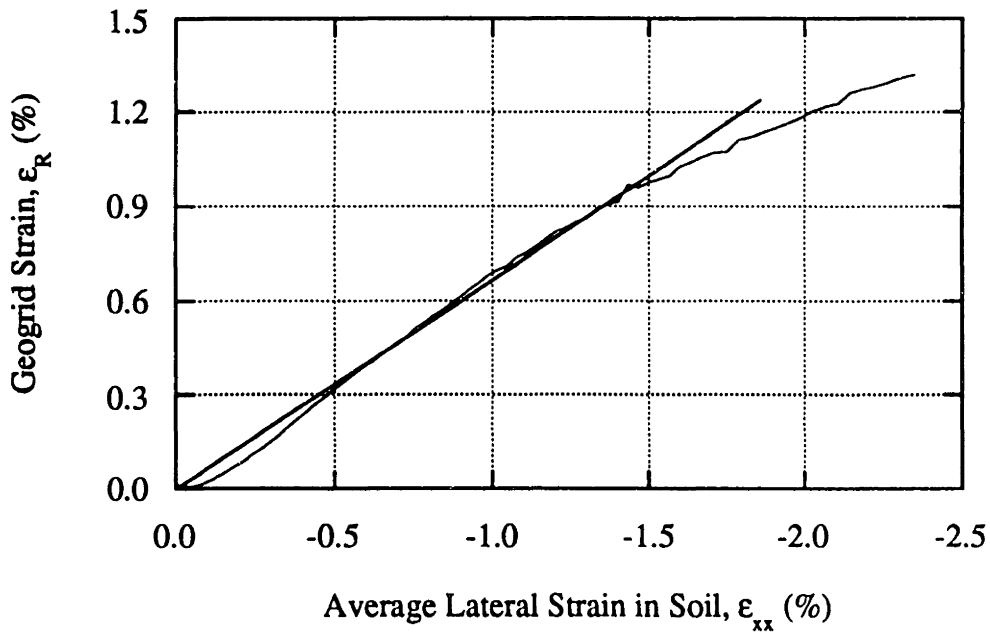


Figure 6.14: Non-Linear Failure Envelope for RMC Sand.



(a) Relationship between Applied Stress Ratio and Maximum Tensile Force in Nylon 6/6 Inclusion.



(b) Tensile Force in Nylon 6/6 Inclusion as a Function of Lateral Strain in the Soil Matrix.

Figure 6.15: APSR Measurements of Load-Transfer for 360 mm Long Nylon Inclusion.

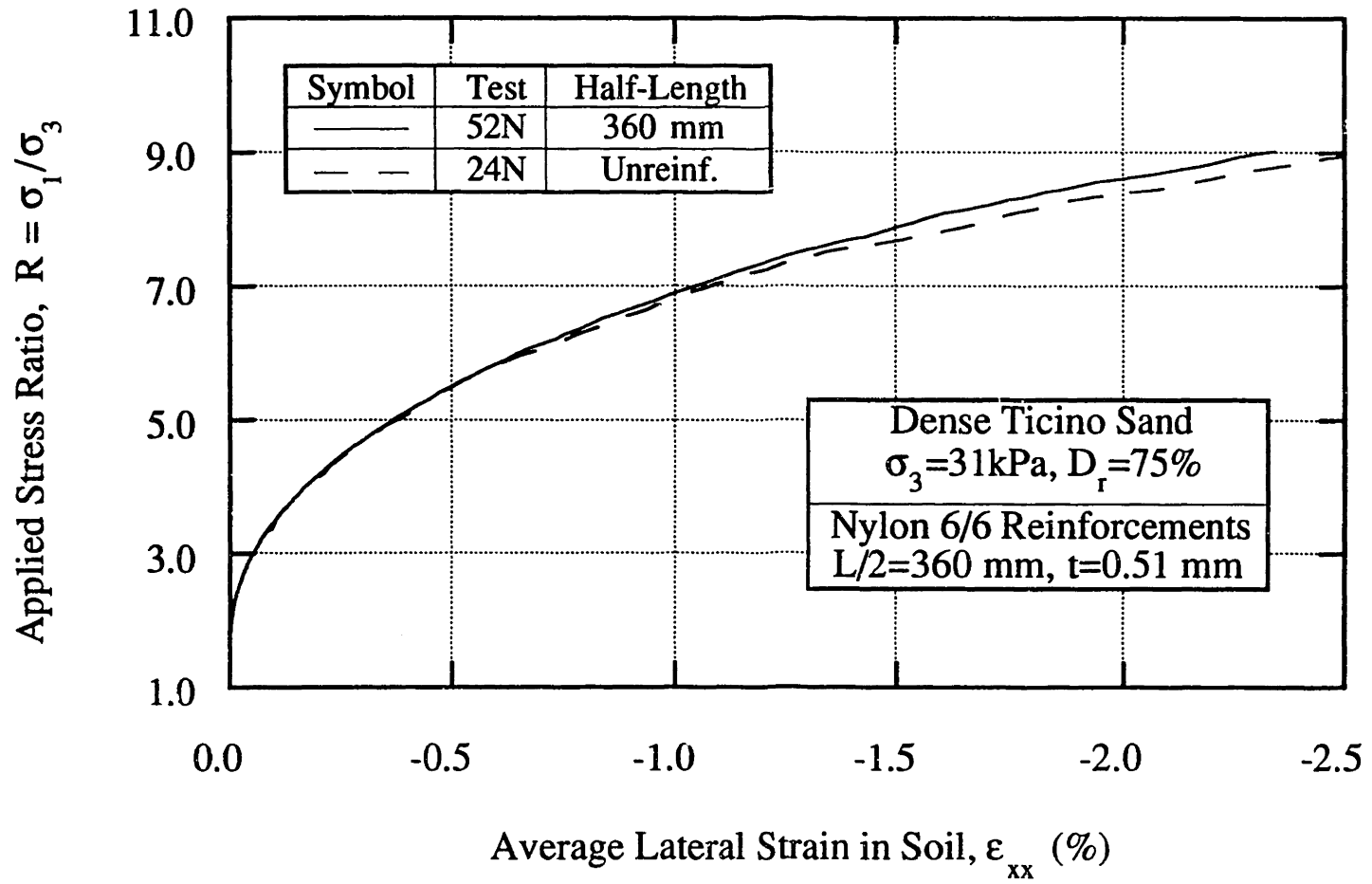
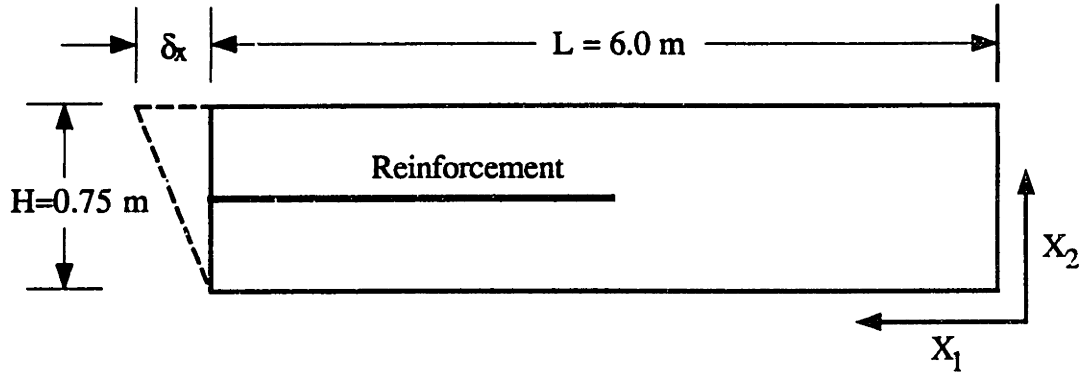
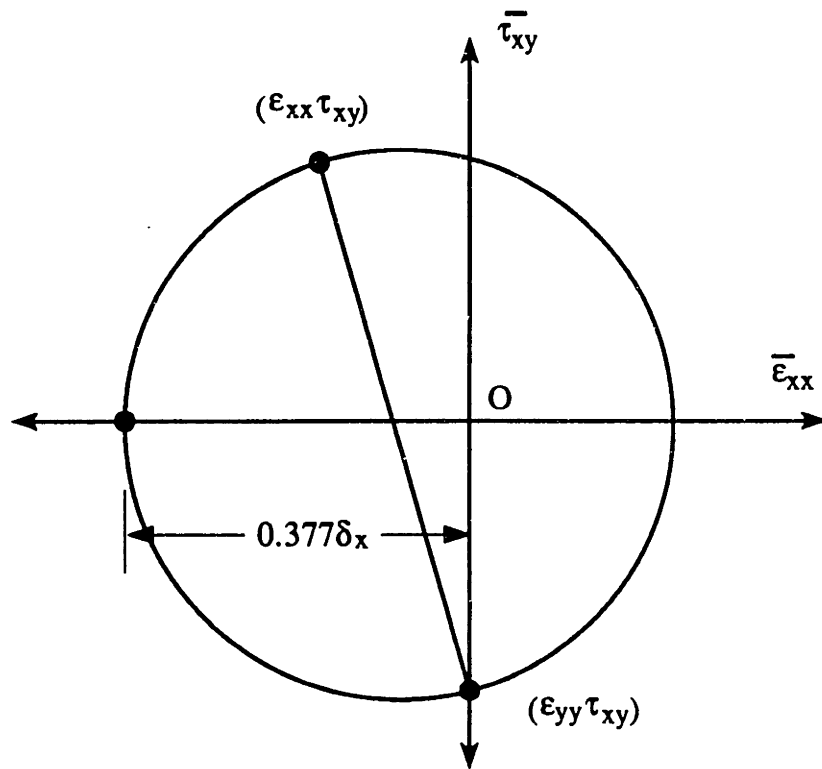


Figure 6.16: Effect of 360 mm Long Nylon 6/6 Inclusion on Externally Measured Lateral Strain in the Soil Matrix.



(a) Deformation of Reinforced Soil Layer due to Facing Movement.



(b) Relationship between Horizontal Strain and Maximum Strain in the Soil Layer.

Figure 6.17: Computation of Maximum Strains in the Backfill.





# Chapter 7: Summary and Conclusions

## 7.1. Summary

This thesis describes the measurements of tensile stresses which develop in geosynthetic inclusions due to shearing of the surrounding soil mass in a plane strain compression mode. The experiments have been performed in a laboratory device, referred to as the Automated Plane Strain Reinforcement (APSR) cell, originally designed and built by Larson (1992). The APSR cell simulates one half of the soil-reinforcement composite element shown in Figure 2.1 and hence, allows direct measurements of the tensile stress at the center of the reinforcing inclusion as the surrounding soil matrix is sheared.

The APSR cell contains a soil specimen measuring 570 mm high by 450 mm wide by 152 mm deep (plane strain direction) enclosed in a thin rubber membrane (Figure 2.3). The cell can accommodate a single reinforcing inclusion in size up to 450 mm in length (equivalent to a total sample length,  $L = 900$  mm) and 150 mm in width, oriented parallel to direction of the minor principal stress,  $\sigma_3$ . All rigid contact surfaces between the specimen membrane and cell are lubricated with two layers of silicone grease separated by thin rubber sheet to minimize surface friction. The soil specimen is sheared by increasing the major principal stress,  $\sigma_1$ , which is applied through two rigid end platforms. As the specimen is sheared, it deforms freely in the lateral direction against an airbag (rubber bladder) which applies uniform confining stress,  $\sigma_3$ , to the front face of the specimen. The APSR cell maintains plane strain conditions throughout the test using an active compensation mechanism provided by pressurized water

diaphragms within the side walls. Active control is also used to maintain the inclusion symmetry along the rear wall of the cell. The reinforcing inclusion is clamped externally to a load cell and is held in position by jacking against a support arch. A custom-built infrared sensing device monitors position of the location equivalent to the centerline of an inclusion with length,  $L$  ( $X$  in Figure. 2.2). As the soil is sheared, a linear actuator arm manipulates the position of the inclusion such that there is no displacement of the inclusion at the reference point.

The APSR cell has been extensively instrumented with a variety of electronic transducers. Linear voltage displacement transducers (LVDT's) and magnetic proximity sensors measure movements of the specimen boundaries and pressure transducers measure the applied major, minor, and intermediate principal stresses. A load cell measures the maximum tensile loads in the inclusion while additional instrumentation can be designed to measure the local strains and/or stresses at locations along the inclusion for certain types of reinforcing materials. The APSR cell is fully automated with eight digital feedback control loops that use PID algorithms to control displacements of the platform jacks, the positions of the plane strain walls and the inclusion, and the confining air pressure. An IBM compatible personal computer (PC) performs the digital feedback control and also acquires, processes, and displays data while the test is running. All these tasks are coordinated and managed by a sophisticated, real-time software written in C language. The software employs a Graphical User Interface (GUI) to provide the user with numerous options for controlling the loading sequence, inspecting the test results, and interfacing additional instrumentation at any selected time during the process.

In order to make the device more robust and hence, improve test reliability and efficiency, the following modifications have been made in the hardware and software configuration of the APSR cell:

1. Use of an airbag for applying the confining pressure.
2. An independent referencing frame and non-contact proximity sensors for the plane strain side walls.
3. Use of an infrared sensor for reinforcement movement detection.
4. Design of a rigid actuator arm for controlling the position of reinforcement.
5. Implementation of full digital feedback control and development of a modular, real-time software system integrating the control and data acquisition tasks.

These and other changes have enabled the APSR cell to achieve extremely high sensitivity and accuracy in the measurements and control of the boundary conditions. The digital feedback control system is able to maintain: 1) all platform pistons within 2  $\mu\text{m}$  of their target position at a controlled displacement rate as low as 0.01 mm/min.; 2) the out-of-plane strain within 0.001% which corresponds to less than 1  $\mu\text{m}$  deflection of an individual side wall; 3) the steady-state value of confining air pressure within 0.05 kPa; and 4) the reinforcing inclusion within  $\pm 1 \mu\text{m}$  of its reference position. The modified APSR cell now provides a capacity for routine measurements of load-transfer using a variety of reinforcing inclusions including extensible sheets and grids.

A comprehensive program of load-transfer measurements was performed using steel and Nylon 6/6 flat sheet reinforcements whose axial stiffness represent the upper and lower limits for typical reinforcing materials used in practice. The experimental program focused on the effects of mobilized shear, and inclusion length and axial stiffness on the tensile stresses transferred to the inclusions. A total of 10 high quality

tests were performed using steel (axial stiffness,  $J = 52,500$  kN/m, thickness,  $t = 0.254$  mm) and Nylon 6/6 ( $J = 980$  kN/m,  $t = 0.51$  mm) sheet inclusions embedded in Ticino sand at a relative density,  $D_r = 75\%$ , and sheared at a confining pressure,  $\sigma_3 = 31$  kPa. Inclusions with four different lengths,  $L/2 = 360, 270, 180, 90$  mm, were used to evaluate the effects of inclusion length on load-transfer. A detailed comparison between the results of APSR tests on planar (sheet) inclusions and predictions of a shear-lag model, developed by Abramento and Whittle (1993), has been used to propose a procedure for adapting the original linear, elastic shear-lag solutions to account for the measured non-linear behavior of the matrix material (Ticino sand).

Further experiments have been carried out using a polyester, woven geogrid product sold under the trade name Fortrac<sup>®</sup>. The axial stiffness of Fortrac geogrid inclusions ( $J = 1400$  to  $2000$  kN/m) is of the same order of magnitude as the Nylon 6/6 sheet reinforcements used in the APSR research. A series of ten successful tests was performed using grid inclusions with three different lengths,  $L/2 = 360, 195,$  and  $105$  mm, to evaluate the effects of inclusion length and geometry on tensile load-transfer.

Finally, Chapter 6 summarizes the direct application of the APSR load-transfer measurements to interpret working stress behavior of two well instrumented geogrid-reinforced. This case study included careful measurements of tensile strains within the layers of geogrid reinforcements and was part of a long-term research project conducted by the Royal Military College of Canada (RMC). The observed horizontal movements of the facing panels were used to estimate the stress state prevailing in the sand backfill such that maximum reinforcement strains could be estimated directly from the APSR measurements.

## 7.2. Conclusions

The following main conclusions are based on the APSR test results and shear-lag analyses presented in the previous chapters.

### 7.2.1. Shear Behavior of Unreinforced Ticino Sand

The procedures established for specimen preparation, consolidation, and shear in the APSR cell provide very consistent repeatable results. The plane strain shear behavior of slightly over-consolidated ( $OCR = 2$ ), medium-dense ( $D_r = 75\%$ ), unreinforced Ticino sand at confining stress,  $\sigma_c = 31$  kPa (Figure 3.16) shows that:

1. The soil exhibits a non-linear stress-strain behavior throughout shearing with secant shear modulus decreasing as a function of strain level, from  $G_m = 30$  MPa at  $\epsilon_a = 0.05\%$  to  $G_m = 2.5$  MPa at  $\epsilon_a = 2.5\%$ .
2. The peak shear ratio,  $R = \sigma_1/\sigma_3 = 9$ , which corresponds to friction angle,  $\phi' = 53^\circ$ , is mobilized at an axial strain,  $\epsilon_f = 2.5\%$ .
3. The intermediate principal stress ratio,  $b = (\sigma_2 - \sigma_3)/(\sigma_1 - \sigma_3)$ , ranges from 0.35 to 0.45 with the higher value typically achieved as the soil specimen approaches failure.
4. The volumetric strains are mostly compressive throughout shearing. The peak volumetric compressive strain,  $\epsilon_{vol} = 0.3\%$  is achieved at an axial strain,  $\epsilon_a = 1\%$ .
5. The secant value of equivalent Poisson's ratio for soil increases in a non-linear fashion from,  $\nu_m < 0.1$  at the beginning of the test to  $\nu_m \approx 0.5$  as failure is approached.

The shear behavior of Ticino sand measured in the APSR cell follows the trends reported for plane strain testing of other sands in the literature. The high values of

friction angle measured in the APSR cell are due to the plane strain boundary condition and relatively low confining stress,  $\sigma_c = 31$  kPa, employed in the APSR test.

Relatively large compressive volumetric strains are attributed to three factors: a) increased confinement in plane strain, b) the sand is medium dense ( $D_r = 75\%$ ), and c) the specimen is sheared in the isotropic plane.

## 7.2.2. Load-Transfer in Planar Inclusions

### 7.2.2.1. Measurements of Inclusion Tensile Loads

The results of APSR tests using two types of reinforcements with planar (flat sheet) geometry demonstrate that it is possible to obtain reliable and repeatable measurements of tensile loads in the inclusion provided the inclusion position is controlled at the exit point within  $\pm 1$   $\mu\text{m}$ . The plots of the maximum tensile load,  $F_R$ , (measured at the exit point) versus the externally applied stress ratio,  $R = \sigma_1/\sigma_3$ , (Figures 4.5 and 4.10) show that the tensile load-transfer does not begin until the applied stress ratio,  $R \geq R^*$  where  $2.5 < R^* < 4.5$  depending on the length and axial stiffness of the inclusion. The higher values of  $R^*$  are observed for short inclusions mainly because small non-uniformity in the distribution of lateral strain has an effect of delaying the onset of load transfer in the short inclusions. Measurements of inclusion position indicate that before tensile load-transfer begins, the inclusion is forced into compression and moves out of the cell by extremely small amounts that range from 1 to 50  $\mu\text{m}$  depending on the length and stiffness of the inclusion.

The relationship between the inclusion centerline load,  $F_R$ , and the applied shear stress ratio is typically non-linear with both  $F_R$  and the transfer gradient,  $dF_R/dR$ , increasing with the stress ratio,  $R$ . The centerline tensile loads in steel sheet reinforcement increase significantly with length of the reinforcement. Loads measured

for the longest steel inclusion ( $L/2 = 360$  mm) at  $R = 7$ , are almost 8 times the magnitude of those for the shortest inclusion ( $L/2 = 90$  mm). Although the tensile loads in the Nylon 6/6 inclusions tend to increase with the inclusion length, the gain is only 20% for an increase in the inclusions half-length from  $L/2 = 180$  to 360 mm. At the same applied shear stress and inclusion length, the maximum tensile loads increase with the axial stiffness of the inclusion. Tensile loads in the Nylon 6/6 sheet reinforcements are up to 5 times smaller than in the steel reinforcements. The maximum tensile strain in the inclusion is approximately 0.05% and 1.0% for the steel and Nylon 6/6 inclusions respectively. The ratio of average lateral strain in the soil matrix to maximum tensile strain in the reinforcement varies from 7.5 for the steel reinforcement to 3.0 for Nylon 6/6. These results clearly show that the conventional assumption of strain compatibility which equates lateral strains in the soil mass with reinforcement strains is not realistic.

The reinforcement loads can be well described as linear functions of the average lateral strain,  $\epsilon_{xx}$ , in the sand matrix with average linear regression coefficient,  $r^2 = 0.9938$ . The load-transfer gradient,  $dF_R/d\epsilon_{xx}$ , depends on the length and stiffness of the inclusion and ranges from 7000 to 550 kN/m for steel reinforcements of lengths,  $L/2 = 360$  mm and 90 mm respectively, and 370 to 150 kN/m for Nylon 6/6 sheet reinforcements of the same lengths. These results show that tensile stresses develop in the reinforcement as a direct result of lateral strains within the soil matrix. For short inclusions ( $L/2 \leq 180$  mm), a small offset lateral strain,  $\epsilon_{xx}^* = 0.1$  to 0.15% is observed at the specimen boundary before the onset of tensile load-transfer. This offset strain can be attributed to small non-uniformities in the distribution of the lateral strains within the soil specimen due to cohesion of the lubricant acting along the plane strain side walls.

In general, the ability of a reinforcement to modify the externally measured shear behavior of the soil matrix is proportional to the magnitude of tensile loads carried by the inclusion. The steel sheet reinforcements cause a significant increase in the shear stiffness of the soil and reduce the average lateral strains,  $\epsilon_{xx}$ , by as much as 50%. Nylon 6/6 reinforcements, on the other hand, have virtually no effect on the externally measured stress-strain behavior of Ticino sand.

#### 7.2.2.2. Comparison with Shear-lag Predictions

The shear-lag analysis developed by Abramento and Whittle (1993) expresses reinforcement stresses as closed-form functions of the geometry and elastic properties of the constituent materials (soil and inclusion). Within the range of strains encountered in the APSR cell, both steel and Nylon 6/6 reinforcements can be treated as linearly elastic materials with the axial stiffness,  $E_f$ , obtained from in-isolation uniaxial tensile tests. The shear-lag predictions of load-transfer are not very sensitive to the Poisson's ratio,  $\nu_f$ , of reinforcement (Figure 4.21a) and any realistic value of  $\nu_f$  can be used. In the past, Larson (1992) and Abramento (1993) approximated the soil behavior in the APSR cell by assuming a constant value of the equivalent secant shear modulus,  $G_m$ , and Poisson's ratio,  $\nu_m$ , for shear-lag computations. The measured shear stress-strain behavior of medium-dense Ticino sand in the APSR cell is highly non-linear (Figures 3.15 and 3.16). Therefore, use of constant elastic soil parameters does not provide satisfactory predictions for the entire range of inclusion lengths and materials employed in the present research program.

An improved approximation procedure is proposed in which non-linearity in the soil behavior is taken into the account by expressing the secant elastic parameters ( $G_m$ ,  $\nu_m$ ) as continuous functions of the applied stress ratio,  $R$ , using a fourth order polynomial (Equation 4.9). These parameters underestimate the average stiffness of the



soil matrix. Hence, a further reduction coefficient,  $\beta$ , is used to compute an the effective value of the applied stress ratio,  $\bar{R}$  (Equation 4.10). The coefficient  $\beta$  reflects the fact that the externally measured stress ratio,  $R$ , is an upper bound on the effective shear stress within the soil matrix. The APSR data are well described by the shear-lag analysis with a constant coefficient  $\beta$  which is fitted to the experimental data. The results (Figures 4.23, 4.25) show  $\beta = 0.78$  for steel and  $\beta = 0.65$  for Nylon 6/6 sheet reinforcements.

The shear-lag predictions of load-transfer using the proposed methodology provide excellent overall agreement with the measured loads for both steel and Nylon 6/6 sheet reinforcements (Figures 4.23 and 4.25). The analyses tend to slightly under predict the inclusion forces for two cases: a) the initial portion of the load transfer ( $R < 5$ ), and b) the shortest inclusion ( $L/2 = 90$  mm). These comparisons demonstrate that the shear-lag analysis gives reliable predictions of load-transfer for planar inclusions with stiffness ratio,  $320 < E_f/G_m < 35000$ , when non-linearity in soil behavior is simulated by varying the secant shear modulus and Poisson's ratio of the matrix. The shear-lag predictions of tensile load-transfer as a function of inclusion length (Figures 4.24 and 4.26) indicate that the maximum tensile stresses measured in the longest steel sheet inclusion used in the APSR test ( $L/2 = 360$  mm) represent half the stresses that would be transferred to an infinitely long inclusion. For relatively flexible Nylon 6/6 reinforcements, however, the analysis shows that the stresses in the 360 mm long inclusion are very close to what may be expected in a field situation.

### 7.2.3. Behavior of Geogrids in the APSR Cell

The results from a series of tests performed using two grades of Fortrac geogrid reinforcements embedded in Ticino sand at a relative density of 75% and sheared at a confining pressure of 31 kPa, show that the load-transfer characteristics of the grid

inclusions are qualitatively very similar to the planar inclusions. Tensile loads are not transferred to the grid reinforcements until the applied stress ratio is higher than a characteristic value,  $R^*$ , where,  $R^* \approx 4.5$ . For both Fortrac 110/30-20 and 80/30-20 grid reinforcements, an increase in inclusion half-length from,  $L/2 = 109$  mm to  $L/2 = 193$  mm results in an approximately 100% increase in the inclusion tensile loads. There is an excellent agreement between the measured tensile load in Fortrac grid inclusions with half-length,  $105 \leq L/2 \leq 195$  mm and axial stiffness,  $1400 \leq J \leq 2000$ , and shear-lag prediction using the coefficient,  $\beta = 0.65$  (Figures 5.13 and 5.14). These comparisons demonstrate that within these ranges of parameters, there is no difference between the load-transfer behavior of a grid and a planar inclusion of equivalent axial stiffness.

However, the similarities between the behavior of grid and planar inclusions do not extend to the inclusion with half-length,  $L/2 = 360$  mm. Although the measured tensile loads for the  $L/2 = 360$  mm inclusion are not significantly different from those with  $L/2 = 193$  mm (Figure 5.2), the longer grid inclusion has a much more significant effect on the overall stress-strain behavior of the soil matrix as shown in Figures 5.7 and 5.8. These results suggest that the grid geometry affects the tensile load distribution and provides greater reinforcing efficiency compared to a planar reinforcement when the size of reinforced zone is significant compared to the size of the soil matrix. Further experiments conducted to verify this hypothesis indeed show that if most of the transverse members (ribs) are removed from the longest grid inclusion, its behavior in the APSR cell approaches that of a planar inclusion of equivalent stiffness (refer to Figure 5.17). These results suggest possible limitations of all analyses including the finite element method which treat grid reinforcements as planar elements with equivalent stiffness and do not explicitly account for the effects of

transverse elements on the performance of grids at working stress levels. More detailed studies, however, are necessary to validate these findings.

#### 7.2.4. Geogrid Reinforced Wall Case Study

Chapter 6 shows that the APSR measurements are consistent with maximum reinforcement strains measured for the RMC geogrid walls built using both incremental and propped construction techniques. These calculations were based on measured wall movements and hence do not represent true predictions of wall performance. In contrast, some of the most commonly used limit equilibrium design methods fail to capture the differences in behavior due to construction techniques and grossly over-predict reinforcement strains in both model walls. The results show that the APSR cell, in conjunction with simple elastic equilibrium analyses, such as the shear-lag analysis proposed by Abramento and Whittle (1993), can provide a useful framework for predicting and interpreting the stresses within reinforced soil structures at working loads. Further development of such models will have to account for effects of factors such as compaction, base restraint, and connection between reinforcement and face panel, which are known to influence the reinforcement loads in full scale reinforced soil structures.

### 7.3. Suggestions for Future Work

Suggestions for future research in the APSR project are divided into the following two categories: 1) improvements in the APSR cell equipment and 2) experimental programs using the APSR cell.

### 7.3.1. Improvements in APSR Cell

Although the APSR cell in its present configuration is suitable for routine testing using wide variety of reinforcing inclusions and soils, several minor modifications can improve its versatility and performance. The gap between the infrared source and detector pair used to monitor inclusion position (refer to Section 2.2.6) is presently about 5 mm. For grid inclusions with thickness,  $t = 3$  to 4 mm, the clearance between the inclusion and sensor support fork (Figure 2.14) is very small. Even a small misalignment between the two causes the inclusion to rub against the fork. This usually results in gross errors in the reinforcement position measurement and temporary instability of the reinforcement feedback control system. It is possible to increase the air gap between the infrared source and detector to about 10 mm to provide more leeway without sacrificing sensitivity or performance of the measurement system.

At present, four hydraulic jacks are used to move the rigid platforms that apply the major principal stress to the specimen. Compliance of this hydraulic system varies with the amount of entrapped air which makes it difficult to achieve fast, responsive control of the platforms, especially during stress-controlled test mode. For creep and cyclic APSR tests, it would be advantageous to replace the hydraulic pistons by direct linear drives of the kind used for controlling the reinforcement position. Compared to hydraulic systems, direct drives are easier to control because they have relatively low, constant compliance and negligible dead time.

Larson (1992) attempted to measure the tensile stress distribution along the length of steel sheet inclusions using conventional foil type strain gages. However, these gages respond to small changes in the curvature of the inclusion due to confining stresses acting normal to the plane of the inclusion and hence, do not perform satisfactorily in the APSR cell. Semiconductor (piezo-resistive) strains gages hold

promise for measurements of local strains in planar inclusions. Due to their extremely small size (0.5 mm typically), semiconductor strain gages are less susceptible to flexural strains. Semiconductor strain gages also offer an added benefit of approximately 50 times higher sensitivity (gage factor) compared to metal foil strain gages.

The stress ratio varies throughout the APSR soil specimen due to the additional confinement provided by tensile stresses in the reinforcement and the action of surface friction. Additional instrumentation can be added to measure the local stresses within the soil specimen. Miniature earth pressure cells may be designed to measure local normal stresses acting along the rear wall of the cell and along the reinforcement.

### 7.3.2. Experimental Program

The APSR tests done so far using Fortrac<sup>®</sup> polyester fiber geogrids have shown that although the tensile loads in the grid inclusions are similar in magnitude to loads in planar (sheet) reinforcements of comparable stiffness, there are significant differences in the load-transfer behavior of grids observed as function of length. Compared to planar inclusions of equivalent stiffness and length, a grid inclusion is able to support higher value of externally applied shear stress for a given tensile stress in the inclusion. This result has significant impact on both the design of reinforced structures as well as the design of future grid materials. Further studies using different types of geogrids are required to quantify the effects of inclusion geometry (i.e., length, number and spacing of transverse elements, thickness of junctions, etc.) on externally measured shear behavior of the soil.

The load-transfer behavior of linear elastic reinforcements with planar geometry in the APSR cell is well established. The next step is to introduce planar materials with

strongly non-linear stress-strain response such as woven and non-woven geotextiles. These materials have been used in several major reinforced wall construction projects around the world (Allen et al., 1992; Billard and Wu, 1991). The conventional reinforced wall design methods, such as tie-back wedge analysis, are unsatisfactory for highly flexible materials. It has been found that under working stress levels, strains developed in these materials typically range from 1 to 5%. The strain levels corresponding to the stresses predicted by most design methods are much larger than these strains (Rowe and Ho, 1993). Clearly, either the axial stiffness of certain geosynthetics change under confined conditions or the assumptions made by the current design methods are not realistic. There is evidence for both effects. In one study of geosynthetic reinforced walls (Rimoldi, 1988), the ratio of reinforcement tension calculated using in-situ parameters to tensile forces measured (or deduced from strain) was found to be anywhere between 2.5 to 13.5. The effect of confining stress on the axial stiffness of needle-punched and spun-bonded non-woven geotextiles has long been documented (McGown et al., 1982; Kokkalis and Papacharisis, 1989). Recent research by Ling et al. (1992), however suggests that earlier estimates of such effects may have been exaggerated due to improper testing procedures. Using a modified triaxial apparatus Ling et al. (1992) found four to five fold increase in the small strain (< 5%) secant modulus of spun-bonded and needle-punched geotextiles. Even such a modest increase in the axial stiffness of the reinforcement can be verified in the APSR cell by back calculating the stiffness from careful measurements of load-transfer.

With little modification of the equipment and test procedure, the existing APSR cell can be adapted to perform standard plane strain compression tests using woven and non-woven materials. An epoxy impregnation technique described by Ballegeer and Wu (1993) can be used to reinforce and grasp the free end of a flexible geotextile inclusion. Radiography, which has been used for measuring strains in non-woven

geotextiles in pull out tests (Kharchafi and Dysli, 1993), can be used in the APSR cell to avoid the problems associated with the use of strain gages on extensible materials. Another promising technique comprises tracking the movements of miniature hall effect sensors mounted onto the inclusion with respect to strong, permanent magnets buried in the soil. The purpose of such tests would be to verify: a) if shear-lag analysis, modified to account for reinforcement non-linearity is applicable and b) effect of confining stress on the axial stiffness of non-woven geosynthetics. Finally, the design of structures built using flexible geotextile reinforcement is usually limited by the amount of allowable deformations under working loads and yet there is, at present, no reliable method of predicting such deformations. The APSR test results combined with the shear-lag analysis can be used to formulate design guidelines regarding working stress deformations in reinforced walls built using highly extensible materials.

Depending on the site conditions, as much as 50% of the total cost of reinforced wall construction may be spent on high quality granular backfill (Durukan and Tezcan, 1992). Therefore, there is a strong incentive to use locally available low grade backfill materials for reinforced soil construction. In fact, clayey and silty-clay soils have been successfully used with geosynthetics on many occasions (Hayden et al., 1991; Sego et al., 1990). However, with very limited research available, knowledge in this area is generally lagging the state-of-the-practice. The APSR measurements of load-transfer using fine silty sands or clayey soils can be used to verify whether the shear-lag analysis can be used to describe reinforcing behavior of matrix materials with low values of shear stiffness and friction angle.

All polymeric materials exhibit time dependent stress-strain properties to a varying degree. Technical literature is abundant with information on the creep behavior of geosynthetics. Koerner et al. (1993) present a comprehensive review of available

data on creep and stress relaxation of geotextiles, geogrids, and geomembranes. Most reported creep experiments are product specific, in-isolation, and usually performed at ambient temperature. Confinement has been shown to affect the creep (and stress relaxation) behavior of certain types of non-woven geotextiles. The APSR cell can be used to study the effects of confined creep on load-transfer in woven and non-woven geotextile reinforcements under very realistic loading conditions.

The compaction induced lateral stresses can significantly increase reinforcement tension at shallow depths in reinforced soil walls built using relatively flexible reinforcements (Ehrlich and Mitchell, 1994). The effect of compaction can be simulated in the APSR cell by performing one or more unload-reload cycles. Limited information from one experiment performed in the APSR cell indicates that most of the induced load in the reinforcement remains locked-in during unloading of the major principal stress (see Figure 4.5).

Pull out tests have been widely used to investigate the shear strength of soil-reinforcement bond (Juran et al., 1988; Farrag et al., 1993). Abramento (1993) used the APSR cell to perform pull out tests on Nylon 6/6 sheet inclusions. The APSR cell offers the following advantages over a conventional pull out box design.

1. In a conventional pull out box, the sample is build in successive layers and the reinforcement is pulled out under  $K_0$  condition. However, in real reinforced structures, the soil is usually under active conditions. The APSR cell allows independent control of the major and minor principal stresses and hence it is possible to perform pull out tests at various shear stress levels.
2. Using X-ray technique, Kharchafi and Dysli (1993) found that the soil deformations in a pull out box are highly asymmetrical about the centerline



defined by the reinforcement because: a) the top boundary in a conventional pull out box is usually flexible while the bottom is rigid, and b) non-uniform stress distribution due to self weight. Similar results have been reported by Gourc et al. (1992) using centrifuge tests. Since the loading plane in the APSR cell is horizontal and there are identical boundary conditions on either side of the reinforcement, the specimen is symmetrical.

3. In most pull out tests the reinforcement is pulled out at a constant rate of displacement. As noted by Farrag et al. (1993), very few pull out tests are reported under stress controlled mode. The APSR cell can be used to pull the reinforcement under constant stress (load) using feedback control. It is also possible to study creep behavior by maintaining a constant load on the reinforcement.
4. Because of their large sizes, the major principal stress in most pull out boxes is restricted to 35 to 70 kPa. The major principal stress in the APSR cell can be as high as 300 kPa.
5. Because the APSR cell uses an active plane strain control system, the side walls are sufficiently thin to allow measurements of reinforcement and soil strains using radiography.



# References

- Abramento, M (1993), Analysis and Measurement of Stresses in Planar Soil Reinforcements, Ph.D. Thesis, Massachusetts Institute of Technology, Cambridge, MA.
- Abramento, M. and Whittle, A.J. (1993), "Shear-Lag Analysis of a Planar Soil Reinforcement in Plane Strain Compression," *ASCE Journal of Engineering Mechanics*, v. 119, n. 2, pp. 270-291.
- Allen, T.M., Christopher, B.R., and Holtz, R.D. (1992) "Performance of a 12.6 m High Geotextile Wall in Seattle, Washington," *Proceedings of International Symposium on Geosynthetic Reinforced Soil Reinforced Walls*, Ed. Denver, Colorado, pp. 81-100.
- An, J., Doumas, T.A., and Yorkey, T.J. (1988), "Interfacing to the IBM PC Bus", in *Interfacing Sensors to IBM PC*, Eds. Tompkin, W. and Webster, J., Prentice Hall, Englewood, New Jersey, pp. 59-106.
- Analog Devices (1992), *Data Converter Reference Manual*, Volume I and II, Analog Devices Inc. Norwood, Massachusetts.
- Andersen, G.R. (1991), Physical Mechanisms Controlling the Strength and Deformation Behavior of Frozen Sand, Ph.D. Thesis, Massachusetts Institute of Technology, Cambridge, MA.
- Baladi, G., et al. (1985), "Laboratory Validation of In-Situ Tests," AGI Jubilee Volume, *Proc. 11th International Conference on Soil Mechanics and Foundation Engineering*, San Francisco, CA.
- Baligh, M. M. (1975), "Cavity Expansion in Sands with Curved Envelopes", *ASCE Journal of Geotechnical Engineering*, v. 102, n. 11, pp. 1131-1146.

- Ballegeer, J.P. and Wu, J.T.H. (1993) "Intrinsic Confined and Unconfined Load-Deformation Properties of Geotextiles", *Geosynthetic Soil Reinforcement Testing Procedures*, ASTM STP 1190, Ed., S. C. Jonathan Cheng, pp. 16-31.
- Bateson, R.N. (1993), *Introduction to Control System Technology*, Fourth Edition, MacMillan Publishing Co., New York.
- Bathurst, R.J. and Koerner, R.M. (1987), "Results of Class A Predictions for the RMC Reinforced Soil Wall Trials," *The Application of Polymeric Reinforcement in Soil Retaining Structures*, Eds. P.M. Jarrett and A McGown, NATO Advanced Study Institutes Series, Kluwer Academic Publishers, pp. 127-171.
- Bathurst, R.J., Wawrychuk, W., and Jarrett, P.M. (1987), "Laboratory Investigation of Two Large-Scale Geogrid Reinforced Soil Walls," *The Application of Polymeric Reinforcement in Soil Retaining Structures*, Eds. P.M. Jarrett and A McGown, NATO Advanced Study Institutes Series, Kluwer Academic Publishers, pp. 71-125.
- Bathurst, R.J., Benjamin, D.J., and Jarrett, P.M. (1988), "Laboratory Study of Geogrid Reinforced Soil Walls," *Geosynthetics for Soil Improvements, ASCE Special Publication*, n.18, pp. 178-192.
- Bathurst, R.J., Benjamin, D.J., and Jarrett, P.M. (1989), "An Instrumented Geogrid Reinforced Soil Wall," *Proc. 12th Proc. International Conference on Soil Mechanics and Foundation Engineering*, pp. 1223-1226.
- Bennett, S. (1988), *Real-Time Computer Control*, Prantice Hall International Series in Systems and Control Engineering, Prantice Hall, New York.
- Billard, J.W. and Wu, J.T.H. (1991) "Load Test of a Large-Scale Geotextile Reinforced Retaining Wall," *Proceedings of Geosynthetics '91 Conference*, New Orleans, v. 2, pp. 537-548.
- Bonaparte, R. and Schmertmann, G.R. (1987), "Reinforcement Extensibility in Reinforced Soil Wall Design," *The Application of Polymeric Reinforcement in Soil Retaining Structures*, Eds. P.M. Jarrett and A McGown, NATO Advanced Study Institutes Series, Kluwer Academic Publishers, pp. 409-453.

- Branrup, J. and Immergut, E.H. (1975), *Polymer Handbook*, John Wiley and Sons, New York.
- Carr, J.J. (1991), *Designer's Handbook of Instrumentation and Control Circuits*, Academic Press, San Diego, California.
- Chanda, M. and Roy, S.K. (1992), *Plastics Technology Handbook*, Marcel Dekker, New York.
- Claybourn, A.F. and Wu, J.T.H. (1992), "Failure Loads of the Denver Walls by Current Design Methods", Geosynthetic-Reinforced Soil Retaining Walls, Ed. Wu, J.T.H., *Proceedings of International Symposium on Geosynthetic-Reinforced Soil Retaining Walls*, Balkema, Rotterdam, pp. 61-77.
- Cox, H.L. (1952), "The Elasticity and Strength of Paper and Other Fibrous Materials," *British Journal of Applied Physics*, v. 3, pp. 72-79.
- Derenzo, S.E. (1990), *Interfacing - a Laboratory Approach Using the Microcomputer for Instrumentation, Data Analysis, and Control*, Prentice Hall, Englewood, New Jersey.
- Deterling, P.A. (1984), Behavior of Sand Stressed under Three Independently Controlled Principal Stresses, S.M. Thesis, Massachusetts Institute of Technology, Cambridge, MA.
- Duncan, J. M. and Chang, C. Y. (1970), "Non-linear Analysis of Stress and Strain in Soils," *ASCE Journal of Soil Mechanics and Foundation Engineering*, v. 92, n. 6, pp. 81-104.
- Durukan, Z. and Tezcan, S.S. (1992) "Cost Analysis of Reinforced Soil Walls" *Geotextiles and Geomembranes*, Vol. 11, pp. 29-43.
- Dyer, M.R. (1985), *Observation of the Stress Distribution in Crushed Glass with Application to Soil Reinforcement*, Ph.D. Thesis, University of Oxford.
- Eggebrecht, L.C. (1990), *Interfacing to the IBM Personal Computer*, Second Edition, SAMS Publishing, Carmel, Indiana.

- Ehrlich, M. and Mitchell, J.K. (1994), "Working Stress Design Method for Reinforced Soil Walls," *ASCE Journal of Geotechnical Engineering*, v. 120, n. 4, pp. 625-645.
- Farrag, K., Acar, Y. B, and Juran, I. (1993) "Pull-Out Resistance of Geogrid Reinforcement," *Geotextiles and Geomembranes*, Vol. 12, pp. 133-159.
- Franco, C. (1989), *Caratteristiche Sforzi-Deformazioni-Resistenza Delle Sabbie*, Ph.D. Thesis, Politecnico di Torino, Italy.
- Garret, P.H. (1981), *Analog I/O Design - Acquisition: Conversion: Recovery*, Reston Publishing Co., Reston, Virginia.
- Gourc, J.P., Villard, P., and Matichard, Y. (1992) "Pull-Out Behavior of Reinforcements-Centrifuge Tests and Theoretical Validations," *Earth Reinforcement Practice*, Eds. Ochiai, Hayashi and Otani, Balkema, Vol. 1, pp. 73-78.
- Green, G.E., and Reades, D.W., (1975), "Boundary Conditions, Anisotropy and Sample Shape Effects on the Stress-Strain Behavior of Sands in Triaxial Compression and Plane Strain," *Géotechnique*, v. 25, n. 2, pp.333-356.
- Hall, C. (1981), *Polymer Materials: An Introduction for Technologists and Scientists*, The Macmillan Press, London.
- Handy, R.L. (1985) "The Arch in Soil Arching", *ASCE Journal of Geotechnical Engineering*, v. 111, n. 3, pp. 302-318.
- Hayashi, S., Ochiai, H., Yoshimoto, A., and Kitamura, T. (1988), "Functions and Effects of Reinforcing Materials in Earth Reinforcement," *International Geotechnical Symposium on Theory and Practice of Earth Reinforcement*, Fukuoka, Japan, pp. 99-104.
- Hayden, R.F, Schmertmann, G.R., Qedan, B.Q., and McGuire, M.S. (1991) "High Clay Embankment Over Cannon Creek Constructed with Geogrid Reinforcement," *Geosynthetics' 91 Conference*, Atlanta, pp. 799-822.

- Huesker (1994), *Fortrac® Geogrids: Geogrid Reinforcement for Roads, Walls, Slopes, and Embankments*, 02246/Hue, BuyLine 6579, Huesker Inc., Charlotte, North Carolina.
- Jacquot, R.G. (1995), *Modern Digital Control Systems*, Second Edition, Marcel Dekker, Inc., New York.
- Jarrett, P.M. and McGown, A. (editors) (1988), *The Application of Polymeric Reinforcement in Soil Retaining Structures*, Proceedings of the NATO Advanced Research Workshop on Application of Polymeric Reinforcement in Soil Retaining Structures, NATO Advanced Study Institutes Series, Kluwer Academic Publishers, Holland.
- Jewell, R.A. (1980), *Some Effects of Reinforcement on the Mechanical Behavior of Soils*, Ph.D. Thesis, University of Cambridge.
- Jewell, R.A., Milligan, G.W.E., Sarsby, R.W., and Dubois, D. (1984), "Interaction between Soils and Geogrids," *Polymer Grid Reinforcement*, Thomas Telford Ltd., London, pp. 18-30.
- Jewell, R.A. (1987), "Analysis and Predicted Behavior for the RMC Trial Wall," *The Application of Polymeric Reinforcement in Soil Retaining Structures*, Eds. P.M. Jarrett and A McGown, NATO Advanced Study Institutes Series, Kluwer Academic Publishers, pp. 193-235.
- Juran, I., Knochenmus, G., Acar, Y.B., and Arman, A. (1988) "Pull-Out Response of Geotextiles and Geogrids (Synthesis of Available Experimental Data)," *Proceedings of Symposium on Geotextiles for Soil Improvement*, ASCE, Geotechnical Special Publication 18, pp. 92-111.
- Kharchafi, M. and Dysli, M. (1993) "Study of Soil-Geotextile Interaction by an X-Ray Method," *Geotextiles and Geomembranes*, Vol. 12, pp. 307-219.
- Koerner, R.M. (1993), *Designing with Geosynthetics*, Third Edition, Prantice Hall, Englewood Cliff, New Jersey.

- Koerner, R.M., Hsuan, Y., and Lord, A. E. (1993) "Remaining Technical Barriers to Obtaining General Acceptance of Geosynthetics," *Geotextiles and Geomembranes*, Vol. 12, pp. 1-52.
- Kokkalis, A and Papacharisis, N (1989) "A Simple Laboratory Method to Estimate the In-Soil Behavior of Geotextiles," *Geotextiles and Geomembranes*, Vol. 8, pp. 147-157.
- Kuo, B.C. (1991), *Automatic Control Systems*, Sixth Edition, Prantice Hall, Englewood Cliffs, New Jersey.
- Ladd, C.C., Foott, R., Ishihara, K., Schlosser, F., and Poulos, H.G. (1977), "Stress-Deformation and Strength Characteristics", State-of-the-Art Report for Session I, Proc. 9th ICSMFE, Tokyo, vol. 2, pp. 421-494.
- Lambe, T.W. (1973), "Predictions in Soil Engineering: 13th Rankine Lecture," *Géotechnique*, v. 23, n. 2, pp. 149-202.
- Larson, D.G. (1992), *A Laboratory Investigation of Load-Transfer in Reinforced Soil*, Ph.D. Thesis, Massachusetts Institute of Technology, Cambridge, MA.
- Ling, H.I., Wu, J.T.H., and Tatsuoka, F. (1992) "Short-term Strength and Deformation Characteristics of Geotextiles Under Typical Operating Conditions," *Geotextiles and Geomembranes*, Vol. 11, pp. 185-219.
- Marachi, N.D., Duncan, J.M., Chan, C.K., and Seed, H.B. (1981), "Plane-Strain Testing of Sand," *Laboratory Shear Strength of Soil*, ASTM STP 740, Eds., R. N. Young and F.C. Townsend, American Society of Testing and Materials, pp. 294-302.
- McGown, A., and Andrawes, K.Z. (1977), "The Influence of Non-woven Fabric Inclusions on the Stress-Strain Behavior of a Soil Mass," *C.R. Coll. Int. Sols Textiles*, Paris, pp. 161-165.
- McGown, A., Andrawes, K.Z., and Al-Hasani, M.M. (1978), "Effect of Inclusion Properties on the Behavior of Sand," *Géotechnique*, v. 28, n. 3, pp. 327-346.



- McGown, A., Andrawes, K.Z., and Kabir, M.H. (1982) "Load-Extension Testing of Geotextiles Confined in Soil," *Proc. Second International Conference on Geotextiles*, Las Vegas, pp. 793-798.
- McGown, A., Andrawes, K.Z., and Yeo, K.C. (1985), "The Load-Strain-Time Behavior of Tensar Geogrids," *Polymer Grid Reinforcement*, Thomas Telford Ltd., London, pp. 11-17.
- Monzon, J.E. (1988), "Grounds, Shields, and Power Supply", in *Interfacing Sensors to IBM PC*, Eds. Tompkin, W. and Webster, J., Prentice Hall, Englewood, New Jersey, pp. 35-58.
- Oda, M., Koishikawa, L., and Higuchi, T. (1978), "Experimental Study of Anisotropic Shear Strength of Sand by Plane Strain Test," *Soils and Foundations*, v. 18, n. 1, pp. 25-38.
- Ogorkiewicz, R.M. (1970), *Engineering Properties of Thermoplastics*, Wiley-Interscience, London.
- O'Rourke, T.D., Druschel, S.J., and Nevralati, A.N. (1990), "Shear Strength Characteristics of Sand-Polymer Interfaces", *ASCE Journal of Geotechnical Engineering*, v. 116, n. 3, pp. 451-469.
- Palmeria, E.M. (1987), *The Study of Soil-Reinforcement Interaction by Means of Large Scale Laboratory Tests*, Ph.D. Thesis, University of Oxford.
- Rad, N.S., and Tumay, M.T. (1987), "Factors Affecting Sane Specimen Preparation by Raining," *Geotechnical Testing Journal*, GTJODJ, v. 10, n. 1 pp. 31-37.
- Rimoldi, P. (1988) "A Review of Field Measurements of the Behavior of Geogrid Reinforced Slopes and Walls," *Proceedings of International Geotechnical Symposium on Theory and Practice of Earth Reinforcement*, pp. 571-578.
- Rowe, R.K. and Ho, S.K. (1993), "Keynote Lecture: A Review of the Behavior of Reinforced Soil Walls," *Earth Reinforcement Practice*, Eds. Ochiai, Hayashi and Otani, Balkema, v. 2, pp. 801-830.

- Rumbaugh, (1991), *Object-Oriented Modeling and Design*, Prantice Hall, Englewood Cliff, New Jersey.
- Seah, T.H. (1990), Anisotropy of Resedimented Boston Blue Clay, Ph.D. Thesis, Massachusetts Institute of Technology, Cambridge, MA.
- Sego, D.C., Scott, J.D., Richards, E.A., and Liu, Y. (1990) "Performance of a Geogrid in a Cohesive Soil Test Embankment," *Proceedings of the 4th International Conference on Geotextiles, Geomembranes and Related Products* (Vol. 1), Ed. G.D. Hoedt. Balkema, The Hague, pp. 67-72.
- Sheahan, T.C. (1992), An Experimental Study of the Time-Dependent Undrained Shear Behavior of Resedimented Clay Using Automated Stress Path Triaxial Equipment, Sc.D. Thesis, Massachusetts Institute of Technology, Cambridge, MA.
- Sheahan, T.C. and Germaine, J.T. (1992), "Computer Automation of Conventional Triaxial Equipment", *Geotechnical Testing Journal*, ASTM, v. 15, n. 4, pp. 311-322.
- Sheingold, D.H. (1980), *Transducer Interfacing Handbook - a Guide to Analog Signal Conditioning*, Analog Devices Inc, Norwood, MA.
- Shewbridge, S.E. and Sitar, N. (1989), "Deformation Characteristics of Reinforced Sand in Direct Shear," *Journal of Geotechnical Engineering*, ASCE, v. 115, n. 8, pp. 1134-1147.
- Tatsuoka, F., Sakamoto, M., Kawamura, T., and Fukushima, S. (1986), "Strength and Deformation Characteristics of Sand on Plane Strain Compression at Extremely Low Pressures," *Soils and Foundations*, v. 26, n. 1, pp. 65-84.
- Terzaghi, K. (1936) "Distribution of the Lateral Pressure of Sand on the Timbering of Cuts", *Proceedings of the 1st International Conference on Soil Mechanics and Foundation Engineering*, Cambridge, MA.
- Travers, D. (1984), *Precision Signal Handling and Converter-Microprocessor Interface Techniques*, Instrumentation Society of America, North Carolina.

- Tschebotarioff, G.P. (1951), *Soil Mechanics, Foundations and Earth Structures*, McGraw Hill, New York.
- Van Landingham, H.F. (1985), *Introduction to Digital Control Systems*, MacMillan Publishing Co., New York.
- Van Santvoort, G.P.T.M. (1994), *Geotextiles and Geomembranes in Civil Engineering*, Second Edition, A. A. Balkema, Rotterdam.
- Virk, G.S. (1991), *Digital Computer Control Systems*, MacMillan New Electronic Series, MacMillan Education Ltd., London.
- Whittle, A.J., Larson, D.G., and Abramento, M. (1991), "Annual Technical Report on Geosynthetic Reinforcement of Soil Masses," Research Report, Department of Civil and Environmental Engineering, Massachusetts Institute of Technology, Cambridge, MA.
- Whittle, A.J., Germaine, J.T., Larson, D.G., and Abramento, M. (1992), "Measurements and Interpretation of Reinforcement Stresses in the APSR Cell," *Proceedings of International Symposium on Earth Reinforcement Practice*, Fukuoka, Japan.
- Ward, I. M. (1971), *Mechanical Properties of Solid Polymers*, Wiley Interscience, London.
- Wong, R.K.S. (1985), *Sand Subjected to Cyclic Principal Stress Rotations*, Ph.D. Thesis, University Collage, London.



# Appendix A: APSR Improvements

## A.1. Introduction

The development (design, construction and proof testing) of the APSR cell was described by Larson (1992). This work included preliminary measurements of load-transfer for steel sheet reinforcements embedded in dense Ticino sand. Although, these measurements provided significant new information (Whittle et al., 1992), it was realized that a successful program of testing involving a variety of geosynthetic materials would require modifications and refinements of the original design. Chapter 3 and 4 present data showing the improvement in the quality of the test results due to these design modifications. This appendix presents a more detailed description of the changes made in the reinforcement control system.

## A.2. Reinforcement Positioning System

The key design feature of the APSR cell is that the reinforcement is clamped externally to a load cell such that the plane through which the inclusion enters the cell is also a plane of symmetry in the specimen. To maintain symmetry, the position of the reinforcement is controlled by an actuator such that there is no displacement of the reinforcing inclusion at the reference entry point. In the original design, tensile forces in the reinforcements were adjusted using an analog proportional control based on displacements monitored by proximity sensor at the exit point of the cell. Although

Larson (1992) describes several refinements of this referencing system, he concludes that the position control is prone to malfunctions.

Figure A.1 shows the details of the inclusion position control system used by Larson (1992). The reinforcement was attached to an external support arch through a system of yoke linkage where it was loaded by an hydraulic jack. The system was made up of many components with inevitable compliance between them and hence, was lacking the rigidity necessary for precise control of reinforcement position. It was, therefore, decided to replace the hydraulic piston by a linear actuator assembly (ball screw actuator). The new design, described in Chapter 2, uses a light-weight, self-supported linear actuator and greatly reduces the compliance of the grip/linkage system. Another important improvement was achieved by replacing the original analog proportional control system with a fully digital PID control.

The above modifications allowed a better control of the reinforcement position but did little to improve the ability to measure the reinforcement position at the exit point. In a feedback system, under the best possible situation, control of a parameter can only be as good as measurement of that parameter. Larson (1992) described several revisions of the position detection system. Figure A.2.a shows the final version of the mechanism used by Larson (1992) in which a proximity sensor is mounted rigidly in a plexiglas flap. One end of the flap is secured to the rear wall of the cell through a hinge while the other end is attached to the reinforcement using a flexible linkage. The longitudinal movement of the inclusion causes rotation of the flap which, in turn, changes the distance between the sensor and a metallic target. The target is attached to a small steel square that rests inside the soil specimen and hence, allows the inclusion position to be referenced directly to the rear surface of the specimen. This system was further modified during the present test program as shown in Figure A.2.b

to increase the rigidity of the flap-inclusion connection and hence reduce sensitivity of the system to out-of-plane movements of the inclusion.

However, the externally measured inclusion loads remained lower than expected even when the exit point position, as measured by the system described above, was controlled reasonably well. The problem was finally resolved when movements of the rear wall of the cell in the vicinity of the exit point were measured using a displacement transducer (LVDT) attached to an independent reference frame. Figure A.3a shows that as stresses are applied to the specimen, points along the centerline of the rear wall move inwards by a very small but significant amount. As the specimen is loaded, even though the inner plane strain walls remain stationary, the outer plane strain walls show significant bulging due to water pressure within the sidewall assembly. The top and bottom sidewalls are attached rigidly to the front and rear cell walls and therefore, outward (convex) bending of the sidewalls causes inward (concave) deflection of the rear and front perimeter walls. Since the proximity sensor flap is attached to the rear wall of the cell near the centerline, the measurement system underestimates the true inclusion movements because of the rear wall bending. In order to find out possible effects of errors in the measurements (and hence control) of inclusion position, a 'pull out' test was performed on a 270 mm long, steel inclusion at the applied stress ratio,  $R = 6.4$ . The shearing of the soil specimen was stopped and the inclusion was pulled out of the cell at a sufficiently low rate to allow measurements of inclusion position and loads. Figure A.3.b shows that a position error of the order

of several micrometers can alter inclusion loads significantly<sup>1</sup> especially for a relatively stiff reinforcing material such as steel.

These findings emphasized the importance of choosing the correct frame of reference for active control of the reinforcement. The infrared sensor assembly described in Chapter 2 was designed such that the inclusion position was measured with respect to the rigid aluminum frame which supports the APSR cell.

---

<sup>1</sup> The maximum expected load (at  $R = 9$ ) in the steel sheet inclusion with  $L/2 = 270$  mm is 25 to 30 kN/m.



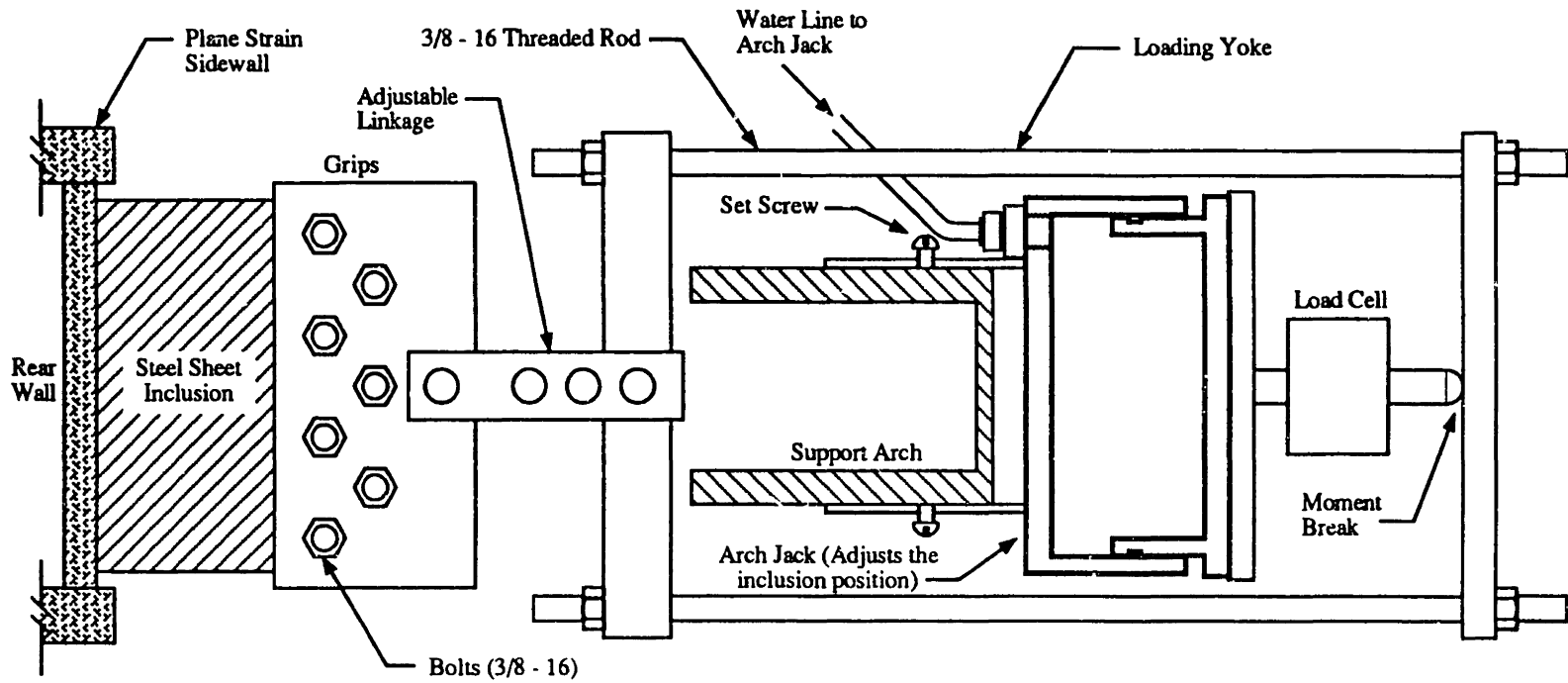
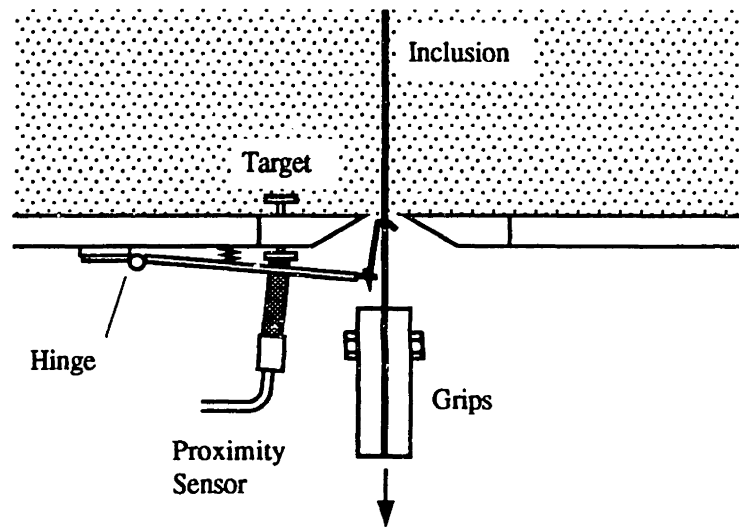
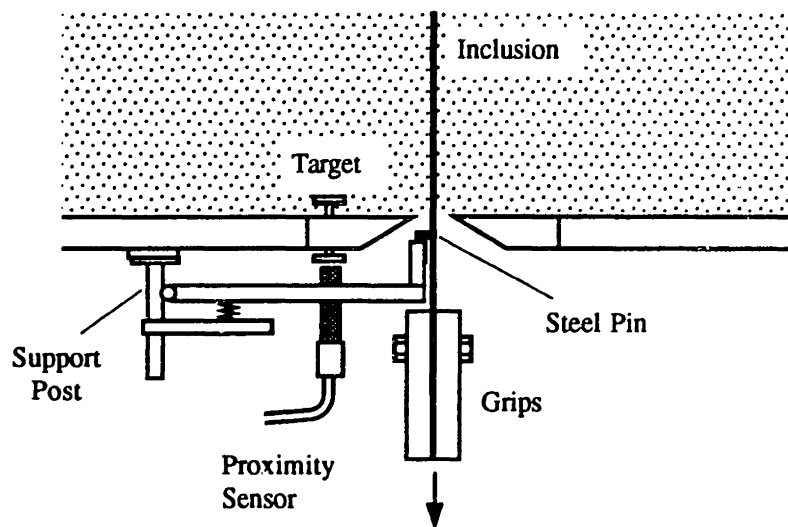


Figure A.1: Connection from Arch to Reinforcement Grips (after Larson, 1992).

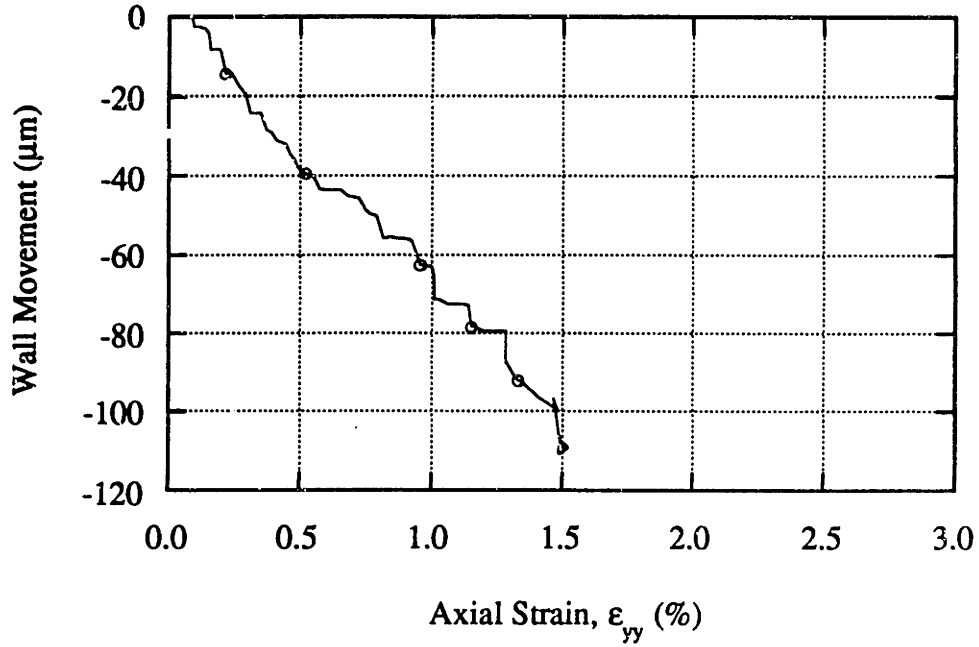


(a) Reinforcement Position Measurement Using Proximity Sensor, System 1.

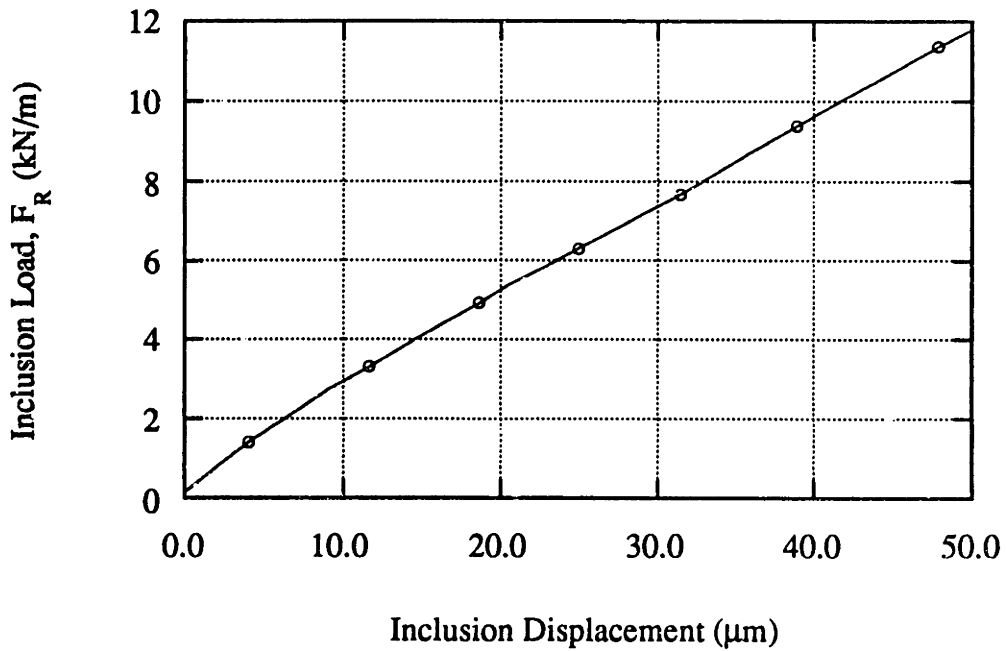


(b) Reinforcement Position Measurement Using Proximity Sensor, System 2.

Figure A.2: The Original Designs for Referencing the Inclusion Position.



(a) Bending of the Rear Wall due to Internal Stresses within the APSR Cell.



(b) Exit Point Load as a Function of Displacement of the Steel Sheet Inclusion.

Figure A.3: Data Showing Limitations of the Original Inclusion Referencing System.



# Appendix B: Digital Feedback Control

## B.1. Introduction

The APSR cell uses digital feedback control to maintain specified boundary conditions during the test. Chapter 2 (Section 2.4.1) provided a few basic ideas and terminology related to control system engineering. This appendix extends that discussion and provides references to advanced topics in digital control. The Section B.2 discusses important performance criteria and tradeoffs for designing feedback control systems. Derivations of direct digital control (DDC) algorithms used in the present research and guidelines for "tuning" a feedback loop to achieve optimum performance are presented in Section B.3.

Control system is often represented using block diagrams which consist of a set of blocks connected by lines representing the signal paths. The blocks retain only the essential characteristics of the components they represents. Each block receives an input signal from some part of the system and produces an output signal for the other part of the system (Figure B.1a). The most important characteristic of a component is the relationship between the input signal and the output signal. The relationship is expressed by the transfer function,  $G$ , which is defined as the ratio of the output signal divided by the input signal. Gain of the transfer function is the change in output amplitude for a unit change in the input amplitude. Thus, a DC motor that produces a change in speed of 1000 revolutions per minute for each 1V change in input has a gain of 1000 rpm/volt.

Feedback is the action of measuring the difference between the actual result and the desired result and using that difference to drive the actual result toward the desired result. Figure B.1b shows block diagram of a simple feedback (closed loop) control system. The reference value,  $R$ , is the input to the controller. The forward path consists of three components: a) the controller, b) the manipulating (servo) element, and c) the plant (or process) to be controlled. The combined transfer function of the forward path is the product of the three component transfer functions:

$$G = G_c G_m G_p \quad (\text{B.1})$$

The feedback is usually obtained through a passive device (transducer) with a transfer function,  $H$ , which converts the controlled variable,  $C$ , into a suitable signal,  $C_m$ . The output of the controller is usually based on a comparison between the reference,  $R$ , and the measured value,  $C_m$ . Using simple rules of block diagram reduction (Kuo, 1991) the overall transfer function,  $C_m/R$ , of the feedback system in Figure B.1b can be derived as:

$$\frac{C_m}{R} = \frac{GH}{1 + GH} \quad (\text{B.2})$$

It is possible to obtain mathematical expressions describing the response of the feedback system (i.e., changes in  $C_m$  for a given change in the reference value,  $R$ ) provided the individual transfer functions,  $G_c$ ,  $G_m$ , and  $G_p$  are known. The purpose of control system design is the following: given  $G_m$ ,  $G_p$ , and  $H$ , obtain controller transfer function,  $G_c$  (i.e., the relationship between the error signal,  $E$ , and control action,  $Y$ ) such that the overall system is stable and faithfully follows the reference signal,  $R$ , all time regardless of external disturbances. Section B.3 presents some commonly used controller transfer functions also called control algorithms. The next

sections describes primary objectives of a control system and defines some commonly used control system performance criteria.

## B.2. Control Objectives and Performance Criteria

Since the objective of any control system is to maintain the controlled variable exactly equal to the reference value at all time regardless of changes in the environment or the reference value, the system must respond to a change before error occurs. However, this is impossible in the case of a feedback system since the controller requires a finite amount of error in order to produce changes in the manipulated variable. It is, therefore, reasonable to judge the performance of a control system based on how closely it resembles an ideal system. Since the errors in a control system occur after a change in the reference value, it is natural to define performance in response to such changes. The response of a control system to a step change in the reference value is widely used to evaluate control systems although it is possible to quantify performance of a control system using sinusoidal or ramp reference functions (Bateson, 1993).

Figure B.2 shows a typical response of a control system to a step change in the reference value.<sup>1</sup> The first objective of a control system is to minimize the maximum value of the error signal since a large transient errors can have detrimental effects. Eventually, the control system should return the error to a steady (non-changing) value.

---

<sup>1</sup> This discussion assumes that the system is stable, that is, a bounded (finite) input results into a bounded output.

The time required to accomplish this is called the settling time. A second objective of a control system is to minimize the settling time. The third objective is to minimize the steady state (residual) error after settling out. Unfortunately, these three objectives tend to be mutually incompatible. For example, the steady state error can be usually reduced by increasing the gain of the controller within a limit. However, an increase in gain tends to increase the settling time and may increase the maximum value of transient error. The optimum performance requires a compromise between various objectives. It is therefore useful to quantify various features of the step response shown in Figure B.2.

The *maximum overshoot* ( $M$ ) is the maximum deviation of the controlled variable above its steady state value  $x_{ss}$ . It is sometimes expressed as a percentage of the final value. Normally, the maximum overshoot increases with decreasing damping in the system and hence, it is sometimes used as an indicator of the relative stability of the system. Some control systems eventually reduce the error to zero while others require a residual error called the *steady state error*. The *peak time* ( $t_p$ ) is the time at which the maximum overshoot occurs. The *settling time* ( $t_s$ ) is the time required for the oscillations to stay within some specified small percentage value of the final value. The most common values used are 2% and 5%. The *rise time* ( $t_r$ ) is usually defined as the time required to rise from 10% to 90% of its final value although other definitions of rise time exist. Finally, the *delay time*,  $t_d$ , is the time required for the output to reach 50% of its final value. For quasi-static geotechnical tests, such as the APSR test, the characteristic times ( $t_p$ ,  $t_r$ , etc.) are of the order of several minutes and hence, are not very critical. The steady state error, on the other hand, can be very important whenever a high degree of long-term accuracy is desired.



## B.3. Control Algorithms

### B.3.1. Proportional Control

The most important characteristic of a controller is the way it uses the error signal to form the control action. The simplest feedback control algorithm is to generate a control action that is proportional to the difference between the reference value and controlled output.

$$Y = K_P (R - C_m) = K_P E \quad (\text{B.3})$$

Thus, the transfer function of the proportional controller,  $G_c = K_P$ . One problem with the proportional control is that it cannot completely eliminate the error caused by a change in the reference value. A change in the reference signal means a control action is required which can only be generated if there is an error in the system. Since most real actuator devices stop responding when the control action falls below a finite threshold, a residual error called the *proportional offset* remains.

### B.3.2. Integral Control

The integral control mode changes the output of the controller by an amount proportional to the integral of the error signal. As long as there is an error, the integral control will keep accumulating the error and produce an output such that:

$$Y = K_I \int_0^t E dt \quad (\text{B.4})$$

The integral control is frequently combined with the proportional control to provide a PI controller that eliminates the proportional offset. One problem with the integral mode is that it increases the tendency for oscillation and reduces the ability of the controller to

respond to large changes in the reference value. If the process has a large dead-time lag (such as due to play in a ball screw actuator), the error signal will not immediately reflect the actual error in the process. This delay often results in overcorrection; that is, the integral controller continues to change the controller output after the error is actually reduced to zero because it is acting on an "old" error. This is sometimes referred to as the *integral windup*.

### B.3.3. Derivative Control

One problem with the proportional feedback control is that to avoid problems with severe overshooting, the gain  $K_P$  is limited and this in turn limits the rise time. The integral control is inherently slow in response. In order to achieve a more rapid response without inducing oscillations, the output of the controller is made proportional to the rate of change of the error signal:

$$Y = K_D \frac{dE}{dt} \quad (\text{B.5})$$

This form of control is not very useful by itself but is generally combined with proportional and integral control to improve the performance of the controller.

### B.3.4. Direct Digital Control

We can combine all three forms of feedback control to form what is generally referred to as a PID controller. This combination reduces the steady state error to zero and often yield satisfactory dynamic response. The equation of an ideal PID controller is:

$$Y = K_P E + K_I \int_0^t E dt + K_D \frac{dE}{dt} \quad (\text{B.6})$$

Equation B.6 is applicable to continuous-time (analog) control systems since derivation and integration are only defined for continuous systems. With direct digital control (DDC) used in the APSR cell, the control action must be realized in the computer. It is possible to obtain a digital algorithm that will be similar in character to the continuous-time scheme given above. It is common to approximate the integral with a simple summation and the derivative with a backward difference equation:

$$Y_n = K_P E_n + K_I T_s [E_0 + E_1 + \dots + E_n] + K_D/T_s [E_n - E_{n-1}] \quad (\text{B.7})$$

where  $T_s$  is the sampling time and subscripts  $n, n-1$  denote values at time  $T, T-T_s$ , etc. When the sampling interval is fixed,  $T_s$  can be incorporated into the values of loop constants such that:

$$Y_n = K_P E_n + K_I \sum_{i=0}^{i=n} E_i + K_D [E_n - E_{n-1}] \quad (\text{B.8})$$

The above equation is also called 'position control' PID algorithm since the output signal,  $Y_n$ , of the algorithm is a direct measure of the position of the actuator. It can be seen that Equation B.8 implies that in the steady state, with zero error, the controller output is zero. This may not be the case for many devices such as electro-pneumatic pressure regulators which require certain constant voltage to maintain a constant target pressure. For this type of devices, the integral term 'builds up' and provides the steady state output under zero error condition. However, one problem that often occurs in using Equation B.8 for plants which require large steady state output is that, unless the process is already pre-set to the nominal conditions so that the start-up error is small, there will be a transient 'bump' whenever there is a large, abrupt change in the reference signal.

This problem can be eliminated by using the so called 'velocity' form of Equation B.8 (Bennett, 1988). In the velocity control algorithm, the output signal of

the controller is a measure of the adjustment of the actuator with regard to the previous position. The incremental term  $\Delta Y$  is computed such that:

$$Y_n = Y_{n-1} + \Delta Y \quad (\text{B.9})$$

Combining Equation B.9 with B.8 yields:

$$\Delta Y = K_P(E_n - E_{n-1}) + K_I E_n + K_D(E_n - 2E_{n-1} + E_{n-2}) \quad (\text{B.10})$$

A smooth start-up condition is ensured by using a reasonable initial value of the controller output,  $Y_0$ , in Equation B.10 which can be estimated from the open loop response of the system (refer to Section 2.4.4).

### B.3.5. Tuning Procedures

The PID algorithm presented above is simple enough to provide some general "tuning" rules without exact knowledge of transfer functions of the actuator and plant and yet sufficiently effective to achieve a reasonable control. These qualities have made PID control very popular for process control applications. The classic tuning procedure, known as the Ziegler-Nichols method (Van Landingham, 1985), is to increase the gain,  $K_P$  (with  $K_I = K_D = 0$ ) until a sustained oscillation occurs in the measured variable with  $K_P = K_{\max}$ . Then, if the period of the oscillation is  $T_0$  seconds, the PID parameters are:

$$K_P = 0.6K_{\max} \quad (\text{B.11a})$$

$$K_I \leq 2K_P/T_0 \quad (\text{B.11.b})$$

$$K_D \geq 0.125 K_P T_0 \quad (\text{B.11.c})$$

If only PI control is used,

$$K_P = 0.45K_{\max}; K_I \leq 2K_P/T_o \quad (\text{B.12})$$

For proportional control only:

$$K_P = 0.5K_{\max} \quad (\text{B.12})$$

There are tuning methods that do not require sustained oscillations which can be harmful in certain cases. These methods rely on experimental evaluation of two parameters obtained from open loop response of the plant to a step function of magnitude  $\Delta$ , shown in Figure B.3 (Virk, 1991; Jacquot, 1995). The first parameter is the slope of the response curve at inflection point R, which is an indication of the speed of the response. The second is the time L, which is measure of the lag of the plant.

For PID control:

$$K_P = \frac{1.2\Delta}{RL}; K_I = \frac{0.5K_P}{L}; K_D = 0.5LK_P \quad (\text{B.13})$$

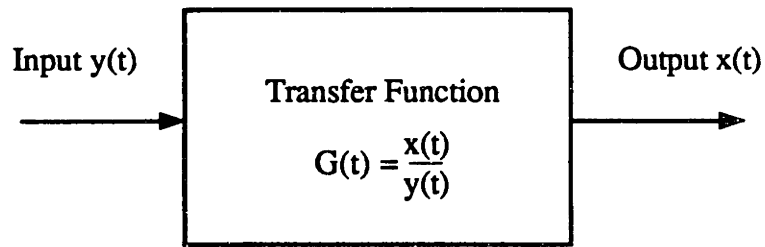
If only PI control is used then:

$$K_P = \frac{0.9\Delta}{RL}; K_I = \frac{0.3K_P}{L} \quad (\text{B.14})$$

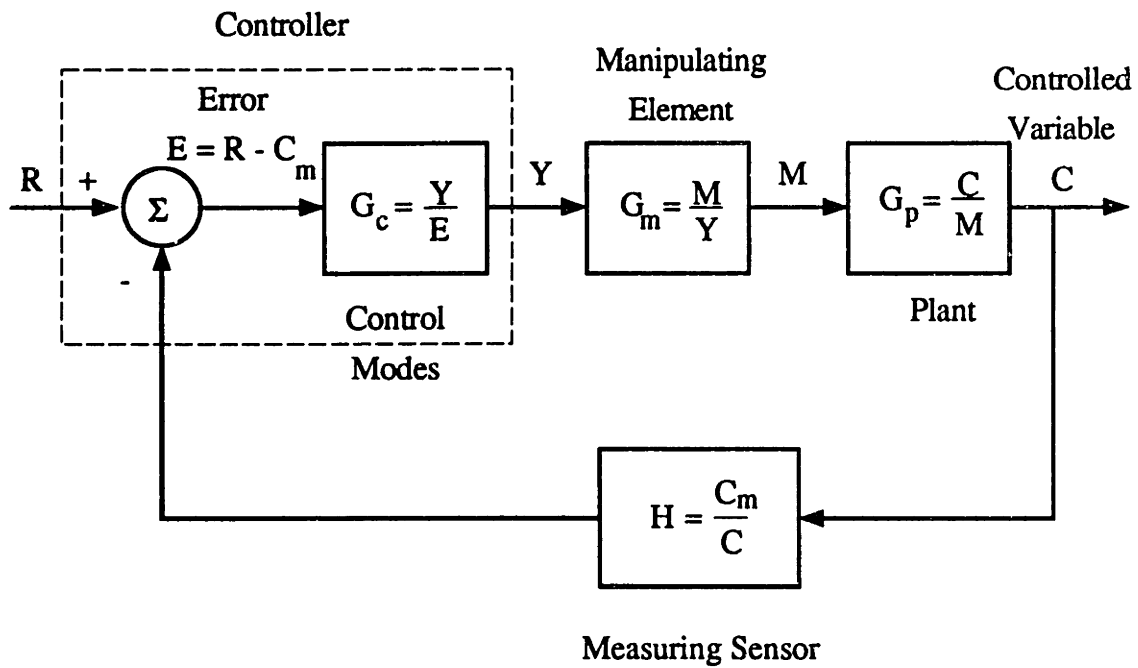
For proportional control only:

$$K_P = \frac{\Delta}{RL} \quad (\text{B.15})$$





(a) Transfer Function of a Simple Block Unit.



(b) Block Diagram of a Closed-Loop Control System.

Figure B.1: Block Diagram Representation of Control Systems.

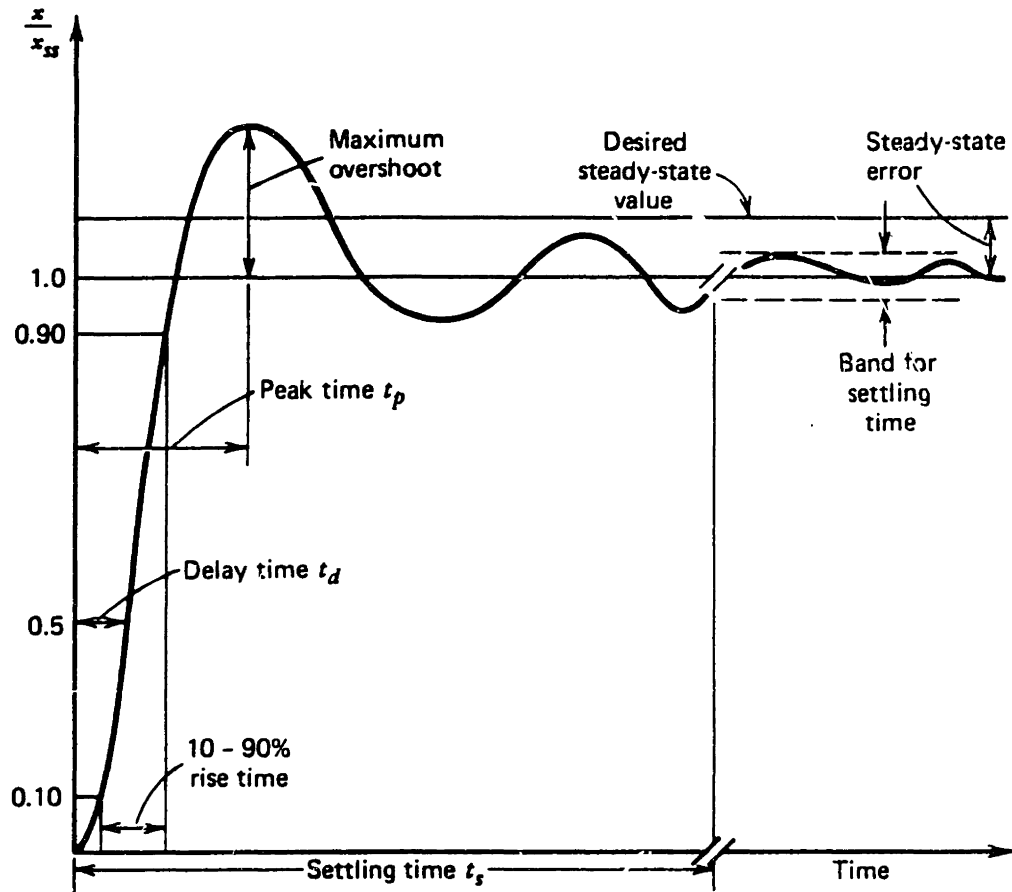


Figure B.2: Step Response of a Typical Feedback Control System.



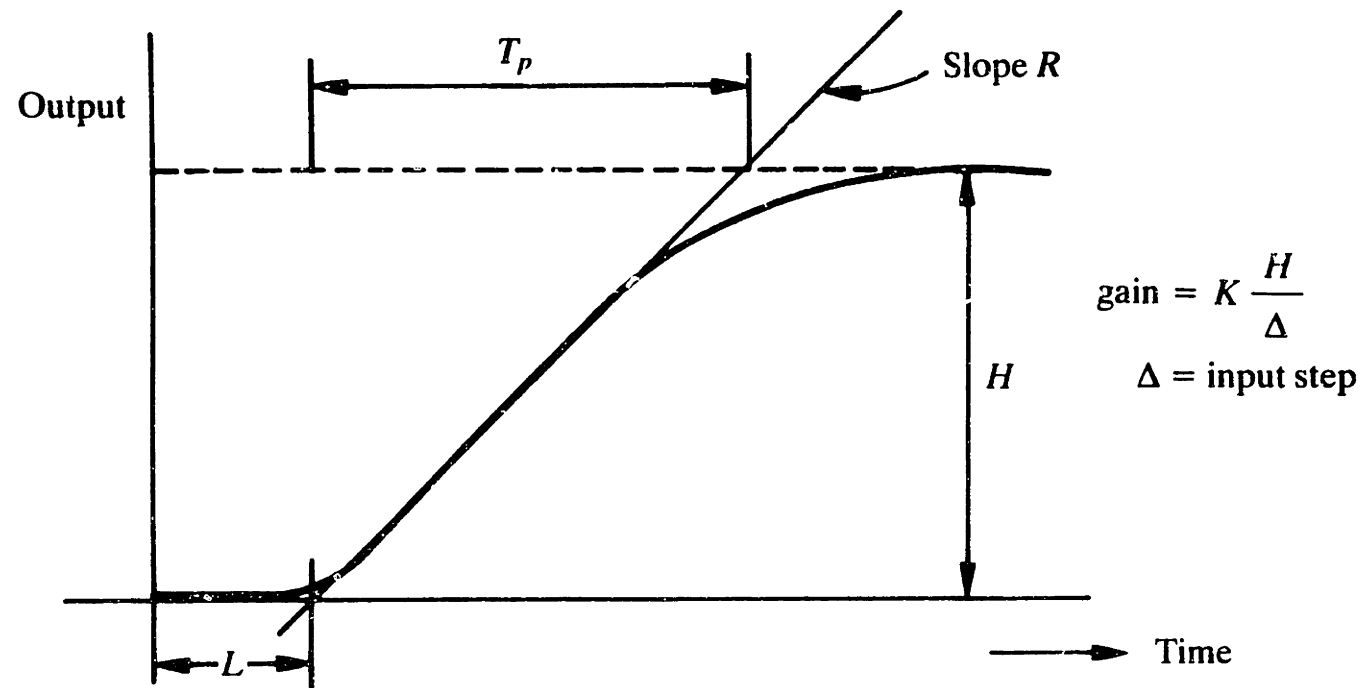


Figure B.3: Open Loop Response to a Step Function.



# Appendix C: Data Converter Cards

## C.1. Introduction

The feedback control of a dynamic system involves collecting information about a set of output (controlled) variables. This information is then manipulated to arrive at values of input (manipulated) variables such that the system under consideration follows a desired trajectory in time. With the recent availability of powerful, low cost microprocessors, it is becoming increasingly advantageous to carry out information acquisition, processing, and transmission functions using digital methods in almost every branch of engineering. In general, a digital system offers higher noise immunity, flexibility, and reliability compared to a functionally equivalent analog system. However, in control engineering, output from (and input to) the outside world is typically in analog form. An analog signal varies in a continuous manner and may take any value between its limits. The digital computer, on the other hand, is capable only of dealing with discrete numbers and not continuous signals. If a digital computer is to accomplish a control task, two additional hardware devices are needed to interface the analog and digital domains. The first of these devices is the analog-to-digital converter (ADC), which samples the output signal periodically and converts these samples to binary numbers to be processed by the computer, which generates a control strategy in the form of a number. The second device is a digital-to-analog converter (DAC), which converts the numerical control strategy generated by the computer in digital form to an analog signal. The impressive advances in the microprocessors have been matched by rapid developments in the data converter technology.

This appendix presents detailed description two custom-designed data converter cards developed as part of the APSR project. It also provides some basic concepts in data conversion and definitions of some commonly used terms.

## C.2. Analog-to-Digital Conversion

Analog-to-digital conversion is essentially a ratioing operation. The input analog signal  $V_i$  is converted into a fraction by comparing it against a reference signal  $V_r$  such that

$$X = V_i / V_r \quad (C.1)$$

The digital output of the converter is a binary coded representation of  $X$ . If the converter output consists of  $N$  bits, the number of discrete outputs is fixed at  $2^N$ . For one-to-one correspondence, the input range must be quantified into this same number of levels. Each level (quantum) is the analog value by which two adjacent codes differ. It is also called the least-significant bit (LSB) size. Thus:

$$Q = \text{LSB} = \text{FS} / 2^N \quad (C.2)$$

where  $\text{FS}$  is the full-scale range of the converter.

Figure C.1 shows the transfer function (relationship between the input and output) of an ideal 3 bit converter with the analog levels on the horizontal axis and the digital outputs which correspond to those input values on the vertical axis. All analog values within a given quantum are represented by the same digital code. Thus, the A/D conversion process is affected by an inherent quantification uncertainty of  $\pm 1/2$  LSB whose effect can only be minimized by increasing the number of bits in the converter

output code. Bit resolution is, therefore, the most important characteristic of an analog-to-digital converter. The quantification error ranges from 0.19% for 8-bit resolution to 0.00076% for 16-bit resolution.

Other important A/D converter specifications are: a) accuracy, b) conversion rate, c) thermal drift, and d) input impedance. Accuracy incorporates factors such as offset error, gain error, and linearity. The dotted line in Figure C.1 represents the ideal transfer function with an infinite resolution. It passes through the origin and has 1:1 slope. Gain error is a change in the slope of this line from the ideal slope shown in the figure. The gain error affects all levels of input by the same percentage amount. Offset error is a displacement, left or right, of the infinite resolution line with no change in slope. Offset error becomes more significant as input signal approaches zero. Linearity error is a deviation of the infinite resolution line from a straight line. The gain and offset errors can be easily corrected by hardware or software schemes but linearity error is very difficult to compensate. The conversion rate, defined as the number of times the input signal can be sampled in unit time, is another important parameter. There is always an inherent trade-off between bit resolution and the time required for analog-to-digital conversion. The conversion rate should be at least twice the highest frequency component of an input signal. For typical geotechnical laboratory applications, conversion rate of 10 Hz is more than adequate. The changes in the offset and gain errors with temperature are referred to as thermal drift which is specified in ppm/°C. Normally, a certain amount of warm-up time is required before a converter achieves its stable operating condition. The process of measuring voltage in any circuit draws some amount of current from it and hence alters the voltage being measured. Input impedance, expressed in ohms, is a measure of source loading by any electronic device. The higher the impedance, the smaller the amount of current drawn by the converter and the smaller the error due to source loading.

The most common analog-to-digital conversion techniques implemented in monolithic or hybrid integrated circuits are: a) parallel or flash conversion, b) successive approximation, c) dual and multiple slope integration, d) voltage-to-frequency conversion, and e) over sampling or Sigma-Delta technique. The converter chip used in the present research program (AD1170) is based on the voltage-to-frequency conversion method described briefly in Section C.2.1.1. For a complete discussion on this and other analog-to-digital conversion techniques, the reader should refer to Garret (1981), Travers (1984), Derenzo (1990), Carr (1991), and Analog Devices (1992).

The AD1170 was chosen, in part, because it was used in an earlier effort to build a high resolution A/D converter card at MIT Geotechnical laboratory. The Multichannel Analog-To-Digital Conversion (MADC) board, built and described by Sheahan (1992), offers only eight input channels due to an unfortunate decision to use one instrumentation amplifier per channel. Since the APSR setup required at least 25 analog input channels, a decision was made to design a new card which would double the channel density to 16 channels per card. This was accomplished by: a) using two analog multiplexers, one each for the high and low side of a differential input, and b) using only one instrumentation amplifier following the multiplexers instead of 16 amplifiers placed before the multiplexers as would have been required by the old design. The next section describes the architecture of the AIC-16 Analog Input Converter card in detail.

### C.2.1. AIC-16 Architecture

Figure C.2 shows the block diagram of the AIC-16 analog input card. The card fits into one of the several expansion (bus) slots provided on the motherboard of an IBM compatible computer. In addition to the AD1170 converter chip, the AIC-16 card

contains a number of integrated circuits (ICs) and discrete components. The card can be divided into three major functional blocks: a) the AD1170 chip, b) the analog section, and c) the digital section. The AD1170, manufactured by Analog Devices, is a high resolution, integrating A/D converter intended for applications that require the highest accuracy and stability, but relatively low conversion rate. It is a hybrid device packaged as a 40 pin, sealed, non-serviceable module. The analog section processes the analog input received from the outside world before feeding it to the converter. The digital section is mainly comprised of TTL (Transistor-Transistor Logic) integrated circuits which provide interface between the card and the computer's bus.

#### C.2.1.1. AD1170 Chip

The AD1170 is a complete microcomputer-based measurement subsystem composed of three major elements: a highly precise charge balancing converter, a single chip microcomputer, and a custom controller chip (Figure C.3a). Charge balancing (Figure C.3b) is the most common voltage-to-frequency conversion technique. A current of opposite polarity is added to the analog input signal and the result is integrated by a free running integrator in a feedback loop. The feedback loop continually seeks to null the input of the integrator by subtracting precisely determined packets of charge when the accumulated charge exceeds a reference value. The number of charge packets per second (or frequency) required to balance a given input is proportional to the input. A digital counter is used to convert the resulting serial pulse train to a binary word. The integration process tends to filter out high frequency noise. In particular, if the integration time is a multiple of 50/60 Hz AC line frequency, such noise is completely eliminated. Integrating type converters are therefore ideal for applications requiring accurate conversion of low frequency signals.

The AD1170 offers independently programmable integration time and data format from 7 to 22 bits. It is possible to achieve conversion rates as high as 50 conversions per second with a very good noise rejection. The converter is fully self-calibrating and exhibits total non-linearity of 0.001% and thermal drift of only 9ppm/°C assuring accurate, stable readings. Table C.1 lists the important specifications of the device. For maximum stability, the AD1170 periodically calibrates itself by performing measurements against internal references. Calibration cycles may be programmed to take place whenever the AD1170 is idle, or may be invoked under system control. In addition, the chip offers an electronic (digital) calibration feature which eliminates the system level offset and gain errors without manual intervention. The calibration data can be stored in an internal nonvolatile memory chip. The AD1170 can be interfaced to a personal computer via an 8-bit data bus and its advanced features are controlled by simple software commands sent to it via this bus (refer to Section C.2.2.3). For an in-depth discussion of these and other features of the AD1170 chip, the reader is referred to Analog Devices (1992).

#### C.2.1.2. Analog Section

The analog section contains: a) two analog multiplexers, b) an instrumentation amplifier, and c) a DC-DC step-up converter. The analog multiplexer is a type of solid state electronic switch which selects one of a number of inputs depending on the state of its digital control lines and routes the selected input to a single output terminal. The AIC-16 card uses ADG526A monolithic, CMOS<sup>1</sup>, latched multiplexer from Analog Devices. The ADG526A switches one of 16 inputs to a common output depending on

---

<sup>1</sup> The Complimentary Metal-Oxide Semiconductor fabrication process produces a switch whose 'on' resistance remains relatively constant with the changes in temperature and input current.



the state of four binary address lines. A *latch* on-board the multiplexer captures the state of the address lines under an explicit computer command and holds it irrespective of any subsequent changes in the address bits. Since the ADG526A can switch only one input line at a time, two multiplexers are required to switch the double-ended (differential) input of the AIC-16 card. The address pins of both multiplexers are driven from a common source to ensure that they are in a consistent state at any given time. The 'break-before-make' feature of the multiplexer ensures that two input lines are never shorted together. The ADG526A has about  $300\Omega$  resistance in 'on' state which means that even a small current passing through the device would cause a voltage drop and hence distort the input signal. This problem can be solved by providing a very high input impedance ( $> 100\text{ M}\Omega$ ) device between the multiplexer and the converter to prevent any current flow through the former. The AD524 precision instrumentation amplifier manufactured by Analog Devices is one such device.

An instrumentation amplifier is a precision differential gain device with a high input impedance and low output impedance. The role of AD524 instrument amplifier in the AIC-16 card is to: a) minimize the current through the multiplexers and b) convert a differential signal into a ground referenced single-ended signal and feed it to the AD1170 converter chip. The AD524 has a high input impedance ( $1000\text{ M}\Omega$ ), low non-linearity (0.003%), and high common mode rejection ability. Common-mode rejection, the property of canceling out any signals which are common (the same potential on both inputs) while amplifying any signals that are differential is the most important property of an instrumentation amplifier. Although the gain (the ratio of output to input) of the AD524 may be set as high as 1000, the AIC-16 board uses the amplifier in the unity gain configuration.

The multiplexers, instrumentation amplifier, and A/D converter chip require  $\pm 15$  V DC power supply. The AD949 DC-DC converter from Analog Devices converts +5 volt supply tapped from the computer's bus into regulated  $\pm 15$  volts. It is capable of supplying up to 60 mA of ripple free current which is more than sufficient power for all the devices connected to it.

### C.2.1.3 Digital Section

The computer is able to send commands to the AD1170 converter (as well as the multiplexers) and retrieve converted digital data from it over the PC system bus. A bus is a set of electrical lines connected simultaneously to a number of devices which use these lines to communicate with each other in a coordinated manner. Every IBM compatible PC has anywhere from 3 to 6 system bus slot on the back of the motherboard. They are used for connecting peripheral devices such as display adapters, disk controllers, printers, memory boards, etc. to the computer. Figure C.4 shows the 62-pin assignments of the original IBM PC-XT system bus. In addition to 8 data lines (D0 to D7) and 20 address lines (A0 to A19), it has many signal lines, and power and ground pins. The later versions of PC's (AT and higher) have buses with 16 or even 32 data lines to achieve higher data transfer rates. However, since any card designed for the PC-XT bus can be used with these more advanced buses, both AIC-16 and AOC-8 (see Section C.3) cards were designed for the XT bus.

Every device connected to the system bus of a PC is identified by a unique address. It is possible to either send (write) a byte (a set of 8 bits) to a device or receive (read) a byte from it using IN and OUT instructions of the Intel 8088 microprocessor. The address decoding circuits contained in the digital section of the card continuously monitor the bus address lines for a particular binary combination called the *base address* of the card. When the base address is detected, interface circuits on board the card

capture the state of the data lines if a write operation is underway. If it is a read operation, the interface circuits release a byte on bus data lines which is captured by the microprocessor in the next cycle. Whether it is a read or a write operation is determined by examining the state of two special control lines ( $\overline{\text{IOW}}$  and  $\overline{\text{IOR}}$ ). For more information on the IBM bus and how to interface with it, the reader is referred to An et al. (1988) and Eggebrecht (1992).

### C.2.2. Operation of AIC-16 Card

This section describes how to install the AIC-16 card and communicate with it in a high level language (such as BASIC and C). The base address of the AIC-16 card is set by changing the positions of eight DIP switches located on the card. The card decodes the 12 lower address lines of the PC system bus (A0 to A11) making it possible to set the base address anywhere from 0 to 4095 ( $2^{12} - 1$ ). However, in the IBM PC architecture, certain base addresses are reserved for specific purposes and are not available for general use. IBM recommends the base address of 768 decimal (1100000000 binary) for all peripheral devices. This base address corresponds to DIP switch combination 00110000, where 1 and 0 represent the on and off states respectively. Since the IBM PC specifications use only 10 address lines, the extended decoding scheme (i.e., the use of 12 lines) provides three additional addresses (1792, 2816, 3840) whose lowest 10 bits are same as decimal 768. The computer is unable to distinguish between these four addresses, however, they can be used to differentiate two or more AIC-16 cards installed in the same computer. After setting a proper base address, the card can be inserted into any empty expansion (bus) slot.<sup>2</sup> The analog inputs from transducers and other sources are fed to the AIC-16 card via a 36 pin D-sub

---

<sup>2</sup> The card should never be inserted into or extracted from a slot while the computer is on.

connector located at the rear end of the card. The channels are arranged such that each channel takes up two consecutive pins beginning with pin 1 of the connector. The odd numbered pins are always positive (i.e., higher polarity). The differential inputs are ground referenced internally and hence there is no need to connect the transducer's power supply ground to the computer's ground. However, the transducer cable should be properly shielded to protect the signal from electromagnetic and radio frequency noise. For more information on shielding and grounding practices the reader is referred to Sheingold (1980) and Monzon (1988).

Most higher level computer languages provide equivalent of Intel 8088 assembly language IN and OUT instructions for performing I/O operations. In BASIC, statement `OUT n, i` sends an eight bit integer `i` to decimal address `n`. Similarly, statement `j = INP n` reads a byte from address `n` and assigns it to integer variable `j`. In C language, standard library functions `outportb(int n, int x)` and `inportb(int n)` accomplish the same tasks. Each AIC-16 card occupies its base address plus next seven locations (also called ports) in the computer's I/O address space. Out of the total of eight ports, the lower four ports belong to the AD1170 chip while the multiplexers occupy the fifth location (i.e., base address + 4). All other locations are undefined and writing to or reading from these addresses does nothing. The sixteen analog input channels are numbered from 0 to 15. Executing a single BASIC statement:

`OUT (address + 4), 10`

switches the multiplexers and selects input channel 10.

#### C.2.2.1. Communicating with AD1170

As mentioned earlier, the four successive bytes of read/write address space, beginning with the base address of the card, are used for communicating with the

AD1170 chip. Table C.2 lists the functions of these four address locations (also called registers). A special byte, always available from the lowest register (i.e., base address), is called *status byte* and is very useful in coordinating I/O activity. The status byte contains six bits of information about the current status of the AD1170. Table C.3 shows the interpretation of various bits in the status byte. When low, the BUSY bit (bit 0) indicates that the AD1170 is ready to receive a command. Since any command sent while the BUSY bit is high is ignored by the converter, it is necessary to examine the status of the BUSY bit between two consecutive commands sent to the converter. In BASIC, this is most efficiently done by the statement:

WAIT i, j, n

This statement suspends the execution until the byte read at the port address n has a specified bit pattern which is related to integers i and j. The statement WAIT 1, 1 will halt the execution of a program until the BUSY bit is cleared. The WAIT statement of BASIC can be simulated in C language by a function containing a 'while' loop. The reader should refer to the FlexCAT code given in Appendix D for an example of such a function.

Table C.4 lists the most commonly used AD1170 commands, their binary values, and execution time. A command is executed by writing its binary value in the command register (i.e., base address) when the BUSY bit is low. Some of these commands require no parameters while others require one or two parameters which must be loaded into the *Parameter 1* and *Parameter 2* registers (Table C.2) prior to loading the command register. For more information on the AD1170 command set, the reader should refer to Analog Devices (1992). The next section shows how to start a conversion under computer command and retrieve the conversion data. The last section

describes how to program the AD1170 chip to control its advanced features such as user specified integration time, background calibration, and electronic null.

### C.2.2.2. Retrieving Conversion Data

Analog-to-digital conversion using the default value of integration period (See Section C.2.2.3) is triggered by writing CNV command (decimal 8) to the command register. The conversion data can be read as soon as the BUSY bit goes low. Since the AD1170 can present data in formats up to 22 bits, the digital data is retrieved in three bytes from three separate registers (Table C.2). Since the actual number of bits in the output can be anywhere from 7 to 22, the output data is always right justified (i.e., flushed to the right) within the three output bytes. The three output bytes are converted to a voltage reading by 'bit shifting' and relating the bit count to the full span ( $\pm 5$  V) the converter as follows<sup>3</sup>:

$$\text{COUNT} = \text{LC.BYTE} + 256 * \text{MIDBYTE} + 65536 * \text{HIBYTE} \quad (\text{C.3})$$

$$\text{VOLT} = (\text{COUNT} / 2^{\text{RESOLUTION}}) * 10.0 - 5.0 \quad (\text{C.4})$$

where LOBYTE, MIDBYTE, and HIBYTE are low, mid, and high bytes respectively, and RESOLUTION is the current value of the converter bit precision. The FlexCAT code listing in Appendix D contains a C language function called AdcardRead which shows how to select an input channel, start a conversion, read the conversion data and compute voltage reading.

---

<sup>3</sup> This assumes that the data is in offset binary format.

### C.2.2.3. Advanced Features of AD1170

The AD1170 architecture allows a programmable data format from 7 to 22 bits in either offset binary coding or two's complement coding. In the two's complement code, positive numbers are represented with a zero sign bit. The negative of a number is obtained by complementing each bit of the positive number and adding one. The offset binary code is similar to two's complement except the MSB (Most-Significant Bit) is inverted. Programming the data format is accomplished by loading a format code (refer to Table C.5) in *Parameter 1* register and executing the SDF command. It should be noted that the useful resolution of AD1170 is largely dependent upon factors such as integration period and calibration period. The reader should refer to Analog Devices (1992) for the relationship between integration period and usable resolution.

It is possible to set the default integration period to one of the seven preset periods using the SDI command (Table C.4). For single conversions without altering the default integration time, the CNVP command may be used, which also allows the selection of one of these seven preset periods. Table C.6 shows the last three bits which should be inserted into the SDI and CNVP commands to select a particular integration time. The AD1170 achieves its excellent specifications by calibrating itself against its internal reference voltages. The user can control the frequency of occurrence of calibration cycles and their duration. The calibration period is set to one of the seven preset periods (Table C.6) using the SDC command. The argument for the SDC command is the same three bits used for the SDI and CNVP commands. The user may also disable or enable background calibration using the CALEN and CALDI commands or perform a single calibration cycle by sending the SCAL command. When the background calibration mode is enabled, the converter will initiate a calibration cycle

whenever it is idle. However, any conversion commands received during such a calibration cycle will cause that cycle to be aborted in favor of the conversion.

The electronic null feature of the AD1170 is very useful for compensating offset errors of the analog circuitry preceding the converter without the use of cumbersome trim potentiometers. When both the high and low sides of an input are exactly equal, the output of the instrumentation amplifier should ideally be zero. However, combined input and output offset errors of the amplifier give rise to a small output signal (several millivolts typically). The electronic null operation is performed in two steps. The voltage to be offset is presented to the converter and the NULL command executed. This stores the measured value of offset in an internal random access memory (RAM). The NULEN command enables the null capability by subtracting this offset from all subsequent conversions. The NULDI command cancels the NULEN command's effect but does not affect the values of the offset stored in internal RAM. The offset value to be nulled should ideally be no more than a few hundred millivolts in amplitude. Another useful feature of the AD1170 is the capability to correct gain (slope) errors of the converter and the analog circuitry preceding it. The ECAL function measures the ratio of the internal reference voltage (+5 V) to an externally applied voltage and stores it in internal RAM. The resulting coefficient is applied to the mathematical computations for all subsequent conversions. In order to use the electronic calibration feature, the input of the AD1170 is presented with a very accurate +5 V reference signal before invoking the ECAL command.

The AD1170 contains an internal non-volatile random access memory (NVRAM) whose contents are preserved even when there is no power. The non-volatile memory is used to store the various parameters associated with the converter's operation (e.g. the integration and calibration periods, data format, NULL and ECAL



coefficients, etc.). The current values of these parameters can be saved by sending the SAVA command. The RESA command resets all operating parameters to their respective values stored in the non-volatile memory during the last SAVA command. Eight 16-bit words of the non-volatile memory are available to the user for general purpose use and may be accessed using RDNV and WRNV commands. The non-volatile memory used in the AD1170 has a finite endurance of 1000 write cycles and is best used for data which change infrequently.

### C.3. Digital-to-Analog Conversion

A digital-to-analog (D/A) converter accepts a digital input and produces an analog output. The D/A converter can be thought of as a digitally controlled potentiometer which produces a voltage or current output that is a normalized fraction of its "Full Scale" setting. At minimum, a D/A converter consists of a voltage or current reference, a set of electronic switches, a binary weighted precision resistor network, and a means of summing the weighted currents. The full-scale setting is most commonly provided by a stable, precision voltage reference. The electronic (solid-state) switches, driven by digital logic, steer currents through the resistor network to an operational amplifier (op amp). There are many ingenious schemes of arranging a set of resistors to produce a binary weighted output. The simplified circuit in Figure C.6a illustrates how the most popular type of resistor network, the R-2R ladder works. If all bits are low (i.e., all switches connected to the ground), the output voltage, given by the product  $IR$ , will be zero. If the most-significant bit (MSB) is made 1 (high), the output voltage will be approximately  $V_{ref}/2$ . Similarly, if the next most significant bit is turned on while all others are low, the output will be  $V_{ref}/4$  and so on. The least-

significant bit (LSB) will contribute  $V_{\text{ref}}/2^N$  to the total output where  $N$  is the bit resolution of the converter. The transfer function of this type of D/A converter is:

$$V_o = V_{\text{ref}} A/2^N \quad (\text{C.5})$$

where  $V_o$  and  $V_{\text{ref}}$  are the output and reference potentials, and  $A$  the decimal value of the applied binary word.

It is clear from the above discussion that output of a D/A converter is a step function and the smallest possible step size (also called LSB value) is related to the bit resolution. For a 12 bit converter with 10 V full-scale range (0-10 V or  $\pm 5$  V), the LSB value is about 2.44 mV. This resolution is adequate for most control applications. The static (DC) accuracy of a D/A converter is described by three error terms encountered earlier in Section C.2. These are: a) offset (zero) error, b) gain (full-scale) error, and c) linearity (relative accuracy). The offset error is the actual output of a D/A converter when the digital code calls for a zero output. It affects all codes by the same additive amount. The definitions of other two error terms are very similar to the ones given in Section C.2 for analog-to-digital converters. The change in offset and gain errors with temperature (drift) is mainly related to temperature stability of the voltage reference. It is, therefore, advisable to use only the highest quality reference with a D/A converter which requires an external reference.

### C.3.1. AOC-8 Architecture

Figure C.5 shows the block diagram of the AOC-8 analog output card. The card has been designed for the PC-XT bus (8 bit data path) and can be used with any IBM compatible computer. The AOC-8 card provides up to eight analog channels whose output can be independently set to one of the four ranges: 0-5V, 0-10V,  $\pm 2.5$ V, and  $\pm 5$ V. The heart of the card is the AD767 digital-to-analog converter chip

manufactured by Analog Devices. AD767 is a high stability, voltage output, 12 bit D/A converter chip complete with an output amplifier and on-chip data latch. Because a 12 bit digital-to-analog converter costs considerably less compared to a high resolution A/D converter, the AOC-8 card uses one converter chip per channel. In addition to the AD767 chips, the card contains a number of TTL (Transistor-Transistor Logic) integrated circuits which provide an interface between the card and the computer's bus. Since the IBM PC-XT bus is able to transfer only 8-bits of data at a time, the 12-bit digital data to the converters must be sent in two consecutive write cycles. Two 8-bit write-only data registers on board the AOC-8 card temporarily store the bytes received from the computer. Under a separate command, they combine these data into a 12-bit word and release it on an internal 12-bit bus shared by all AD767 chips.

#### C.3.1.1. AD767 Chip

The AD767 combines a voltage output 12-bit D/A converter, a high stability buried Zener reference, and an input latch on a single IC chip (Figure C.6b). It uses 12 precision high-speed bipolar current steering switches and a laser-trimmed thin-film resistor network to achieve high accuracy (Analog Devices, 1992). Table C.7 lists some of the important specifications of the device. The data is loaded into the converter via 12 parallel digital lines. Microprocessor compatibility is achieved by on-chip latch which captures the data when triggered by a pulse applied to the *chip select* ( $\overline{CS}$ ) pin. The internal output amplifier provides sufficient current to drive a 1 k $\Omega$  load. Internal scaling resistors provided in the AD767 can be used to produce unipolar output voltage ranges of 0 to +5 V or 0 to +10 V, or bipolar output voltage ranges of  $\pm 2.5$ ,  $\pm 5$ , or  $\pm 10$  volts. Although the converter is guaranteed to work with power supply voltage as low as  $\pm 12$  V, a minimum of  $\pm 12.5$  V supply voltage is required for the  $\pm 10$  V output range. A DC-DC step up converter (Model J05D15 from Computer Products, Inc.)

converts the +5 V voltage tapped from the computer's bus into  $\pm 15$  V and is able to supply up to 200 mA current.

### C.3.1.2. Digital Interface

The main functions of the digital circuitry on the AOC-8 card are: a) to decode the PC system bus address signals and capture 8-bit data during a write cycle, b) to provide a temporarily storage for the data received in two consecutive bytes, and c) shift and combine these two bytes to form a 12-bit word and feed it to the selected digital-to-analog converter. The AOC-8 card has a DIP switch bank very similar to the one provided on the AIC-16 card (Section C.2.2) for setting base address. The card occupies three consecutive locations (ports) in the computer's address space beginning with the base address. Two write-only data registers (Figure C.5) occupy the two lowermost ports and hold two bytes of information. The registers are connected to an internal 12-bit wide bus which is shared by all D/A converters. The output channels are numbered from 0 to 7. A three bit code<sup>4</sup> written to the highest port (i.e., base address + 2), called *channel select* register, selects one of the converter chips and releases the byte information stored previously in the data registers onto the internal bus. The selected converter captures this new word and its analog output immediately reflects the most recently loaded code.

### C.3.2. Operation of AOC-8 Card

The AOC-8 card also uses the extended address decoding scheme described in Section C.2.2. It is recommended that the base address of the card should be set to 768 or its next three equivalents (i.e., 1792, 2816, 3840). The analog output range of each

---

<sup>4</sup> Although the actual code sent is in form of a byte (8-bits), all higher bits are ignored.

converter can be independently selected by changing the positions of a set of switches provided on the card.<sup>5</sup> Although, the AD767 chip offers five output range settings, the present AOC-8 card design does not offer  $\pm 10$  V range. A group of eight DIP switches (similar to the base address switches) select between unipolar and bipolar modes. The full-scale (FS) range of a converter is set to either 5 V or 10 V by the larger sliding switch located next to the converter chip. The combinations of these two switches provide four output ranges: 0-5 V, 0-10 V,  $\pm 2.5$  V, and  $\pm 5$  V. Once the base address and output ranges are set properly, the card can be installed in any empty expansion (bus) slot located on the motherboard of an IBM compatible computer.<sup>6</sup>

The value of desired output voltage is first converted to an integer number (bit count) between 0 and 4095 ( $2^{12} - 1$ ) in following manner. For unipolar setting:

$$\text{COUNT} = \text{INT} ((\text{VOLT} / \text{RANGE}) * 4095) \quad (\text{C.5})$$

while for bipolar setting:

$$\text{COUNT} = \text{INT} ((\text{VOLT} + 0.5 * \text{RANGE}) / \text{RANGE}) * 4095 \quad (\text{C.6})$$

where RANGE is the full-scale (FS) range of the converter. The INT command in BASIC returns integer value of a fractional decimal. Since the 12-bit input is sent to the converter in two write cycles, bit shifting is required to split the bit count into two 8-bit bytes. In BASIC language, the bit shifting can be accomplished using integer arithmetic:

$$\text{HIBYTE} = \text{INT} (\text{COUNT} / 256) \quad (\text{C.7})$$

---

<sup>5</sup> The ranges on the hand-wired prototype card are fixed.

<sup>6</sup> The card should never be inserted into or extracted from a slot while the computer is on.

$$\text{LOBYTE} = \text{INT}(\text{COUNT} - \text{HIBYTE} * 256) \quad (\text{C.8})$$

where LOBYTE and HIBYTE respectively are the integer numbers corresponding to the lower 8 and upper 4 bits of a 12 bit word. Bit shifting is not necessary in C language since the outport(int address, int count) function automatically splits a 16-bit integer and sends the constituents to two adjacent ports (i.e., address and address+1). After sending the low and high bytes to their destinations, the desired channel number is written to the channel select register (i.e., base address+2). This completes the process and updates the analog output reading of the selected channel. The function DacardWrite in the FlexCAT code listing (Appendix D) shows how to write an efficient driver routine for the AOC-8 card using C language.

### C.3.3. Calibration Procedure

The AD767 has an internal low-noise buried Zener diode reference which is laser-trimmed to 10.00 V with a  $\pm 1\%$  maximum error. Although this is adequate for most applications, the AOC-16 card provides potentiometers (variable resistors) which can be used to trim the gain error to zero. The gain error can be manually trimmed using a precision voltmeter and a screw driver for turning the potentiometers. Each AD767 converter on the AOC-8 card has been connected to a 20-turn  $100\Omega$  potentiometer. To trim the gain error in 0-10 V unipolar mode, turn all bits ON (i.e., write 4095 decimal) and adjust the potentiometer until output of the selected channel is 9.9976 volts. In bipolar  $\pm 5$  V range, turn ON all bits and adjust the potentiometer to give a reading of +4.9976 volts.

Characteristic	Typical Value	Units
Maximum Resolution**	18	bits
Accuracy (Nonlinearity)	$\pm 0.001$	% SPAN
Conversion Rate		Conv./sec.
Integration time = 1 ms	250	
Integration time = 16.67 ms	50	
Integration time = 100 ms	9	
Stability	$\pm 9$	ppm SPAN/ $^{\circ}$ C
Input Range	$\pm 5$	Volts
Input Impedance	100	M $\Omega$
Warm up Time	15	minutes

\*\* Although the converter can format the output up to 22 bit wide word, the usable resolution is limited by noise, which is largely determined by the integration and calibration period.

Table C.1: Important Specifications of the AD1170 Converter.

Register Location	Read Function	Write Function
Base Address + 3	High Data	(Unused)
Base Address + 2	Mid Data	Parameter 1
Base Address + 1	Low Data	Parameter 2
Base Address	Status Byte	Command

Table C.2: The AD1170 Address Space Utilization.

Bit No.	Mnemonic	Assigned Function
0	BUSY	When low, indicates that the AD1170 is ready to receive a command. If high, it indicates that the converter is busy executing the last command.
1	DATA RDY	When high, indicates that the data from the most recent conversion is available in the data registers. This bit is cleared at the start of a new conversion cycle.
2	DATA SAT	This bit is set high by any conversion which is saturated, i.e. whose output data exceeds positive or negative full scale.
3	CMD ERROR	When high, indicated that the most recently loaded command contained a contextual or syntactic error, or was not recognized. It is cleared when the next command is loaded.
4	INT	This bit goes high to indicate that the input is currently being integrated.
5	PWR UP	This bit goes high when the converter is powered up and remains high until device initialization is complete.

Table C.3: Interpretation of the AD1170 Status Byte.



Mnemonic	Binary Value	Description	Execution Time
CNV	00001000	Perform a single conversion using default integration time	$T_{int} + 3 \text{ ms}$
CNVP <sup>†</sup>	00010C <sub>2</sub> C <sub>1</sub> C <sub>0</sub>	Perform a single conversion using specified integration time	$T_{int} + 3 \text{ ms}$
SDI <sup>†</sup>	00111C <sub>2</sub> C <sub>1</sub> C <sub>0</sub>	Set default integration period	150 $\mu\text{s}$
SDC <sup>†</sup>	01000C <sub>2</sub> C <sub>1</sub> C <sub>0</sub>	Set default calibration period	160 $\mu\text{s}$
SDF	00110000	Set default calibration format	140 $\mu\text{s}$
SCAL	11000000	Perform a single calibration	$2 * T_{cal} + 9 \text{ ms}$
CALEN	10110000	Enable background calibration	300 $\mu\text{s}$
CALDI	10111000	Disable background calibration	310 $\mu\text{s}$
NULL	01110000	Measure the offset voltage at the AD1170 input and store in RAM	$T_{int} + 3 \text{ ms}$
NULEN	01111000	Subtract NULL measured value from all subsequent conversions	250 $\mu\text{s}$
NULDI	10000000	Cancel the effect of the NULEN command	250 $\mu\text{s}$
ECAL	00011000	Perform electronic calibration routine	1.5 seconds
SAVA	01001000	Save all parameters to the non-volatile memory (NVRAM)	150 ms
RESA	01101000	Restore all non-volatile parameters from memory	2.3 ms
WRNV*	10011A <sub>2</sub> A <sub>1</sub> A <sub>0</sub>	Write a word to user NVRAM	22 ms
RDNV*	10100A <sub>2</sub> A <sub>1</sub> A <sub>0</sub>	Read a word from user NVRAM	600 $\mu\text{s}$
RST	10010000	Reset the AD1170 to power up condition	210 ms

<sup>†</sup> The lowest three bits indicate the desired value of integration or calibration period (refer Table C.6).

\* The address of the desired NVRAM location is embedded into the lowest three bits.

Table C.4: The AD1170 Command Set.

Data Format	Binary Format Code
Two's Complement	XXX1XXXX
Offset Binary	XXX0XXXX
22 bits	XXXX1111
21 bits	XXXX1110
20 bits	XXXX1101
19 bits	XXXX1100
18 bits	XXXX1011
17 bits	XXXX1010
16 bits	XXXX1001
15 bits	XXXX1000
14 bits	XXXX0111
13 bits	XXXX0110
12 bits	XXXX0101
11 bits	XXXX0100
10 bits	XXXX0011
9 bits	XXXX0010
8 bits	XXXX0001
7 bits	XXXX0000

Note: X = Do not care.

Table C.5: AD1170 Data Format and Format Code.

Integration Time	C <sub>2</sub>	C <sub>1</sub>	C <sub>0</sub>	Comments
1 Millisec.	0	0	0	
10 Millisec.	0	0	1	
16.667 Millisec.	0	1	0	1 Cycle @ 60 Hz
20 Millisec.	0	1	1	1 Cycle @ 50 Hz
100 Millisec.	1	0	0	50/60 Hz
166.67 Millisec.	1	0	1	10 Cycles @ 60 Hz
300 Millisec.	1	1	0	50/60 Hz

Table C.6: Preset Integration Periods.

Characteristic	Typical Value	Units
Resolution	12	bits
Gain Error	$\pm 0.1$	% of FSR*
Unipolar Offset Error	$\pm 1$	LSB <sup>†</sup>
Nonlinearity (Accuracy)	$\pm 0.5$	LSB
Thermal Drift (Gain)	$\pm 5$	% of FSR
(Offset)	$\pm 1$	
Output Ranges (Unipolar)	0-5, 0-10	Volt
(Bipolar)	$\pm 2.5, \pm 5, \pm 10^{\ddagger}$	
Output Current	$\pm 5$	mA
Digital Inputs		
Logic "0"	0.8	Volt (max.)
Logic "1"	2.0	Volt (min.)
Power Supply		
Rated Voltages	$\pm 12, \pm 15$	Volt
Power Consumption	400	mW

**Notes:** \* Full Scale Range. FSR is 10V for  $\pm 5V$  range and 20V for  $\pm 10V$  range.

<sup>†</sup> Least-Significant Bit =  $FSR/2^n$  where n is resolution in bits.

<sup>‡</sup> This range requires a minimum power supply of  $\pm 12.5V$ .

Table C.7: Important Specifications of the AD767 D/A Converter.



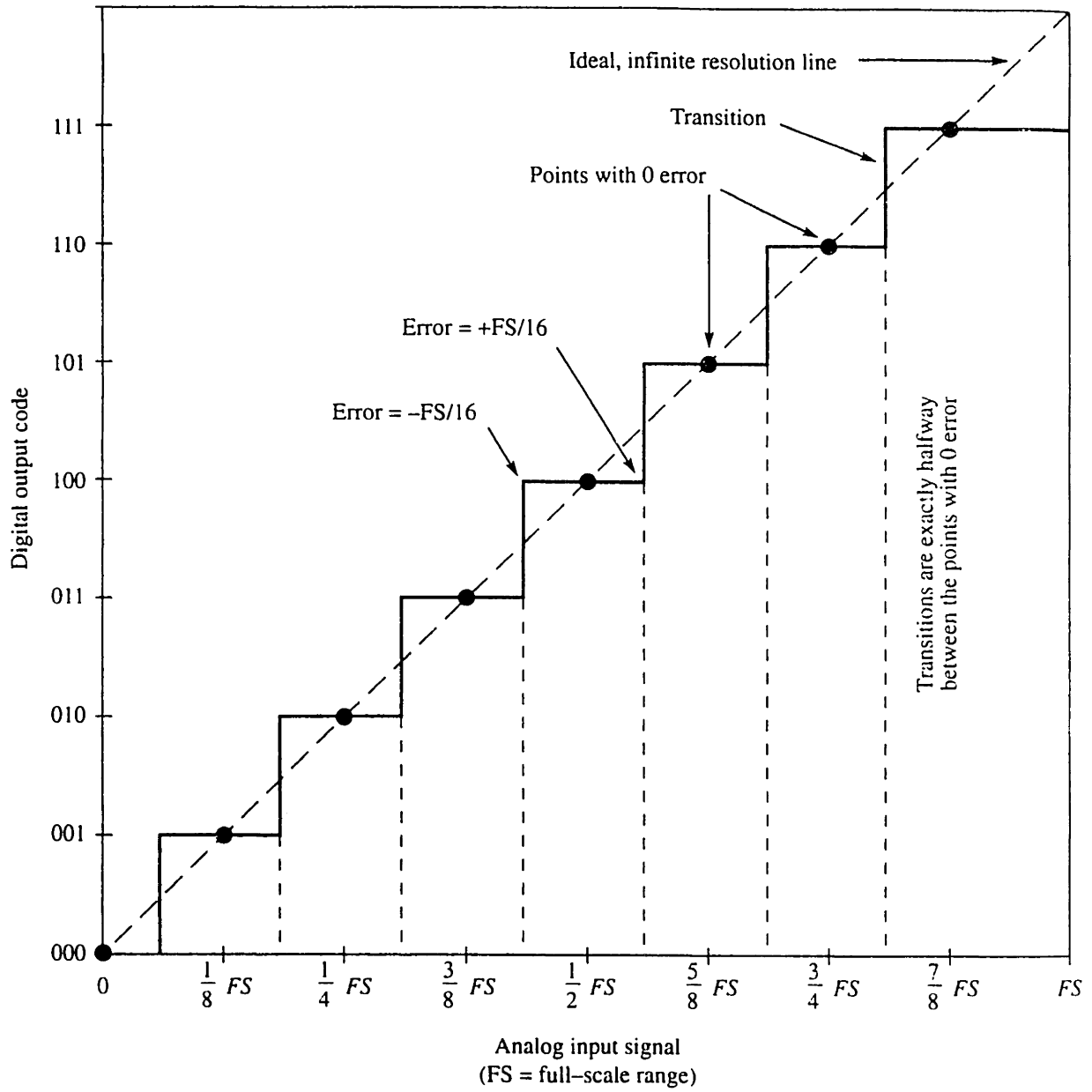


Figure C.1: The Ideal Transfer Function of a 3-bit A/D Converter.

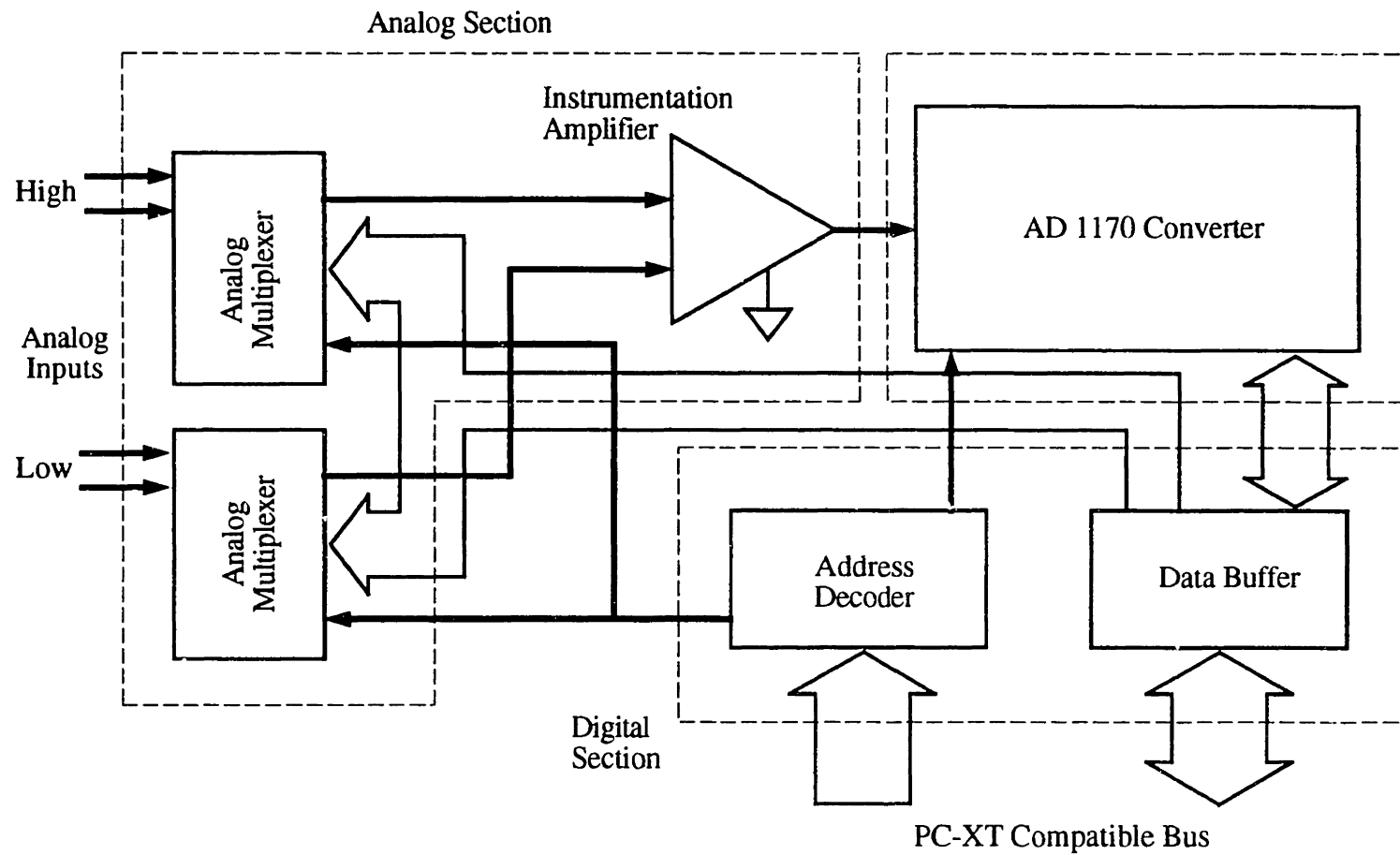
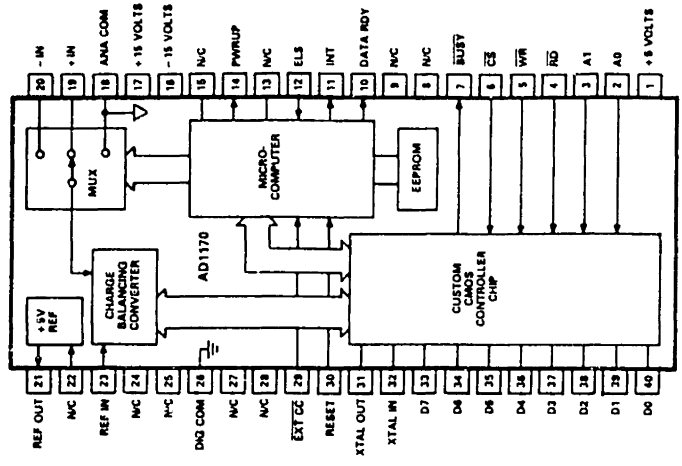
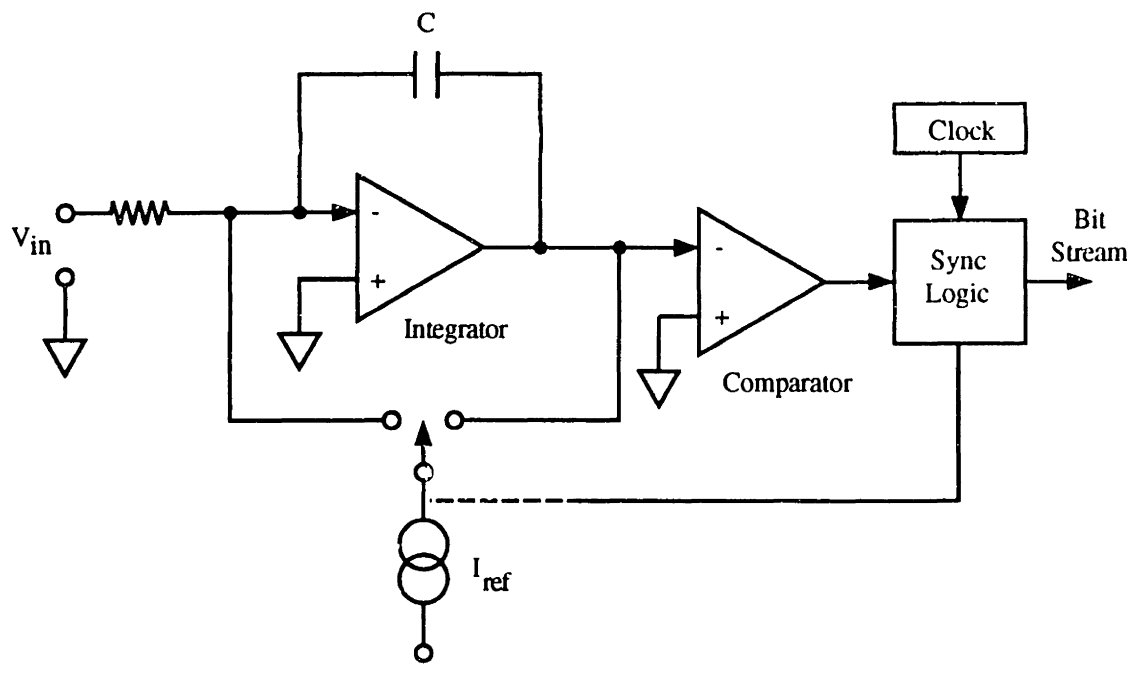


Figure C.2: Block Diagram of the AIC-16 Analog-to-Digital Converter Card.



(a) Architecture of the AD1170 A/D Converter Chip



(b) Voltage-to-Frequency Technique of A/D Conversion

Figure C.3

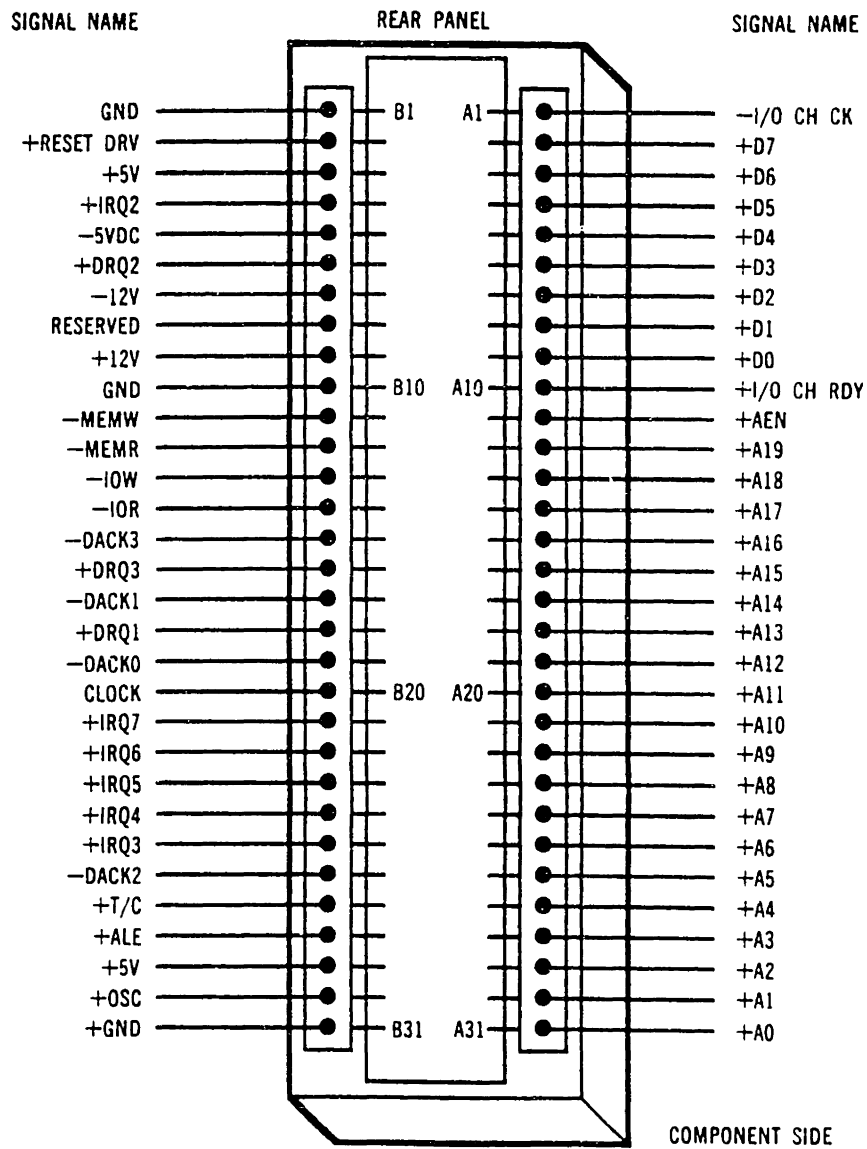


Figure C.4: Pin Assignments of the IBM PC-XT Bus.



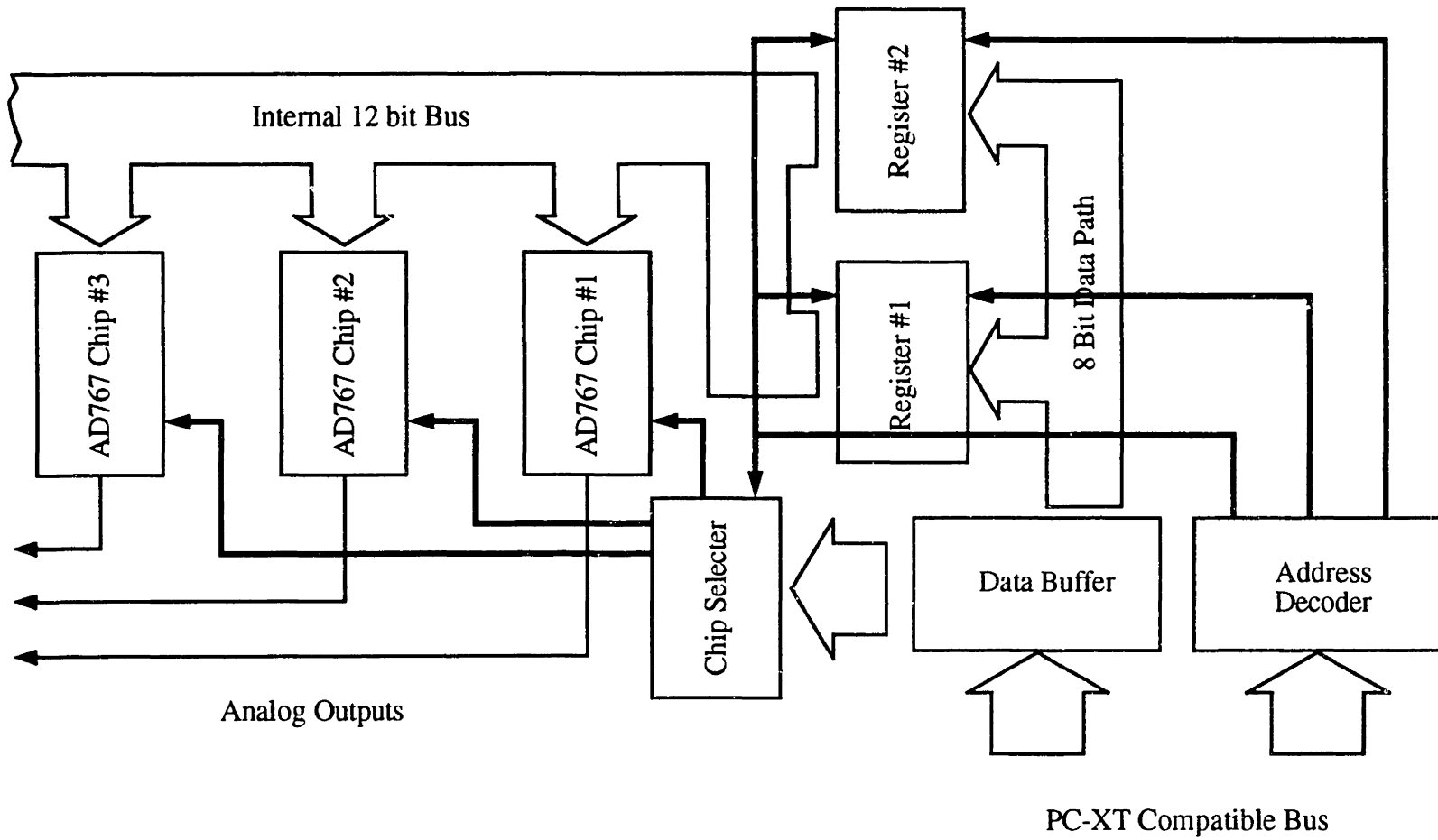
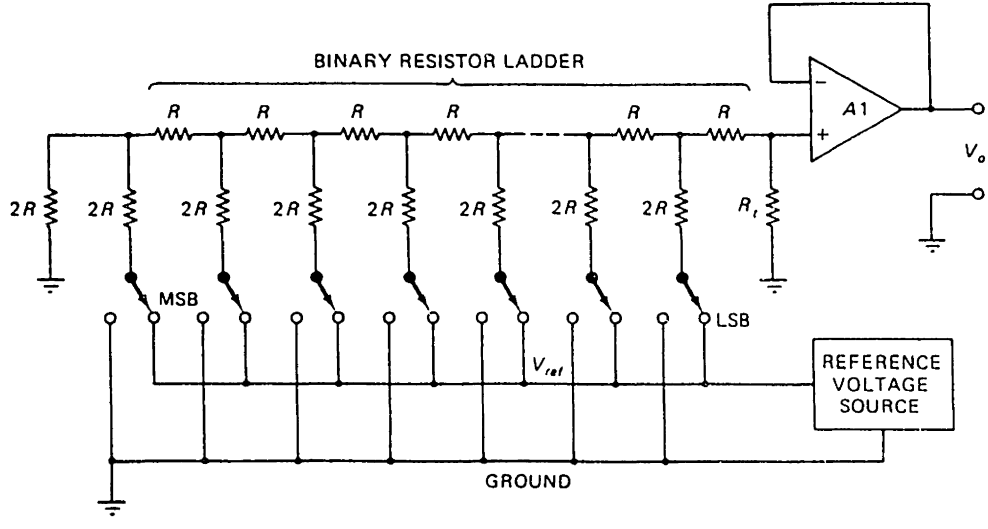
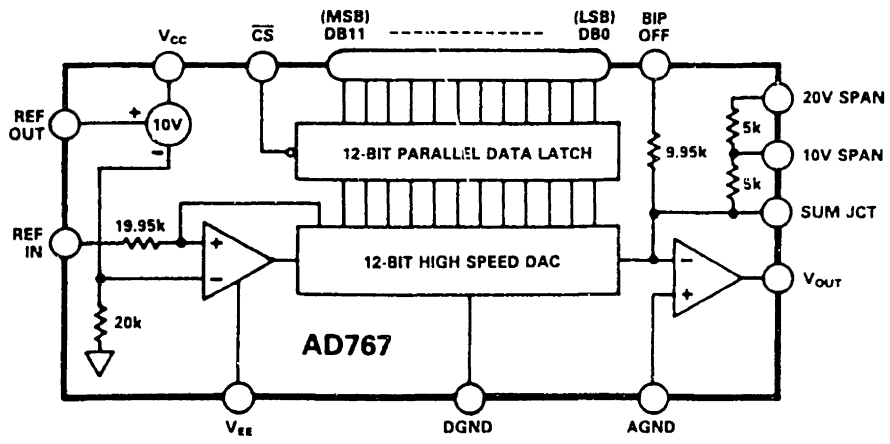


Figure C.5: Block Diagram of the AOC-8 Digital-to-Analog Converter Card.



(a) The Basic D/A Conversion Using a R-2R Ladder Network.



(b) The AD767 Functional Block Diagram.

Figure C.6

# Appendix D: APSR Control Software

## D.1. Introduction

The APSR cell uses an IBM compatible microcomputer (PC) for test automation and data acquisition. The overall performance of the APSR cell depends on the integrity of the software which controls and coordinates various aspects of the test. Therefore, a considerable effort was spent in developing a modular, object-oriented, real-time control software for the APSR test. As part of a long term strategy of encouraging code reusability within the MIT Geotechnical laboratory, the software was divided into two separate modules. A general purpose module, called FlexCAT (FLEXible integrated software framework for Computer Aided Testing), provides a generalized framework for test automation and data acquisition programming. The parts of the code specific to the APSR cell are condensed into a user module which is then plugged into the FlexCAT (module) to create a complete, stand-alone application. Since the user module, which contains just 20% of the total code, can be readily modified for running other types of Geotechnical laboratory tests, this two step approach preserves a substantial portion of the original investment. To a user (operator), the FlexCAT appears as a set of useful commands accessible through a menu bar available at the top of the screen of every application developed within the FlexCAT framework. To a programmer, the FlexCAT is a tool kit of useful functions which facilitate the task of writing a real-time control and data acquisition application in a DOS<sup>®</sup> environment.

The first part of this appendix describes various elements of the APSR user interface and shows how to enter test parameters for a typical APSR test. The second part is a user's guide to the FlexCAT and describes various facilities provided through the FlexCAT menu bar. This is followed by a complete listing of the program code.

## D.2. APSR Interface

The APSR test software presents a mouse-driven, user-friendly interface shown in Figure D.1. The computer screen is divided into five non-overlapping panels. These panels contain control elements such as push buttons, selection lists, and input/output fields which can be manipulated using either the mouse or keyboard. This section describes control elements on each panel and shows how to enter various APSR test parameters.

### D.2.1. Sigma 1 Panel

The horizontal panel at the top of the screen contains controls related to the major principal axis system of the cell. The push button marked *Run* is used to start and stop the piston feedback loops. The FlexCAT system should be in *Run* mode (refer to Section C.3.1.) before any feedback loops can be activated. The platform pistons can be driven in one of the three modes selected through the control marked *Test Mode*: a) Strain Control, b) Stress Control, and c) Hold Stress. The test mode can be changed only when the *Run* button is turned off. The Strain Control mode, which is the standard mode, drives the platforms at a constant rate of displacement. The selected displacement rate is entered through the input field marked *Target*. A positive value of target rate advances the pistons while a negative value retracts them.

In addition to target rate, the controller requires displacement and stress cutoff limits for safety reasons. As soon as either of these limits is reached, the controller turns off all piston motors and displays a warning message. The target rate and cutoff limits can be changed any time during a test. However, these values are buffered and passed on to the controller only when the *Run* button changes from the 'off' state to the 'on' state. This buffering scheme prevents the controller from accepting one data item at a time, and hence ensures integrity of the data set. The program checks if all input parameter values are reasonable and displays a warning message whenever a value falls outside its preset limits. The panel shows piston DCDT readings (in millimeter) in four output fields marked *DCDT 1* through *DCDT 4*. If the difference between the positions of two adjacent pistons exceeds 0.125 mm, the controller turns off all piston motors and displays a warning message. Other output fields on the panel show values of total axial strain and axial stresses for each platform.

### D.2.2. Sigma 2 Panel

The panel in the lower left corner (Figure D.1) contains controls related to the active plane strain system of the APSR cell. The panel has four output fields which display values of the top and bottom wall pressures, total out-of-plane strain, and the *b* parameter. The *Run* button starts the feedback loops which minimize the out-of-plane strain. When this button is turned on, the controller designates the current sidewall proximity sensor readings as the new target values and resets the plane strain to zero. Turning the *Run* button off stops sidewall pressure actuator motors and allows the sidewalls to deform freely.

### D.2.3. Sigma 3 Panel

The panel in the lower right corner of the screen shows controls related to the system that maintains a constant confining stress (air pressure) in the APSR cell. The selected value of air pressure is entered through the input field marked *Target*. Like the axial strain rate, the target air pressure value is buffered and only becomes effective when the *Run* button changes from the 'off' to the 'on' state. The panel has output fields for displaying the actual air pressure, total lateral strain, and shear stress ratio (R Value). The program checks the target pressure value and warns the operator if it is unreasonable or if the controller is unable to maintain air pressure due to any hardware problems.

### D.2.4. Reinforcement and Graph Panels

The central panel pertains to the reinforcement position control system. The *Run* button on this panel activates and deactivates the reinforcement feedback controller. The reinforcement position is monitored by an infrared sensor which automatically sets the target location when the controller is turned on and the position value in the output field is then reset to zero. The small panel located below the reinforcement panel contains a selection list for selecting one of the several available graphs. Selecting a particular graph installs a graph window similar to that shown in Figure D.2. The user can manipulate two floating cursors on the graph window and read off values corresponding to the cursor positions and the slope of the line connecting cursor points. The graph can be saved as a Postscript<sup>®</sup> format file for a subsequent hard copy output by clicking on the *Print* button.

## D.3. FlexCAT User's Guide

The FlexCAT is based on the object-oriented approach to real-time programming. In this approach, every physical (as well as non-physical or logical) object in the problem domain has its counterpart in the program domain. Each object combines a set of information (data) about itself and operations which can be performed on the data. Thus, in the real-time control domain, a Sensor object might have its serial number, calibration factor, offset (zero), unit, maximum limit etc. as part of its database. The *Read*, *Calibrate*, *Test* etc. are examples of operations that can be performed on a Sensor object. The FlexCAT maintains the object database on the computer's hard disk and loads it into RAM every time the program is started. Every test control and data acquisition application program developed within the FlexCAT framework has a special menu bar at the top of the screen. The user can use the menu commands to enter and edit information in object database and perform operations on most FlexCAT objects.

### D.3.1. Overview

The FlexCAT system contains eight objects which are commonly encountered in the feedback control and data acquisition domain. These are: System, A/D Card, D/A card, Sensor, Servo, Loop, Data Acq, and Graph. All test control and data acquisition application programs developed within the FlexCAT framework automatically incorporate these objects. As a result, the task of writing the user module is reduced to manipulating these higher level (abstract) objects. Most of the menu commands described in the following sections deal with these objects.

Table D.1 lists the database contents and operations for all eight FlexCAT objects. The System object is the highest level entity in the FlexCAT. It represents the

agglomeration of all hardware components attached to a testing device. The System database consists of the total numbers of A/D and D/A cards, sensors, servos, and loops present in the system. The Cycle Time is the time interval at which the user background tasks are executed. The System object can exist in either *Run* or *Stop* state. The program performs the background activities only when the System is in the *Run* state. By definition, there is only one System object. There can be multiple copies of all other objects; each one identified by a unique serial number starting with zero. In addition, the Sensor, Servo, and Loop objects may be given up to 15 character long names for easy identification.

The Sensor object represents any measurement device (LVDT's, pressure transducers, strain gages, etc.) attached to the system. Every Sensor object is associated with a unique combination of A/D card number and channel number. The Sensor database has a unit and a calibration factor for converting raw voltage readings into the engineering quantity expressed in that unit. The Sensor database also maintains two values of zero (offset) which are subtracted from a raw voltage reading before it is multiplied by calibration factor. The absolute zero represents the "true" offset value of sensors such as pressure transducers and strain gages. The other (floating) zero is an arbitrarily adjusted offset value. The *Read* operation on a Sensor object returns two numbers; an absolute reading and a (floating) reading. The *Set Zero* operation designates the current voltage reading as the new zero while the *Restore Zero* operation sets the zero to absolute zero. The Servo object is any device that can be manipulated by the computer (e.g. switches, valves, motors, electro-pneumatic regulators, etc.). Every Servo object is associated with a unique set of D/A card number and channel number. The *Write* operation sends a specified voltage value to a servo while the *Reset* operation writes zero voltage to a servo.



The Loop is an abstract object which consists of a Sensor and a Servo object connected in feedback configuration. It stores the PID loop constants and a Cycle Count in its database. The Cycle Count is a counter which is decremented every time the program visits a loop in the background. The PID calculations are updated only if the Cycle Count is equal to zero. By setting the Cycle Count to a value greater than one, it is possible to execute a background control loop at intervals which are multiples of the system Cycle Time. This is useful in a situation when some devices (e.g. axial strain motor in a triaxial test) need more frequent attention than others (e.g. cell pressure controller). The Data Acq object represents a data acquisition task. The FlexCAT is a single user environment and therefore, only one data acquisition task can be setup and run at a time. The data collected during a test can be displayed in graphical form anytime during and after the test. The Graph object represents templates on which the graph data are plotted. A graph can be rescaled during a test by changing the axis limits. Alternatively, the auto-scale feature can be selected by simply by setting both minimum and maximum axis limits to zero.

### D.3.2. System Menu

As mentioned earlier, the System object is the highest level entity in the FlexCAT.

- 1. Run:** This command enables the scheduler and starts background activities. While in Run mode, the program continuously reads all active channels on the analog-to-digital converter cards and updates values of input voltages. The rate at which input is updated depends on the integration time and the number of active channels (refer to Setup→A/D Card command).

- 2. Stop:** This command disables the scheduler and stops all background activities including data acquisition and feedback control.
- 3. About:** Displays a window containing information about the FlexCAT.
- 4. Exit:** Closes a FlexCAT application and returns the control to the DOS<sup>®</sup> command level.

### D.3.3. Setup Menu

Commands in this menu are used to enter various test setup parameters in the FlexCAT database. The FlexCAT stores this information in a binary file called FLEXCAT.DBS in the default directory. The information related to System, A/D Card, and D/A Card objects cannot be modified when the program is in the *Run* mode.

- 1. System:** This command displays the window shown in Figure D.3 and prompts the user to enter the total number of A/D and D/A cards, Sensors, Servos, Loops, and Graphs in the system.
- 2. A/D Card:** The user is able to set the base address, integration time, calibration time, bit resolution, and number of active channels of all AD1170 analog-to-digital cards present in the system through this command. It also allows the user to select the background calibration and electronic null functions of the AD1170 chip (refer to Appendix C for details). The recommended values can be set simply by clicking on the *Default* button.
- 3. D/A Card:** This command lets the user specify the base address, bit resolution, and range/polarity of channels for the AD767 digital-to-analog cards (refer to Appendix C for details) present in the system.
- 4. Sensor:** This command allows the user to enter the name, calibration factor, A/D card number, channel number, unit, and maximum allowable

value of every sensor in the system. The program issues a warning if a sensor reading exceeds its maximum allowable limit during a test.

- 5. Servo:** This command is used to enter the Servo names, D/A card numbers, channel numbers and voltage limits. The program issues a warning whenever a servo output voltage exceeds the set limits. When the auto-reset feature is turned on, the servo voltage is automatically set to zero every time the program starts or terminates.
- 6. Loop:** This command allows the user to input the loop name, sensor number, servo number, values of PID parameters, and *Cycle Count* for any feedback loop in the system. The loop computations are executed at a time interval equal to the system *Cycle Time* multiplied by the *Cycle Count*. The default value of the *Cycle Count* is one.
- 7. Graph:** All graph templates are numbered and their characteristics are stored in the system database. Through this command, the user can enter and modify the graph title, axis titles, axis range, number of ticks, number of plots etc.
- 8. Print:** The information contained in the system database can be printed in the form of an ASCII file using this command. This file can later be printed on any printer using the DOS<sup>®</sup> *print* command.
- 9. Save:** This command saves the setup information on the computer's hard disk. This command must be invoked every time changes are made in the system setup, or else the new information will be lost when the program terminates.

### D.3.4. Calibration Menu

Commands in this menu allow the user to calibrate the analog-to-digital cards and sensors attached to the system. The user can also tune feedback loops using the procedure described in this section.

- 1. A/D Card:** This command uses the built-in electronic calibration facility of AD1170 chip to correct the system level offset and gain errors (refer Appendix C for details). After entering the number of the A/D card to be calibrated, the user can choose any of the four options from Options Menu. *Single Calibration* performs one calibration cycle against the internal reference of the chip. To correct the overall (system level) offset, short out the channel number zero (i.e., connect the +ve and -ve terminals) of the card before executing the *Offset Correction* option. The gain (slope) error of the converter is corrected by presenting +5.0000 volt to channel zero of the card and executing *Gain Correction* option. The *Save Parameters* option writes the current values of offset and gain corrections along with other operating parameters to on-chip non-volatile (permanent) memory. These values are automatically recalled from the memory upon power up. The non-volatile memory can sustain a maximum of 1000 write cycles.
- 2. Sensor:** It is possible to compute the calibration factor of any sensor attached to the system using this command and a suitable calibration device. The command displays the window shown in Figure D.4. The user selects a sensor by entering the sensor number. The FlexCAT confirms the selection by displaying the name and unit of the sensor. It also continuously displays voltage reading corresponding to the

selected sensor in the output field named *Signal (Volt)*. The calibration process consists of increasing (or decreasing) a *Reference* parameter (i.e., load, displacement, or pressure as the case may be) in small steps and collecting voltage readings for each increment by pressing *Read* button. The desired value of step size is entered in *Increment* input field. Whether the *Reference* value is increased or decreased is controlled by the *Direction* button. The data can be displayed in graphical form at any stage of the calibration process simply by pressing the *View* button. When sufficient numbers of readings are collected, pressing the *Compute* button performs a least square analysis on the data and displays the calibration factor and regression coefficient in a pop up window.

**3. Loop:** This command is useful in obtaining the open loop response of any loop in the system. The command displays the window shown in Figure D.5. The user selects a loop by typing the loop number and pressing return. The FlexCAT confirms the selection by displaying the loop name and the names of sensor and servo associated with that loop. The program then continuously scans the sensor input and displays the reading in the output field named *Sensor Reading*. The user enters the desired value servo output voltage in the *Servo Output* field. By clicking on the *Run* button, the user can send this voltage to the servo and monitor its effect on the sensor reading. The relationship between servo voltage and sensor reading describes the open-loop dynamics of the plant (See Appendix B). The *Stop* button sends zero voltage to the servo and stops the process instantly.

### D.3.5. Test Menu

The user can test the A/D and D/A cards, sensors, and servos through commands in this menu. The ability to test individual hardware elements helps the user to isolate a faulty component during troubleshooting.

- 1. A/D Card:** This command displays a window (Figure D.6) and continuously shows input voltage readings for the selected A/D card. The refresh rate depends on the integration time and number of active channels.
- 2. D/A Card:** This command allows the user to send any valid voltage to the selected channel of a digital-to-analog card. Any attempt to send out-of-range values generates a warning message. In conjunction with a precision voltmeter this command can be used to calibrate a D/A card (refer to Appendix C).
- 3. Sensor:** The command displays a window showing the name, input volt, unit, reading, and absolute reading for a selected sensor . It is also possible to assign the current reading as the zero or absolute zero (refer to C.3.1) or restore the previous zero value.
- 4. Servo:** This command allows the user to send voltage to a selected servo in increments of 0.1 volt. It can be used for manipulating servo devices (e.g. motor, air pressure controller etc.) under manual control.

### D.3.6. DataAcq Menu

Commands in this menu are related to the background data acquisition function of the FlexCAT.

- 1. Open:** This command is used to open an old data acquisition file created by the FlexCAT. Opening an old file loads the sensor readings along

with zeros, calibration factors, and data acquisition parameters from that file into the memory.

- 2. New:** This command is used to create a new data acquisition task. It displays a window and prompts the user to enter name of the data file, reading interval, sensor numbers, and maximum readings. The sensor numbers can be any valid numbers separated by a comma or hyphen. If two numbers are separated by a hyphen (e.g. 2-5) all intermediate values are included. A header inserted at the top of the data acquisition file (Figure D.7) contains information such as the current date and time, the maximum and actual numbers of readings, sensor numbers, names, calibration factors, absolute and floating zeros, etc.
- 3. Start:** This command starts the background data acquisition activity. The data acquisition can be started only when the system is in the *Run* mode.
- 4. Stop:** This command stops the background data acquisition. A stopped task can be restarted any time during a program session.
- 5. Interval:** The interval command allows the user to modify the values of reading interval and maximum readings for current data acquisition task.
- 6. Close:** This command closes a data acquisition task and clears all buffers. A new task cannot be started until the existing task is closed.

### D.3.7. Functions Menu

This menu is a collection of miscellaneous functions which are useful in most control and data acquisitions applications.

- 1. Zero Sensors:** This command takes a set of readings for all sensors in the system and assigns them as absolute zero of respective sensors. This command is very useful at the beginning of a test in establishing the initial conditions. The Setup>Save command should be executed to save the new zeros on the hard disk.
- 2. Stop Servos:** This command sends zero volt to all servos in the system. It is useful for stopping a runaway actuator in the case of an emergency. However, this command does not stop a servo that has the *Auto Reset* feature turned off.
- 3. View Graph:** This command displays the selected graph in a special window shown in Figure D.2. The graph window has two floating cursors which allow on screen measurements of slope and offsets. The graph can be printed in form of a Postscript<sup>®</sup> file by clicking the Print button.
- 4. Print Screen:** This command saves the screen as a Postscript<sup>®</sup> file on the hard disk. This file can later be printed by any Postscript<sup>®</sup> compatible printer.



Object Name	Database Content	Operations
System	Numbers of A/D cards, D/A cards, Sensors, Servos, Loops, and Graphs; Cycle Time	Run Stop
A/D Card	Serial number, Base address, Integration time, Calibration time, Resolution, Active channels	Calibrate Test Read
D/A Card	Serial number, Base address, Resolution, Range, Channel polarity	Test Write
Sensor	Serial number, Name, A/D card number, Channel number, Calibration factor, Absolute zero, Zero, Absolute reading, Reading, Maximum reading,	Calibrate Read Set Zero Restore Zero
Servo	Serial number, Name, D/A card number, Channel number, Maximum volt, Minimum volt, Reset Flag	Calibrate Write Reset
Loop	Serial number, Name, Sensor number, Servo number, $K_P$ , $K_I$ , $K_D$ , Cycle count	Calibrate
Data Acq	File name, Sensor numbers, Interval, Maximum readings, Readings taken	Start Stop Close
Graph	Serial number, Name, Number of plots, Number of points, Axis titles, Axis limits, Axis divisions, Log scale flags	Display

Table D.1: Properties of the FlexCAT Objects.



System Setup Calibrate Test DataAcq Functions Graph

### Active Plane Strain Soil Reinforcement

DCDT 1	DCDT 2	DCDT 3	DCDT 4	Axial Strain %	Test Mode
0.000	0.000	0.000	0.000	0.0000	<input type="checkbox"/>
Axial Stress kPa	Strain Rate mm/min.		Limits		
Plat. 1	0.000	Target	Actual	Disp.	Stress
Plat. 2	0.000	0.0000	0.0000	0.00	0.00
					<input type="checkbox"/> Run

Sidewall Pressure kPa	Reinf. Position mm.	Air Pressure kPa
Top	0.0000	Target
Bottom	Load kN/m	Actual
0.000	0.000	0.00
Plane Strain %	<input type="checkbox"/> Run	Lateral Strain %
0.0000		0.0000
b Value	Graph Selection	R Value
0.000	<input type="checkbox"/> Axial Strain-Time	0.000
<input type="checkbox"/> Run		<input type="checkbox"/> Run

Figure D.1: The APSR User Interface.



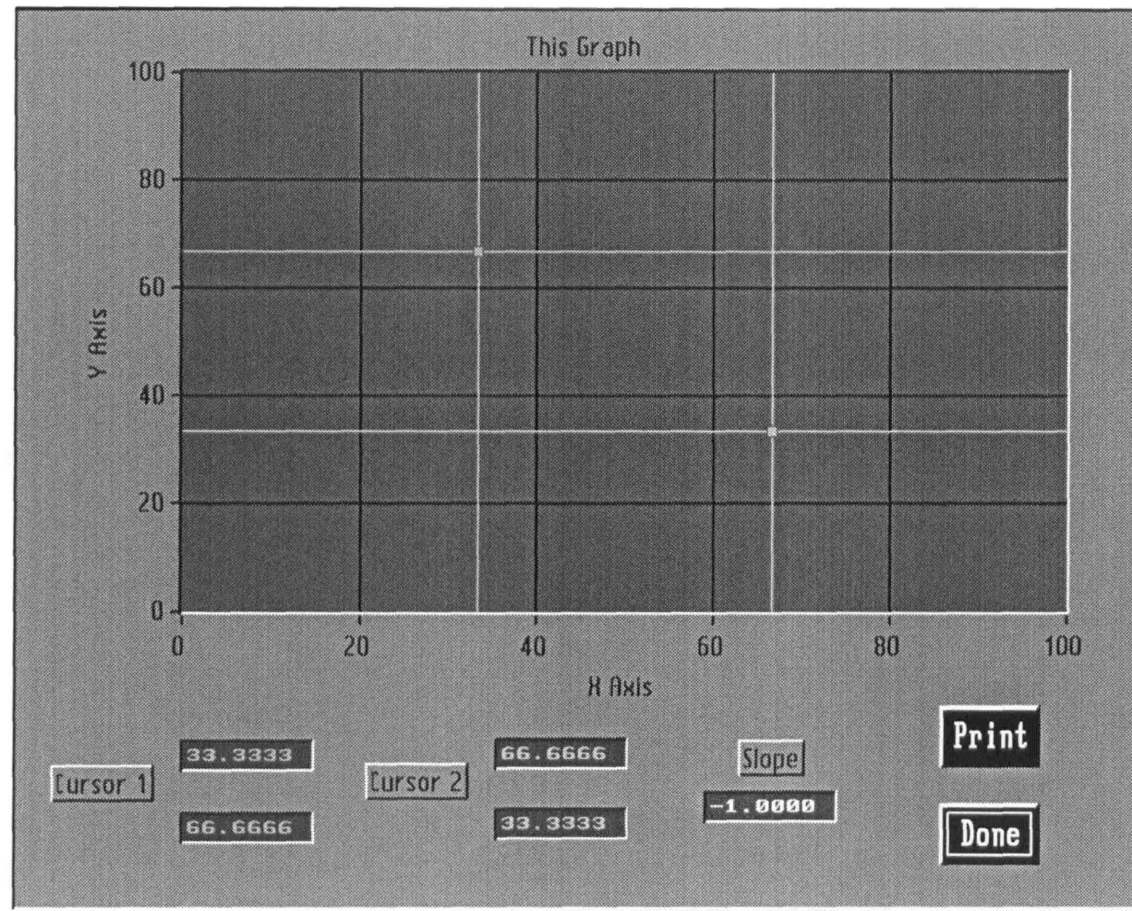
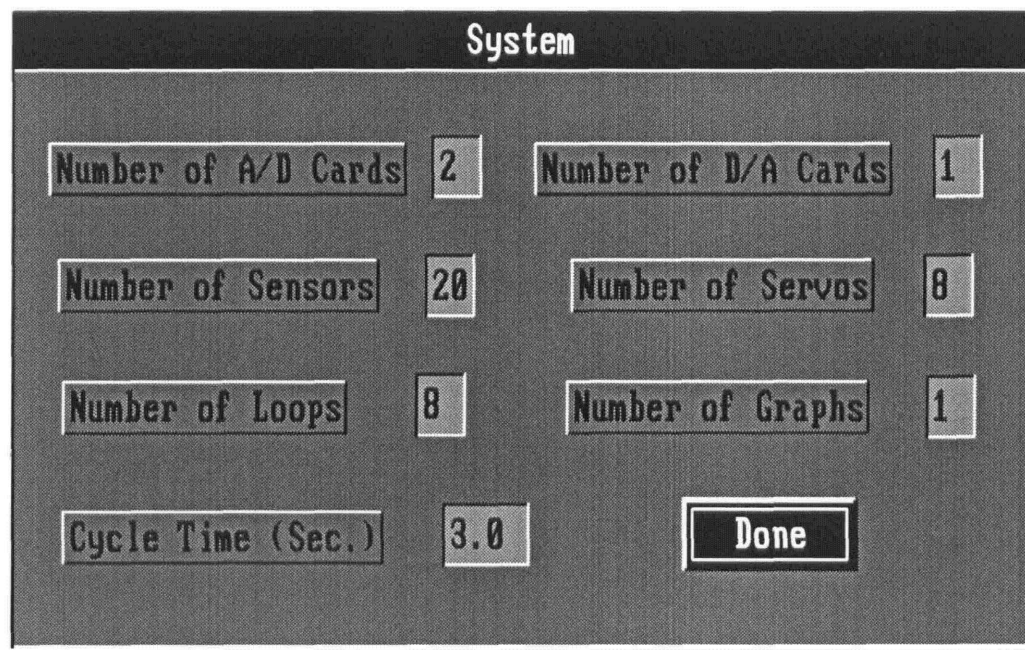


Figure D.2: The FlexCAT Graph Display Window.





The image shows a 'System' setup dialogue box with a dark background and white text. It contains several input fields for configuration parameters and a 'Done' button. The parameters are arranged in two columns. The first column includes 'Number of A/D Cards' (2), 'Number of Sensors' (20), 'Number of Loops' (8), and 'Cycle Time (Sec.)' (3.0). The second column includes 'Number of D/A Cards' (1), 'Number of Servos' (8), and 'Number of Graphs' (1). A 'Done' button is located at the bottom right of the box.

System			
Number of A/D Cards	2	Number of D/A Cards	1
Number of Sensors	20	Number of Servos	8
Number of Loops	8	Number of Graphs	1
Cycle Time (Sec.)	3.0	Done	

Figure D.3: The Layout of the System Setup Dialogue Box.





**Sensor Calibration**

Sensor number	0	Name	Air Pressure
Signal (Volt)	0.000000	Input (Volt)	5.500
Reference	50.0000	Number of readings	0
Unit	kPa	Read	Delete
Increment	20.0000	View	Compute
Direction	<input type="checkbox"/>	Done	

Figure D.4: The Sensor Calibration Window.



**Loop Calibration**

Loop number	0	Name	Piston 1
Sensor name		Servo name	
DCDT 1		Motor 1	
Sensor Reading		Servo Output (V)	
0.000000	mm.	-1.000	
Run	Stop	Done	

Figure D.5: The Loop Calibration Window and Controls.



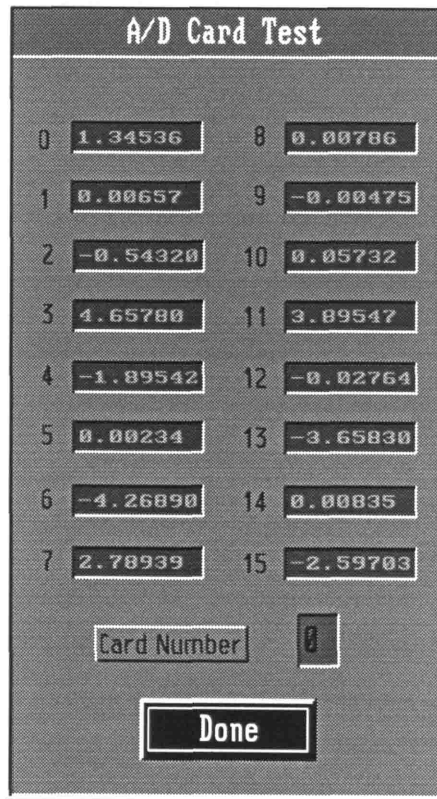


Figure D.6: The A/D Card Test Display.



```

SPC:
"trial" 3 100 60.000 3
"Mon Jul 29, 1995"
"18:16:07"
***
"Sensor num" 2 3 5
"Calib Factor" -5.446060 -5.507990 +5750.107910
"Abs Zero" +0.551233 +0.898781 +0.001059
"Float Zero" -0.287504 +0.316572 +0.001059
"Seconds" "mm." "mm." "kPa"
"Clock" "DCDT 2" "DCDT 3" "Piston Pr. 1"
0.000 -0.015507 -0.015640 +0.007906
60.000 -0.015516 -0.015631 +0.007868
120.000 -0.015612 -0.015717 +0.007839

```

Figure D.7: The FlexCAT Data Acquisition File Header.





## D.4. Program Listings

### D.4.1. File SUPPORT.H

```
/*-----  
| This header file contains all variable and function  
| declarations needed for using system file services. Do not  
| delete or modify the content of this file in any way.  
| Copyright Samir Chauhan, 1994.  
| MIT Geotechnical Laboratories.  
|-----*/
```

```
#ifndef SUPPORT_H_ /* if already included, then skip the rest */  
#define SUPPORT_H_
```

```
#include <stdio.h> /* include most commonly used header files */  
#include <dos.h>  
#include <math.h>  
#include <string.h>  
#include <time.h>
```

```
typedef struct {  
    char file_name[8];  
    int total_adcards;  
    int total_dacards;  
    int total_sensors;  
    int total_servos;  
    int total_loops;  
    int total_graphs;  
    float cycle_time;  
    } system_type;
```

```
typedef struct {  
    int serial_num;  
    char name[15];  
    int sensor_num;  
    int servo_num;  
    int cycle_count;  
    unsigned int counter;  
    float kp;  
    float kv;  
    float ki;  
    } loop_type ;
```

```
typedef struct {  
    char date_created[17];  
    char time_created[10];  
    float elapsed_time;  
    int timer_num;
```

```

    } calander_type;

extern system_type system;    /* these structures are defined in the system file */
extern loop_type loop;
extern calander_type calander;

int MeFirst(void);           /* These functions must be supplied by the user */
int MeLast(void);
int UserForeground(void);
int UserBackground(void);
int UserCompute(void);

extern int GetSystemStatus(void); /* These functions are supplied by the system file */
extern int GetEventPanel(void);
extern int GetEventControl(void);
extern float GetSystemTime(void);
extern int SetUserFlag(void);
extern int ClearUserFlag(void);
extern int IssueWarning(char *message, float interval);

extern int SensorRead(int sensor_num);
extern int SensorTakeZero(int sensor_num);
extern int SensorTakeAbsZero(int sensor_num);
extern int SensorRestoreZero(int sensor_num);
extern float SensorGetVolt(int sensor_num);
extern float SensorGetReading(int sensor_num);
extern float SensorGetAbsReading(int sensor_num);

extern int ServoWrite(int servo_num, float vout);
extern int ServoReset(int servo_num);

extern int GraphPlot(int graph_num, int plot_num, int total_points, float *x_array, float *y_array);
extern int GraphDisplay(void);

#endif /* SUPPORT_H_ */

/* end of include file */

```

## D.4.2. File SYSTEM.H

```
/*-----  
| This header file contains declarations of all FlexCAT objects.  
| In addition, it contains defines for AD1170 converter commands.  
| This file should be included at the top of FLEXCAT.C for its  
| successful compilation.  
| Copyright Samir Chauhan, 1994.  
| MIT Geotechnical Laboratories.  
-----*/
```

```
#ifndef SYSTEM_H_ /* if already included, then skip the rest */  
#define SYSTEM_H_
```

```
#include <stdio.h>  
#include <dos.h>  
#include <math.h>  
#include <string.h>  
#include <time.h>  
#include "pchrt.h"
```

```
#define MAX_ADCARD 3 /* this defines control the size of object arrays */  
#define MAX_DACARD 3  
#define MAX_SENSOR 25  
#define MAX_SERVO 10  
#define MAX_LOOP 8  
#define MAX_GRAPH 15
```

```
#define CALEN 176 /* AD1170 command set */  
#define CALDI 184  
#define CNV 8  
#define ECAL 24  
#define EOI 136  
#define RESA 104  
#define SAVA 72  
#define SDI 56  
#define SDF 48  
#define SCAL 192  
#define SDC 64  
#define WRNV 19  
#define RDNV 20  
#define DONULL 112  
#define NULEN 120  
#define NULDI 128
```

```
typedef struct { /* FlexCAT object data structure declarations */  
    char file_name[8];  
    int total_adcards;  
    int total_dacards;  
    int total_sensors;  
    int total_servos;
```

```
    int total_loops;
int total_graphs;
    float cycle_time;
} system_type;
```

```
typedef struct {
    int serial_num;
    int base_adress;
    int integration_time;
    int calibration_time;
    int resolution;
int total_channel;
    int background_cal_flag;
    int electronic_null_flag;
    int default_flag;
    double max_count;
    unsigned long time_count;
    float involt[16];
} adcard_type ;
```

```
typedef struct {
    int serial_num;
    int base_adress;
    int resolution;
    float range[8];
    int polarity_code[8];
    double max_count;
} dacard_type;
```

```
typedef struct {
    int serial_num;
    char name[15];
    char unit[10];
    int card_num;
    int channel_num;
    float calib_factor;
    float abs_zero;
    float zero;
    float max_reading;
    float volt;
    float reading;
    float abs_reading;
} sensor_type;
```

```
typedef struct {
    int serial_num;
    char name[15];
    int card_num;
```

```
    int channel_num;
int reset_flag;
    float out_volt;
    float delta_volt;
    float max_volt;
    float min_volt;
} servo_type;
```

```
typedef struct {
    int serial_num;
    char name[15];
    int sensor_num;
    int servo_num;
    int cycle_count;
    unsigned int counter;
    float kp;
    float kv;
    float ki;
} loop_type ;
```

```
typedef struct {
    char name[40];
    char x_title[30];
    char y_title[30];
    double min_x, max_x;
    double min_y, max_y;
    int x_divisions;
    int y_divisions;
    int x_logscale;
    int y_logscale;
    int total_plots;
} graph_type;
```

```
typedef struct {
    int graph_panel;
    char name[40];
    char x_title[30];
    char y_title[30];
    double min_x, max_x;
    double min_y, max_y;
    int x_divisions;
    int y_divisions;
    int x_logscale;
    int y_logscale;
    float *x_array[6];
    float *y_array[6];
    int total_plots;
    int total_points[6];
```

```

    } plot_type;

typedef struct {
    int exit_flag;
    int system_run_flag;
    int user_run_flag;
    int event_control;
    int event_panel;
    int thread_num[8];
    int main_menu;
    } control_type;

typedef struct {
    char file_name[15];
    FILE *file_handle;
    char sensor_string[30];
    int sensor_num[25];
    int total_sensors;
    float interval;
    int readings_taken;
    int max_readings;
    int setup_flag;
    int run_flag;
    } data_acq_type;

typedef struct {
    float slope;
    float intercept;
    float r_squared;
    } result_type;

typedef struct {
    int error_flag;
    int error_code;
    } error_type;

typedef struct {
    char date_created[17];
    char time_created[10];
    float elapsed_time;
    int timer_num;
    } calander_type;

#endif /* SYSTEM_H_ */

/* end of include file */

```

### D.4.3. File FLEXCAT.C

```
/*-----  
| This is System module. It provides all data management and  
| basic functions required for writing a control and data  
| acquisition application.  
| Version 1.2.2 Last update Nov 30, 1994.  
| Copy right Samir Chauhan, MIT Geotechnical Laboratory, 1994.  
|-----*/
```

```
#include "system.h"  
#include "flexcat.h"  
  
system_type system;          /* global structures and variables */  
static adcard_type adcard[MAX_ADCARD];  
static dacard_type dacard[MAX_DACARD];  
static sensor_type sensor[MAX_SENSOR];  
static servo_type servo[MAX_SERVO];  
loop_type loop[MAX_LOOP];  
static graph_type graph[MAX_GRAPH];  
static plot_type plot;  
static control_type control;  
calander_type calander;  
static data_acq_type data_acq;  
static error_type error;  
  
static void interrupt (*Old_1C_Vector)(void);  
static void interrupt (*Old_45_Vector)(void);  
  
extern int MeFirst(void); /* All functions defined in the user module are declared here */  
extern int MeLast(void);  
extern int UserForeground(void);  
extern int UserBackground(void);  
extern int UserCompute(void);  
  
static int Initialize(void); /* All functions used in this module are declared here */  
static int VariableInitialize(void);  
static int Terminate(void);  
  
static int ProcessMenu(int ctrl_event);  
static int ProcessGlobalErrors(void);  
  
static int SystemSetup(void);  
static int SystemMessage(float interval);  
static int SystemRun(void);  
static int SystemStop(void);  
static int SystemExit(void);  
static void RunBackground(void);  
static int SetupPrint(void);  
  
static int AdcardSetup(void);  
static int AdcardInitialize(int card_num);
```

```

static int AdcardCalibrate(void);
static int SingleCalibration(int card_num);
static int OffsetCorrection(int card_num);
static int GainCorrection(int card_num);
static int SaveAll(int card_num);
static int AdcardTest(void);

static int DacardSetup(void);
static int DacardTest(void);

static int SensorSetup(void);
static int SensorTest(void);
static int SensorCalibrate(void);
static int LinearRegression(float *x, float *y, int n, result_type *result);
int SensorRead(int sensor_num);
int SensorTakeZero(int sensor_num);
int SensorTakeAbsZero(int sensor_num);
int SensorRestoreZero(int sensor_num);
int SensorZeroAll(void);
float SensorGetVolt(int sensor_num);
float SensorGetReading(int sensor_num);
float SensorGetAbsReading(int sensor_num);

static int ServoSetup(void);
static int ServoTest(void);
int ServoWrite(int servo_num, float volt);
int ServoReset(int servo_num);
int ServoResetAll(void);
static int LoopSetup(void);
static int LoopCalibrate(void);

static int GraphSetup(void);
int GraphDisplay(void);
int GraphPlot(int graph_num, int plot_num, int total_points, float *x_array, float *y_array);
float GetSystemTime(void);
static int GraphPrint(void);
static int ScreenPrint(void);

static int DataAcqSetup(void);
static int DataAcqOpen(void);
static int DataAcqRun(void);
static int DataAcqStop(void);
static int DataAcqInterval(void);
static int DataAcqClose(void);
static void AcquireData(void);

static int DatabaseLoad(void);
static int DatabaseSave(void);
static int InitializeTime(void);
static int Wait(int adress, unsigned char i, unsigned char j);
static void LWKbdHandler(void);
static float AdcardRead(int card_num, int channel_num);

```



```

static void AdcardDrive0(void);
static void AdcardDrive1(void);
static void AdcardDrive2(void);
static int DacardWrite(int card_num, int channel_num, float volt);

int SetUserFlag(void);
int ClearUserFlag(void);
int GetSystemStatus(void);
int GetEventPanel(void);
int GetEventControl(void);
int IssueWarning(char *message, float interval);

int main(void) /* This is Main Function */
{

    Initialize(); /* This function performs all operations related to
                  the initialization of the program */

    while(!control.exit_flag) { /* Enter an infinite loop and wait for the user input */

        GetUserEvent(0, &control.event_panel, &control.event_control); /* Get event from event queue
*/

        if(control.event_panel == control.main_menu) /* Process the menu event */
            ProcessMenu(control.event_control);

        UserForeground(); /* perform user's foreground tasks */

        ProcessGlobalErrors(); /* check if any global error has occurred */
    } /* Infinite loop ends here */

    Terminate(); /* This function performs all house cleaning operations
                  related to termination of the program */

    return 0;
} /* Function Main ends here */

/*-----
| This function performs all initialization routines associated |
| with the program |
|-----*/
int Initialize(void)
{
    int i, card_num;
    int istat, status;
    float cycletime, interval;

```

```
control.main_menu = LoadMenuBar("FLEXCAT.UIR", MENU); /* Load and display the menu
bar */
```

```
Old_1C_Vector = getvect(0x1C); /* swap the LabWindow key board handler interrupt handler
vector */
```

```
Old_45_Vector = getvect(0x45);
setvect(0x45, Old_1C_Vector);
```

```
istat = t_start(); /* initialize the timer system kernal */
if(!istat) {
    IssueWarning("Failed to initialize timer Kernal", -1.0);
    return 1;
}
```

```
t_sched_init(); /* initialize the thread scheduler */
```

```
calander.timer_num = t_alloc(T_MSEC, " "); /* allocate the main timer */
if(calander.timer_num == T_FAIL)
    IssueWarning("Unable to allocate main timer.", -1.0);
t_entry(calander.timer_num); /* start the main timer */
```

```
status = DatabaseLoad(); /* load the data base from hard disk */
```

```
if(status == 1) /* if the data file is missing */
    IssueWarning("Data file is missing, Create new database.", -1.0);
```

```
VariableInitialize(); /* inititalize various global structures */
```

```
/* allocates scheduler thread for keyboard handler task */
control.thread_num[7] = t_sched_set(LWKbdHandler, 150L, T_RESTART);
if(control.thread_num[7] == T_FAIL) /* check if allocation failed for the thread */
    IssueWarning("Unable to schedule a thread", -1.0);
```

```
t_sched_enable(); /* start running scheduler */
```

```
ServoResetAll(); /* stop all eligibles servos */
```

```
MeFirst(); /* User's initializing tasks */
```

```
/* finally hang our favorite message and relax! */
SystemMessage(3.0);
```

```
return 0;
} /* function Initialize end here */
```

```
/*-----
| This function initialize the different structures with |
| default values of parameters and puts them in a known status. |
```

```

-----*/
int VariableInitialize(void)
{
    int i;

    control.exit_flag = 0;          /* initialize control structure */
    control.system_run_flag = 0;
    control.user_run_flag = 0;
    control.event_control = -1;
    control.event_panel = -1;
    for(i=0; i<8; i++)
        control.thread_num[i] = 0;

    error.error_flag = 0;          /* initialize error structure */
    error.error_code = 0;

    strcpy(plot.name, "My Graph"); /* initialize plot structure */
    strcpy(plot.x_title, "My X_title");
    strcpy(plot.y_title, "My Y_title");
    plot.min_x = plot.min_y = 0.0;
    plot.max_x = plot.max_y = 100.0;
    plot.x_logscale = plot.y_logscale = 0;
    plot.total_plots = 0;

    for(i=0; i<6; i++) {          /* direct array pointers to NULL location */
        plot.x_array[i] = NULL;
        plot.y_array[i] = NULL;
        plot.total_points[i] = 0;
    }

    return 0;
} /* function VariableInitialize ends here */

```

```

-----*/
| This function performs all house cleaning tasks associated with |
| with the termination of the program                             |
-----*/

```

```

int Terminate(void)
{
    int status;

    MeLast(); /* first run user's house cleaning tasks */
    UnloadMenuBar(control.main_menu); /* unload the menu bar */

    t_sched_off(); /* shut down the scheduler */
    t_exit(calander.timer_num); /* stop the main timer */
    t_free(calander.timer_num); /* deallocate the main timer */
}

```

```

t_stop();          /* shut down the timer kernel */

ServoResetAll();  /* stop all eligibles servos */

setvect(0x45, Old_45_Vector); /* restore old interrupt vector 45 */

return 0;

} /* function Terminate end here */

```

```

/*-----
| This function processes the menu commands and calls |
| appropriate functions depending on the value of ctrl_event |
|-----*/

```

```

int ProcessMenu(int ctrl_event)
{
    int status;

    /* this function is just one big switch case statement */
    switch (ctrl_event) {
        case MENU_SYS_RUN:
            SystemRun();
            break;
        case MENU_SYS_STOP:
            SystemStop();
            break;
        case MENU_SYS_ABO:
            SystemMessage(-1.0);
            break;
        case MENU_SYS_EXIT:
            SystemExit();
            break;
        case MENU_SETUP_SYS:
            SystemSetup();
            break;
        case MENU_SETUP_ADC:
            AdcardSetup();
            break;
        case MENU_SETUP_DAC:
            DacardSetup();
            break;
        case MENU_SETUP_SEN:
            SensorSetup();
            break;
        case MENU_SETUP_SER:
            ServoSetup();
            break;
        case MENU_SETUP_LOOP:
            LoopSetup();

```

```
break;
case MENU_SETUP_GRAF:
    GraphSetup();
break;
case MENU_SETUP_PRN:
    SetupPrint();
break;
case MENU_SETUP_SAVE:
    DatabaseSave();
break;
case MENU_CAL_ADC:
    AdcardCalibrate();
break;
case MENU_CAL_SEN:
    SensorCalibrate();
break;
case MENU_CAL_LOOP:
    LoopCalibrate();
break;
case MENU_TEST_ADC:
    AdcardTest();
break;
case MENU_TEST_DAC:
    DacardTest();
break;
case MENU_TEST_SEN:
    SensorTest();
break;
case MENU_TEST_SER:
    ServoTest();
break;
case MENU_DATA_NEW:
    DataAcqSetup();
break;
case MENU_DATA_OPEN:
    DataAcqOpen();
break;
case MENU_DATA_STAR:
    DataAcqRun();
break;
case MENU_DATA_STOP:
    DataAcqStop();
break;
case MENU_DATA_INT:
    DataAcqInterval();
break;
case MENU_DATA_CLO:
    DataAcqClose();
break;
case MENU_FUNC_ZERO:
    SensorZeroAll();
break;
```

```

    case MENU_FUNC_STOP:
        ServoResetAll();
    break;
    case MENU_FUNC_VIEW:
        GraphDisplay();
    break;
    case MENU_FUNC_PRSCR:
        ScreenPrint();
    break;
}
return 0;
} /* function ProcessMenu ends here */

```

```

int SetupPrint(void)
{
    int ctrl_event = -1, exit_flag = 0;
    int pan_handle, status, i, j;
    char file_name[9], temp[15], setting[5], polarity[10];
    char *marker = "*****"
        "*****";

    FILE *file_handle;
    float interval;

    pan_handle = LoadPanel("FLEXCAT.UIR", SPRINT);
    Instal!Popup(pan_handle);

    while(!exit_flag)
    {
        GetCtrlVal(pan_handle, SPRINT_NAME, file_name);

        GetPopupEvent(0,&ctrl_event);
        if(ctrl_event == SPRINT_DONE)
            exit_flag = 1;
        if(ctrl_event == SPRINT_CAN) {
            RemovePopup(0);
            UnloadPanel(pan_handle);
            return 1;
        }
    }

    RemovePopup(0);
    UnloadPanel(pan_handle);

    /* open a new setup print file */
    strcpy(temp, file_name);      /* attach .prn extension */
    strcat(temp, ".prn");
    file_handle = fopen(temp, "w+t");
    if(file_handle == NULL) {

```

```

IssueWarning("Error opening set up print file.", 2.0);
return 1;
}

fprintf(file_handle, "\n%s\n", marker);
fprintf(file_handle, "%s\n", "SYSTEM");
fprintf(file_handle, "%s\n", marker);
fprintf(file_handle, "%40s %5d\n", "Number of Analog to Digital cards -", system.total_adcards);
fprintf(file_handle, "%40s %5d\n", "Number of Digital to Analog cards -", system.total_dacards);
fprintf(file_handle, "%40s %5d\n", "Number of Sensors -", system.total_sensors);
fprintf(file_handle, "%40s %5d\n", "Number of Servos -", system.total_servos);
fprintf(file_handle, "%40s %5.2f\n\n", "System cycle time (seconds) -", system.cycle_time);

fprintf(file_handle, "\n%s\n", marker);
fprintf(file_handle, "%s\n", "A/D CARDS");
fprintf(file_handle, "%s\n", marker);
for(i=0; i<system.total_adcards; i++) {
    fprintf(file_handle, "%s %d\n", "A/D card Number -", i);
    fprintf(file_handle, "%40s %5d\n", "Base adress -", adcard[i].base_adress);
    switch (adcard[i].integration_time) {
    case 0:
        interval = 10;
        break;
    case 1:
        interval = 16.7;
        break;
    case 2:
        interval = 20;
        break;
    case 3:
        interval = 100;
        break;
    case 4:
        interval = 166.7;
        break;
    case 5:
        interval = 300;
        break;
    }
    fprintf(file_handle, "%40s %5.1f\n", "Integration time (ms.) -", interval);

    switch (adcard[i].calibration_time) {
    case 0:
        interval = 10;
        break;
    case 1:
        interval = 16.7;
        break;
    case 2:
        interval = 20;
        break;
    case 3:

```

```

        interval = 100;
    break;
    case 4:
        interval = 166.7;
    break;
    case 5:
        interval = 300;
    break;
}
fprintf(file_handle, "%40s %5.1f\n", "Calibration time (ms.) -", interval);
fprintf(file_handle, "%40s %5d\n", "Bit resolution -", adcard[i].resolution + 8);
if(adcard[i].background_cal_flag == 1)
    strcpy(setting, "ON");
else
    strcpy(setting, "OFF");
fprintf(file_handle, "%40s %5s\n", "Background calibration -", setting);
if(adcard[i].electronic_null_flag == 1)
    strcpy(setting, "ON");
else
    strcpy(setting, "OFF");
fprintf(file_handle, "%40s %5s\n", "Electronic null -", setting);
}

fprintf(file_handle, "\n%s\n", marker);
fprintf(file_handle, "%s\n", "D/A CARDS");
fprintf(file_handle, "%s\n", marker);
for(i=0; i<system.total_dacards; i++) {
    fprintf(file_handle, "%s %d\n", "D/A card Number -", i);
    fprintf(file_handle, "%40s %5d\n", "Base adress -", dacard[i].base_adress);
    fprintf(file_handle, "%40s %5d\n", "Bit resolution -", dacard[i].resolution);
    fprintf(file_handle, "%40s %s %s\n", "Channel No.", "Range (V)", "Polarity");
    for(j=0; j<8; j++) {
        if(dacard[i].polarity_code[j] == 1)
            strcpy(polarity, "Unipolar");
        else
            strcpy(polarity, "Bipolar");
        fprintf(file_handle, "%35d %5.1f %5s\n", j, dacard[i].range[j], polarity);
    }
}
fprintf(file_handle, "%s\n", " ");

fprintf(file_handle, "\n%s\n", marker);
fprintf(file_handle, "%s\n", "SENSORS");
fprintf(file_handle, "%s\n", marker);
for(i=0; i<system.total_sensors; i++) {
    fprintf(file_handle, "%s %d\n", "Sensor Number -", i);
    fprintf(file_handle, "%40s %s\n", "Sensor name -", sensor[i].name);
    fprintf(file_handle, "%40s %s\n", "Sensor unit -", sensor[i].unit);
    fprintf(file_handle, "%40s %5d\n", "Card number -", sensor[i].card_num);
    fprintf(file_handle, "%40s %5d\n", "Channel Number -", sensor[i].channel_num);
    fprintf(file_handle, "%40s %15.5f\n", "Calibration factor -", sensor[i].calib_factor);
    fprintf(file_handle, "%40s %10.f\n", "Maximum Reading -", sensor[i].max_reading);
}

```



```

}
fprintf(file_handle, "%s\n", " ");

fprintf(file_handle, "\n%s\n", marker);
fprintf(file_handle, "%s\n", "SERVOS");
fprintf(file_handle, "%s\n", marker);
for(i=0; i<system.total_servos; i++) {
    fprintf(file_handle, "%s %d\n", "Servo Number -", i);
    fprintf(file_handle, "%40s %s\n", "Servo name -", servo[i].name);
    fprintf(file_handle, "%40s %-5d\n", "Card number -", servo[i].card_num);
    fprintf(file_handle, "%40s %-5d\n", "Channel number -", servo[i].channel_num);
    fprintf(file_handle, "%40s %-5.1f\n", "Minimum volt -", servo[i].min_volt);
    fprintf(file_handle, "%40s %-5.1f\n", "Maximum volt -", servo[i].max_volt);
}
fprintf(file_handle, "%s\n", " ");

fprintf(file_handle, "\n%s\n", marker);
fprintf(file_handle, "%s\n", "LOOPS");
fprintf(file_handle, "%s\n", marker);
for(i=0; i<system.total_loops; i++) {
    fprintf(file_handle, "%s %d\n", "Loop Number -", i);
    fprintf(file_handle, "%40s %-5d\n", "Sensor number -", loop[i].sensor_num);
    fprintf(file_handle, "%40s %-5d\n", "Servo number -", loop[i].servo_num);
    fprintf(file_handle, "%40s %-5d\n", "Cycle count -", loop[i].cycle_count);
    fprintf(file_handle, "%40s %-15.5f\n", "Praportional constant -", loop[i].kp);
    fprintf(file_handle, "%40s %-15.5f\n", "Derivative constant -", loop[i].kv);
    fprintf(file_handle, "%40s %-15.5f\n", "Integration constant -", loop[i].ki);
}

fclose(file_handle);

return 0;
} /* Function SetupPrint ends here */

```

```

/*-----
| This function provides a way for functions called in background |
| to convey occurrence of an error condition. |
-----*/

```

```

int ProcessGlobalErrors(void)
{
    char message[80], temp[17];

    switch (error.error_flag) {
        case 1:
            IssueWarning("Attempt to read a nonexistent sensor.", 1.5);
            break;
        case 2:

```

```

        strcpy(message, "Sensor limit exceeds for sensor ");
        itoa(error.error_code, temp, 10);
        strcat(message, temp);
        IssueWarning(message, 1.5);
        break;
    case 3:
        IssueWarning("Attempt to write to a nonexistent servo.", 1.5);
        break;
    case 4:
        strcpy(message, "Attempt to write invalid volt to servo ");
        itoa(error.error_code, temp, 10);
        strcat(message, temp);
        IssueWarning(message, 1.5);
        break;
    case 5:
        DataAcqStop(); /* stop data acquisition task */
        IssueWarning("Maximum number of readings reached in data acq.", -1.0);
        break;
    case 6:
        IssueWarning("Error occured in User background process. ", -1.0);
        break;
} /* switch statement ends here */

error.error_flag = 0; /* clear the error flag so that the same message do not reappear */

return 0;
} /* function ProcessGlobalErrors ends here */

```

```

/*-----
| This function displays a window and obtzins from user the |
| system parameters stored in structure system. |
|-----*/

```

```

int SystemSetup(void)
{
    int ctrl_event = -1, exit_flag = 0;
    int pan_handle, status;

    pan_handle = LoadPanel("FLEXCAT.UIR", SYS);
    SetCtrlVal(pan_handle, SYS_NUMADC, system.total_adcards);
    SetCtrlVal(pan_handle, SYS_NUMDAC, system.total_dacards);
    SetCtrlVal(pan_handle, SYS_NUMSEN, system.total_sensors);
    SetCtrlVal(pan_handle, SYS_NUMSER, system.total_servos);
    SetCtrlVal(pan_handle, SYS_NUMLOOP, system.total_loops);
    SetCtrlVal(pan_handle, SYS_NUMGRAF, system.total_graphs);
    SetCtrlVal(pan_handle, SYS_FRQ, system.cycle_time);
    SetCtrlAttribute(pan_handle, SYS_NUMADC, 13, MAX_ADCARD);
    SetCtrlAttribute(pan_handle, SYS_NUMDAC, 13, MAX_DACARD);
    SetCtrlAttribute(pan_handle, SYS_NUMSEN, 13, MAX_SENSOR);
    SetCtrlAttribute(pan_handle, SYS_NUMSER, 13, MAX_SERVO);

```

```

SetCtrlAttribute(pan_handle, SYS_NUMLOOP, 13, MAX_LOOP);
SetCtrlAttribute(pan_handle, SYS_NUMLOOP, 13, MAX_GRAPH);

InstallPopup(pan_handle);
while(!exit_flag)
{
    GetPopupEvent(0,&ctrl_event);
    status = GetCtrlVal(pan_handle, SYS_NUMADC, &system.total_adcads);
    if(status == -14) {
        IssueWarning("Maximum number of A/D cards allowed is 3", 2.0);
        SetCtrlVal(pan_handle, SYS_NUMADC, 1);
    }
    status = GetCtrlVal(pan_handle, SYS_NUMDAC, &system.total_dacads);
    if(status == -14) {
        IssueWarning("Maximum number of D/A cards allowed is 3", 2.0);
        SetCtrlVal(pan_handle, SYS_NUMDAC, 1);
    }
    status = GetCtrlVal(pan_handle, SYS_NUMSEN, &system.total_sensors);
    if(status == -14) {
        IssueWarning("Maximum number of sensors allowed is 25", 2.0);
        SetCtrlVal(pan_handle, SYS_NUMSEN, 1);
    }
    status = GetCtrlVal(pan_handle, SYS_NUMSER, &system.total_servos);
    if(status == -14) {
        IssueWarning("Maximum number of servos allowed is 10", 2.0);
        SetCtrlVal(pan_handle, SYS_NUMSER, 1);
    }
    status = GetCtrlVal(pan_handle, SYS_NUMLOOP, &system.total_loops);
    if(status == -14) {
        IssueWarning("Maximum number of loops allowed is 8", 2.0);
        SetCtrlVal(pan_handle, SYS_NUMLOOP, 1);
    }
    status = GetCtrlVal(pan_handle, SYS_NUMGRAF, &system.total_graphs);
    if(status == -14) {
        IssueWarning("Maximum number of graphs allowed is 15", 2.0);
        SetCtrlVal(pan_handle, SYS_NUMGRAF, 1);
    }
    status = GetCtrlVal(pan_handle, SYS_FRQ, &system.cycle_time);
    if(status == -14) {
        IssueWarning("Invalid interval value", 2.0);
        SetCtrlVal(pan_handle, SYS_FRQ, 1);
    }
    if(ctrl_event == SYS_DONE)
        exit_flag = 1;
} /* while loop ends here */

RemovePopup(0);
UnloadPanel(pan_handle);
return 0;
} /* Function SystemSetup ends here */

```

```

/*-----
| This function displays a window and writes a message about
| the program at the beginning and whenever system.about menu
| is selected.
|-----*/

```

```

int SystemMessage(float interval)
{
    int ctrl_event = -1, exit_flag = 0;
    int pan_handle, status;
    float init_time;
    char *message = "\n"
        "      Welcome to Flex-CAT.      \n"
        "    A Flexible, Integrated Software \n"
        "  Framework for Computer Aided Testing.\n"
        "    Design and Implementation by \n"
        "      Samir Chauhan,      \n"
        "    MIT Geotechnical Laboratories. \n"
        "      Version 1.2, 1994.      ";

    pan_handle = LoadPanel("FLEXCAT.UIR", INITIAL);
    SetCtrlVal(pan_handle, INITIAL_TEXT, message);
    InstallPopup(pan_handle);
    init_time = GetSystemTime();

    while(!exit_flag) {
        if(interval >= 0.0)
            if((GetSystemTime() - init_time) > interval)
                exit_flag = 1;
        GetPopupEvent(0,&ctrl_event);
        if(ctrl_event == INITIAL_OK)
            exit_flag = 1;
    }
    RemovePopup(0);
    UnloadPanel(pan_handle);

    return 0;
} /* function SystemMessage ends here */

```

```

/*-----
| This function put the system in run mode. It enables the user
| background task and the tasks driving A/D cards.
|-----*/

```

```

int SystemRun()
{
    int status, t_status = 0, i;
    float cycletime;

```

```

/* depending on the number of A/d cards present, restart driver tasks for A/D cards */
if(system.total_adcards > 0) {
    status = AdcardInitialize(0);
    control.thread_num[0] = t_sched_set(AdcardDrive0, adcard[0].time_count, T_RESTART);
    t_status = control.thread_num[0];
}

if(system.total_adcards > 1) {
    status = AdcardInitialize(1);
    control.thread_num[1] = t_sched_set(AdcardDrive1, adcard[1].time_count, T_RESTART);
    t_status = control.thread_num[1];
}

if(system.total_adcards > 2) {
    status = AdcardInitialize(2);
    control.thread_num[2] = t_sched_set(AdcardDrive2, adcard[2].time_count, T_RESTART);
    t_status = control.thread_num[2];
}

if(status == 1) {
    IssueWarning("Unable to put the system in run mode.", -1.0);
    return 1;
}

/* compute system cycle time in milli seconds and start the user background tasks */
cycletime = system.cycle_time*1000.0;
control.thread_num[3] = t_sched_set(RunBackground, (unsigned long) cycletime, T_RESTART);

for(i=0; i<4; i++) { /* check if all threads have been allocated */
    t_status = control.thread_num[i];
}
if(t_status == T_FAIL) {
    IssueWarning("Unable to schedule a thread.", -1.0);
    return 2;
}

control.system_run_flag = 1; /* set the system run flag */

SetMenuBarAttribute(MENU_SETUP_SYS, 6, 0); /* toggle menu items */
SetMenuBarAttribute(MENU_SETUP_ADC, 6, 0);
SetMenuBarAttribute(MENU_SETUP_DAC, 6, 0);
SetMenuBarAttribute(MENU_SYS_RUN, 6, 0);
SetMenuBarAttribute(MENU_SYS_STOP, 6, 1);
SetMenuBarAttribute(MENU_CAL_ADC, 6, 0);
SetMenuBarAttribute(MENU_CAL_SEN, 6, 1);
SetMenuBarAttribute(MENU_CAL_LOOP, 6, 1);
SetMenuBarAttribute(MENU_TEST_ADC, 6, 1);
SetMenuBarAttribute(MENU_TEST_SEN, 6, 1);
SetMenuBarAttribute(MENU_SYS_EXIT, 6, 0);
SetMenuBarAttribute(MENU_FUNC_ZERO, 6, 1);

return 0;

```

```
} /* function SystemRun ends here */
```

```
/*-----  
| This function reads all sensors in the systems and executes |  
| all user background tasks at every system.cycle_time.      |  
-----*/
```

```
void RunBackground(void)
```

```
{
```

```
    int i, status;
```

```
    for(i=0; i<system.total_sensors; i++) /* read all sensors in the system */  
        SensorRead(i);
```

```
    status = UserBackground();          /* run user's background tasks */
```

```
    if(status != 0)
```

```
        error.error_flag = 6;          /* raise a global error */
```

```
    return;
```

```
} /* function RunBackground ends here */
```

```
/*-----  
| This function put the system in stop mode. It disables the |  
| user background task and the tasks driving A/D cards.      |  
-----*/
```

```
int SystemStop()
```

```
{
```

```
    /* if the data acquisition task is running */
```

```
    if(data_acq.run_flag == 1) {
```

```
        IssueWarning("Data acquisition task is still running !", 2.0);
```

```
        return 1;
```

```
    }
```

```
    /* if(control.user_run_flag !=0 ) {
```

```
        IssueWarning("User task(s) still running !", 2.0);
```

```
        return 1;
```

```
    } */
```

```
    t_sched_delete(control.thread_num[3]); /* delete user background task */
```

```
    if(system.total_adcards>0) /* delete A/D card driver threads*/
```

```
        t_sched_delete(control.thread_num.[0]);
```

```
    if(system.total_adcards>1)
```

```
        t_sched_delete(control.thread_num[1]);
```

```
    if(system.total_adcards>2)
```

```
        t_sched_delete(control.thread_num[2]);
```

```

control.system_run_flag = 0;    /* clear system run flag */

SetMenuBarAttribute(MENU_SETUP_SYS, 6, 1);    /* toggle menu items */
SetMenuBarAttribute(MENU_SETUP_ADC, 6, 1);
SetMenuBarAttribute(MENU_SETUP_DAC, 6, 1);
SetMenuBarAttribute(MENU_SYS_RUN, 6, 1);
SetMenuBarAttribute(MENU_SYS_STOP, 6, 0);
SetMenuBarAttribute(MENU_CAL_ADC, 6, 1);
SetMenuBarAttribute(MENU_CAL_SEN, 6, 0);
SetMenuBarAttribute(MENU_CAL_LOOP, 6, 0);
SetMenuBarAttribute(MENU_TEST_ADC, 6, 0);
SetMenuBarAttribute(MENU_TEST_SEN, 6, 0);
SetMenuBarAttribute(MENU_SYS_EXIT, 6, 1);
SetMenuBarAttribute(MENU_FUNC_ZERO, 6, 0);

return 0;

} /* function SystemStop ends here */

```

```

int SystemExit(void)
{
    if(data_acq.setup_flag == 1) /* if data acq task is still open */
        DataAcqClose();

    control.exit_flag = 1;    /* set exit flag */

    return 0;
}

```

```

int GraphSetup(void)
{
    int ctrl_event = -1, exit_flag = 0, i = 0;
    int pan_handle, max_num, status;

    max_num = system.total_graphs;

    pan_handle = LoadPanel("FLEXCAT.UIR", GRSET);
    SetCtrlVal(pan_handle, GRSET_SERNUM, i);
    SetCtrlVal(pan_handle, GRSET_NAME, graph[i].name);
    SetCtrlVal(pan_handle, GRSET_XTIT, graph[i].x_title);
    SetCtrlVal(pan_handle, GRSET_YTIT, graph[i].y_title);
    SetCtrlVal(pan_handle, GRSET_XLOG, graph[i].x_logscale);
    SetCtrlVal(pan_handle, GRSET_YLOG, graph[i].y_logscale);
    SetCtrlVal(pan_handle, GRSET_XMIN, graph[i].min_x);
    SetCtrlVal(pan_handle, GRSET_XMAX, graph[i].max_x);
}

```

```

SetCtrlVal(pan_handle, GRSET_YMIN, graph[i].min_y);
SetCtrlVal(pan_handle, GRSET_YMAX, graph[i].max_y);
SetCtrlVal(pan_handle, GRSET_XDIV, graph[i].x_divisions);
SetCtrlVal(pan_handle, GRSET_YDIV, graph[i].y_divisions);
SetCtrlVal(pan_handle, GRSET_NUMPLOT, graph[i].total_plots);

if(max_num == 1)
    SetCtrlAttribute(pan_handle, GRSET_NEXT, 15, 0);
InstallPopup(pan_handle);

while(!exit_flag) {

    GetCtrlVal(pan_handle, GRSET_NAME, graph[i].name);
    GetCtrlVal(pan_handle, GRSET_XTIT, graph[i].x_title);
    GetCtrlVal(pan_handle, GRSET_YTIT, graph[i].y_title);
    GetCtrlVal(pan_handle, GRSET_XLOG, &graph[i].x_logscale);
    GetCtrlVal(pan_handle, GRSET_YLOG, &graph[i].y_logscale);
    GetCtrlVal(pan_handle, GRSET_XMIN, &graph[i].min_x);
    GetCtrlVal(pan_handle, GRSET_XMAX, &graph[i].max_x);
    GetCtrlVal(pan_handle, GRSET_YMIN, &graph[i].min_y);
    GetCtrlVal(pan_handle, GRSET_YMAX, &graph[i].max_y);
    GetCtrlVal(pan_handle, GRSET_XDIV, &graph[i].x_divisions);
    GetCtrlVal(pan_handle, GRSET_YDIV, &graph[i].y_divisions);
    GetCtrlVal(pan_handle, GRSET_NUMPLOT, &graph[i].total_plots);
    GetPopupEvent(0, &ctrl_event);

    switch (ctrl_event) {
        case GRSET_NEXT:
            i = i+1;
            SetCtrlVal(pan_handle, GRSET_SERNUM, i);
            SetCtrlVal(pan_handle, GRSET_NAME, graph[i].name);
            SetCtrlVal(pan_handle, GRSET_XTIT, graph[i].x_title);
            SetCtrlVal(pan_handle, GRSET_YTIT, graph[i].y_title);
            SetCtrlVal(pan_handle, GRSET_XLOG, graph[i].x_logscale);
            SetCtrlVal(pan_handle, GRSET_YLOG, graph[i].y_logscale);
            SetCtrlVal(pan_handle, GRSET_XMIN, graph[i].min_x);
            SetCtrlVal(pan_handle, GRSET_XMAX, graph[i].max_x);
            SetCtrlVal(pan_handle, GRSET_YMIN, graph[i].min_y);
            SetCtrlVal(pan_handle, GRSET_YMAX, graph[i].max_y);
            SetCtrlVal(pan_handle, GRSET_XDIV, graph[i].x_divisions);
            SetCtrlVal(pan_handle, GRSET_YDIV, graph[i].y_divisions);
            SetCtrlVal(pan_handle, GRSET_NUMPLOT, graph[i].total_plots);
            if (i>0)
                SetCtrlAttribute(pan_handle, GRSET_PREV, 15, 1);
            if (i >= max_num - 1)
                SetCtrlAttribute(pan_handle, GRSET_NEXT, 15, 0);
            break;
        case GRSET_PREV:
            i = i-1;
            SetCtrlVal(pan_handle, GRSET_SERNUM, i);
            SetCtrlVal(pan_handle, GRSET_NAME, graph[i].name);
            SetCtrlVal(pan_handle, GRSET_XTIT, graph[i].x_title);

```



```

SetCtrlVal(pan_handle, GRSET_YTIT, graph[i].y_title);
SetCtrlVal(pan_handle, GRSET_XLOG, graph[i].x_logscale);
SetCtrlVal(pan_handle, GRSET_YLOG, graph[i].y_logscale);
SetCtrlVal(pan_handle, GRSET_XMIN, graph[i].min_x);
SetCtrlVal(pan_handle, GRSET_XMAX, graph[i].max_x);
SetCtrlVal(pan_handle, GRSET_YMIN, graph[i].min_y);
SetCtrlVal(pan_handle, GRSET_YMAX, graph[i].max_y);
SetCtrlVal(pan_handle, GRSET_XDIV, graph[i].x_divisions);
SetCtrlVal(pan_handle, GRSET_YDIV, graph[i].y_divisions);
SetCtrlVal(pan_handle, GRSET_NUMPLOT, graph[i].total_plots);
    if (i<1)
        SetCtrlAttribute(pan_handle, GRSET_PREV, 15, 0);
    if (i<max_num-1)
        SetCtrlAttribute(pan_handle, GRSET_NEXT, 15, 1);
    break;
case GRSET_DONE:
    exit_flag = 1;
    break;
}
}
RemovePopup(0);
UnloadPanel(pan_handle);
return 0;
}

```

```

int GraphPlot(int graph_num, int plot_num, int total_points, float *x_array, float *y_array)
{
    if(graph_num >= system.total_graphs) {
        issueWarning("Attempt to plot on invalid graph.", 2.0);
        return 1;
    }
    if(plot_num >= graph[graph_num].total_plots) {
        issueWarning("Plot number exceeds limit for the graph.", 2.0);
        return 2;
    }
    strcpy(plot.name, graph[graph_num].name);
    strcpy(plot.x_title, graph[graph_num].x_title);
    strcpy(plot.y_title, graph[graph_num].y_title);
    plot.min_x = graph[graph_num].min_x;
    plot.max_x = graph[graph_num].max_x;
    plot.min_y = graph[graph_num].min_y;
    plot.max_y = graph[graph_num].max_y;
    plot.x_divisions = graph[graph_num].x_divisions;
    plot.y_divisions = graph[graph_num].y_divisions;
    plot.x_logscale = graph[graph_num].x_logscale;
    plot.y_logscale = graph[graph_num].y_logscale;
    plot.x_array[plot_num] = x_array;
    plot.y_array[plot_num] = y_array;
}

```

```

    plot.total_points[plot_num] = total_points;
    plot.total_plots = graph[graph_num].total_plots;

    return 0;
}

/*-----
| This function installs a graph window and displays a graph
| depending on the values of parameters in graph structure.
| It also provides the floating cursors and displays cursor
| co-ordinates.
|-----*/
int GraphDisplay(void)
{
    int graph_event = -1, exit_flag=0;
    int i, x_autorange, y_autorange;
    double cursor1_x, cursor1_y, cursor2_x, cursor2_y, dx, dy, slope;
    int color[6] = {0, 1, 4, 12, 14, 15};
    int symbol[6] = {0, 3, 1, 4, 6, 7};

    plot.graph_panel = LoadPanel("FLEXCAT.UIR", GRAPH);

    SetGraphAttribute(plot.graph_panel, GRAPH_GRAF, 0, plot.name);
    SetGraphAttribute(plot.graph_panel, GRAPH_GRAF, 10, plot.x_title);
    SetGraphAttribute(plot.graph_panel, GRAPH_GRAF, 18, plot.y_title);
    SetGraphAttribute(plot.graph_panel, GRAPH_GRAF, 13, plot.x_logscale);
    SetGraphAttribute(plot.graph_panel, GRAPH_GRAF, 21, plot.y_logscale);
    SetGraphAttribute(plot.graph_panel, GRAPH_GRAF, 15, plot.x_divisions);
    SetGraphAttribute(plot.graph_panel, GRAPH_GRAF, 23, plot.y_divisions);

    if(plot.min_x >= plot.max_x) /* implement auto ranging feature */
        x_autorange = 1;
    else
        x_autorange = 0;
    if(plot.min_y >= plot.max_y)
        y_autorange = 1;
    else
        y_autorange = 0;

    SetAxisRange(plot.graph_panel, GRAPH_GRAF, x_autorange, plot.min_x, plot.max_x, \
        y_autorange, plot.min_y, plot.max_y);

    for(i=0; i<plot.total_plots; i++) /* plot graphs */
        PlotXY(plot.graph_panel, GRAPH_GRAF, plot.x_array[i], plot.y_array[i], \
            plot.total_points[i], 3, 3, 1, symbol[i], 1, color[i]);

    InstallPopup(plot.graph_panel); /* display the panel */

    while(!exit_flag) {

```

```

GetGraphCursor(plot.graph_panel, GRAPH_GRAF, 1, &cursor1_x, &cursor1_y);
GetGraphCursor(plot.graph_panel, GRAPH_GRAF, 2, &cursor2_x, &cursor2_y);

SetCtrlVal(plot.graph_panel, GRAPH_CUR1X, cursor1_x);
SetCtrlVal(plot.graph_panel, GRAPH_CUR1Y, cursor1_y);
SetCtrlVal(plot.graph_panel, GRAPH_CUR2X, cursor2_x);
SetCtrlVal(plot.graph_panel, GRAPH_CUR2Y, cursor2_y);

dx = cursor2_x - cursor1_x;
dy = cursor2_y - cursor1_y;
if(dx != 0.0) {      /* check for divide by zero */
    slope = dy/dx;
    SetCtrlVal(plot.graph_panel, GRAPH_SLOP, slope);
}

GetPopupEvent(0, &graph_event);
if(graph_event == GRAPH_DONE)
    exit_flag = 1;
if(graph_event == GRAPH_PRINT)
    GraphPrint();
}

RemovePopup(0);
UnloadPanel(plot.graph_panel); /* unload graph panel */

return 0;
} /* function DisplayGraph ends here */

```

```

/*-----
| This function installs a graph window and displays a graph
| depending on the values of parameters in graph structure.
| It also provides the floating cursors and displays cursor
| co-ordinates.
|-----*/

```

```

int GraphPrint(void)
{
    int ctrl_event = -1, exit_flag=0, status = 0;
    int pan_handle, size_code = 1, device_num = 0;
    int orientation_code = 1, eject_code = 1;
    char file_name[8];
    double height = 10.0, width = 8.0;

    pan_handle = LoadPanel("FLEXCAT.UIR", GPRINT);
    InstallPopup(pan_handle);

    while(!exit_flag) {

```

```

    GetCtrlVal(pan_handle, GPRINT_NAME, file_name);

    GetPopupEvent(0,&ctrl_event);
    if(ctrl_event == GPRINT_DONE)
        exit_flag = 1;
}

/* ConfigurePrinter(file_name, orientation_code, width, height, eject_code); */

RemovePopup(0);
UnloadPanel(pan_handle);

status = OutputGraph(device_num, "graph", size_code, GRAPH, GRAPH_GRAF);

if(status == -25)
    IssueWarning("Hard copy output failed.", 3.0);

return 0;
}

/*-----
| This function installs a window and obtains data acquisition
| parameter from the user. It also creates a new file on the
| drive containing a header for the data acquisition purpose.
|-----*/
int DataAcqSetup(void)
{
    int ctrl_event = -1, exit_flag=0;
    int pan_handle, status, i;
    char temp[20], temp_filename[15];
    FILE *temp_file_handle;

    pan_handle = LoadPanel("FLEXCAT.UIR", DATACQ);
    InstallPopup(pan_handle);

    while(!exit_flag)
    {
        GetPopupEvent(0,&ctrl_event);
        switch (ctrl_event) {
            case DATACQ_NAME:
                GetCtrlVal(pan_handle, DATACQ_NAME, data_acq.file_name);
                strcpy(temp_filename, data_acq.file_name); /* attach .dat extension */
                strcat(temp_filename, ".dat");
                temp_file_handle = fopen(temp_filename, "r"); /* check if a file by that name already
exists */
                if(temp_file_handle != NULL) {
                    IssueWarning("File already exists. It will be overwritten !", 2.0);
                    fclose(temp_file_handle);

```

```

    }
    break;
    case DATACQ_INTERVAL:
        GetCtrlVal(pan_handle, DATACQ_INTERVAL, &data_acq.interval);
        if((data_acq.interval < 0.1) || (data_acq.interval > 10000.0)) {
            IssueWarning("Out of range value for time interval.", 2.0);
            data_acq.interval = 1.0;
        }
        SetCtrlVal(pan_handle, DATACQ_INTERVAL, data_acq.interval);
        break;
    case DATACQ_SENSNUM:
        GetCtrlVal(pan_handle, DATACQ_SENSNUM, data_acq.sensor_string);
        /* parse string into channel numbers */
        status = Parser(data_acq.sensor_string, &data_acq.total_sensors, data_acq.sensor_num);
        if(status == 1)
            IssueWarning("Invalid sensor number(s), try again.", 2.0);
        break;
    case DATACQ_MAXREAD:
        GetCtrlVal(pan_handle, DATACQ_MAXREAD, &data_acq.max_readings);
        if((data_acq.max_readings < 0) || (data_acq.max_readings > 200)) {
            IssueWarning("Out of range value for number of readings.", 2.0);
            data_acq.max_readings = 1;
        }
        SetCtrlVal(pan_handle, DATACQ_MAXREAD, data_acq.max_readings);
        break;
    case DATACQ_CAN:
        RemovePopup(0);
        UnloadPanel(pan_handle);
        return 1;
    case DATACQ_DONE:
        exit_flag = 1;
        break;
}
}

RemovePopup(0);
UnloadPanel(pan_handle);

/* open a new file */
data_acq.file_handle = fopen(temp_filename, "w+t");
if(data_acq.file_handle == NULL) {
    IssueWarning("Error creating data acquisition file.", 2.0);
    return 1;
}

InitializeTime(); /* initialize calander structure */

/*print main header */
fprintf(data_acq.file_handle, "%s\n", "SPC:");
fprintf(data_acq.file_handle, "\'%s\' %-2d %-4d ", data_acq.file_name, data_acq.total_sensors,
data_acq.max_readings);
fprintf(data_acq.file_handle, "%-8.3f %-4d \n", data_acq.interval, data_acq.readings_taken);

```

```

fprintf(data_acq.file_handle, "\n%s\n", calander.date_created);
fprintf(data_acq.file_handle, "\n%s\n", calander.time_created);
fprintf(data_acq.file_handle, "%s\n", "****");

/* print sub headers for each sensor */
fprintf(data_acq.file_handle, "%-15s ", "\nSensor num\n"); /* print first line */
for(i=0;i<data_acq.total_sensors-1;i++)
    fprintf(data_acq.file_handle, "%-15d ", data_acq.sensor_num[i]);
fprintf(data_acq.file_handle, "%-15d\n", data_acq.sensor_num[data_acq.total_sensors-1]);

fprintf(data_acq.file_handle, "%-15s ", "\nCalib Factor\n"); /* print second line */
for(i=0;i<data_acq.total_sensors-1;i++)
    fprintf(data_acq.file_handle, "%-+15.6f ", sensor[data_acq.sensor_num[i]].calib_factor);
fprintf(data_acq.file_handle, "%-+15.6f\n", sensor[data_acq.sensor_num[data_acq.total_sensors-1]].calib_factor);

fprintf(data_acq.file_handle, "%-15s ", "\nAbs Zero\n"); /* print third line */
for(i=0;i<data_acq.total_sensors-1;i++)
    fprintf(data_acq.file_handle, "%-+15.6f ", sensor[data_acq.sensor_num[i]].abs_zero);
fprintf(data_acq.file_handle, "%-+15.6f\n", sensor[data_acq.sensor_num[data_acq.total_sensors-1]].abs_zero);

fprintf(data_acq.file_handle, "%-15s ", "\nFloat Zero\n"); /* print fourth line */
for(i=0;i<data_acq.total_sensors-1;i++)
    fprintf(data_acq.file_handle, "%-+15.6f ", sensor[data_acq.sensor_num[i]].zero);
fprintf(data_acq.file_handle, "%-+15.6f\n", sensor[data_acq.sensor_num[data_acq.total_sensors-1]].zero);

fprintf(data_acq.file_handle, "%-15s ", "\nSeconds\n"); /* print fifth line */
for(i=0;i<data_acq.total_sensors-1;i++) {
    strcpy(temp, "");
    strcat(temp, sensor[data_acq.sensor_num[i]].unit);
    strcat(temp, "\n");
    fprintf(data_acq.ile_handle, "%-15s ", temp);
}
strcpy(temp, "");
strcat(temp, sensor[data_acq.sensor_num[data_acq.total_sensors-1]].unit);
strcat(temp, "\n");
fprintf(data_acq.file_handle, "%-15s\n", temp);

fprintf(data_acq.file_handle, "%-15s ", "\nClock\n"); /* print sixth line */
for(i=0;i<data_acq.total_sensors-1;i++) {
    strcpy(temp, "");
    strcat(temp, sensor[data_acq.sensor_num[i]].name);
    strcat(temp, "\n");
    fprintf(data_acq.file_handle, "%-15s ", temp);
}
strcpy(temp, "");
strcat(temp, sensor[data_acq.sensor_num[data_acq.total_sensors-1]].name);
strcat(temp, "\n");
fprintf(data_acq.file_handle, "%-15s\n", temp);

```

```

fclose(data_acq.file_handle);

/* reopen the file in update mode */
strcpy(temp, data_acq.file_name);      /* attach .dat extension */
strcat(temp, ".dat");
data_acq.file_handle = fopen(temp, "r+t");
if(data_acq.file_handle == NULL) {
    IssueWarning("Cannot reopen data acquisition file.", 2.0);
    return 1;
}

data_acq.readings_taken = 0; /* reset the reading count */
data_acq.setup_flag = 1;    /* set data acquisition setup flag */

SetMenuBarAttribute(MENU_DATA_STAR, 6, 1); /* enable Start menu item */
SetMenuBarAttribute(MENU_DATA_INT, 6, 1); /* enable Interval menu item */
SetMenuBarAttribute(MENU_DATA_CLO, 6, 1); /* enable Close menu item */
SetMenuBarAttribute(MENU_DATA_NEW, 6, 0); /* disable New menu item */
SetMenuBarAttribute(MENU_DATA_OPEN, 6, 0); /* disable Open menu item */

return 0;
} /* function DataAcqSetup ends here */

```

```

/*-----
| This function takes a string pointed by *string and parses it |
| into an integer array pointed by num_array and returns the   |
| number of members of this array as total_numbers. It        |
| recognizes ',' '-' and blank space as delimiters            |
|-----*/
int Parser(char *string, int *total_numbers, int *num_array)
{
    char *endptr, delim[30];
    int number[30];
    int max_num, max_delim;
    int indx = 0, i = 0, j = 0;

    endptr = string; /* set pointer to the beginning of the string */

    /* scan the string for numbers and delimiters and separate them
       in two separate arrays number and delim */
    while(endptr < string + strlen(string)) {
        number[i] = (int) strtol(endptr, &endptr, 10);
        delim[j] = *endptr;
        i++;
        j++;
        endptr++;
    }
}

```

```

max_num = i;    /* obtain the maximum index values for the arrays */
max_delim = j-1;

for(i=0; i<max_num-1; i++)    /* check for consistency in user supplied data */
    if(number[i] >= number[i+1])
        return 1;
for(j=0; j<max_delim; j++)
    if( (delim[j] == '-') && (delim[j+1] == '-') )
        return 1;

indx = 0;    /* generate number vector */
num_array[indx] = number[indx];

for(j=0; j<max_delim; j++) {
    if( (delim[j] == ',') || (delim[j] == ' ') ) {
        indx++;
        num_array[indx] = number[j+1];
    }

    if(delim[j] == '-') {
        for(i=1; i<=(number[j+1]-number[j]); i++) {
            indx ++;
            num_array[indx] = number[j]+i;
        }
    }
}

*total_numbers = indx+1;    /* compute the total number of members of num_array */

for(i=0; i<*total_numbers; i++) {    /* check for any invalid sensor number */
    if(num_array[i] >= system.total_sensors)
        return 1;
}
return 0;
} /* function Parser ends here */

```

```

int InitializeTime(void)
{
    struct tm *time_now;
    time_t seconds_now;

    tzset();
    time(&seconds_now);
    time_now = localtime(&seconds_now);

    strftime(calander.date_created, 17, "%x", time_now);
    strftime(calander.time_created, 10, "%X", time_now);

    calander.elapsed_time = 0.0;
}

```



```

return 0;
} /* function InitializeTime ends here */

```

```

/*-----
| This function uses a built in file selection function to allow
| user to select and open an existing data acquisition file and
| use it to acquire more data. It automatically recreates all
| past readings for the selected test using UserCompute function.
|-----*/

```

```

int DataAcqOpen(void)
{
    int status, i, j, sensor_num, dash_flag[25];
    char *title = "Select Data File";
    char file_name[40], test_string[5], temp[512];
    char num_string[5], sens_string[40];
    float involt;

    status = FileSelectPopup(" ", ".dat", title, 0, 1, 1, file_name);

    if (status == 0) /* if no file is selected then exit */
        return 1;
    if (status == -26) {
        IssueWarning("Invalid directory selection.", 2.0);
        return 1;
    }

    data_acq.file_handle = fopen(file_name, "r+t");
    if (data_acq.file_handle == NULL) {
        IssueWarning("Cannot open the data acquisition file.", 2.0);
        return 1;
    }
    /* see if the selection is a data acquisition file */
    fseek(data_acq.file_handle, 0L, SEEK_SET);
    fscanf(data_acq.file_handle, "%s\n", test_string);
    if (strcmp(test_string, "SPC:") != 0) {
        IssueWarning("Selection is not a data acquisition file.", 2.0);
        fclose(data_acq.file_handle);
        return 1;
    }
    /* read data from main header */
    fscanf(data_acq.file_handle, "%s %d %d ", temp, &data_acq.total_sensors,
    &data_acq.max_readings);
    fscanf(data_acq.file_handle, "%f %d\n", &data_acq.interval, &data_acq.readings_taken);

    for (i=0; i < strlen(temp); i++) { /* strip quotation marks from file name */
        if (temp[i + 1] == '"')
            break;
        data_acq.file_name[i] = temp[i + 1]; /* assign file name */
    }
}

```

```

}

fgets(temp, 512, data_acq.file_handle); /* reject date */
fgets(temp, 512, data_acq.file_handle); /* reject time */
fgets(temp, 512, data_acq.file_handle); /* reject "****" */

/* read data from sub headers */
fscanf(data_acq.file_handle, "%s %s", temp, temp); /* reject "Sensor num" */
for(i=0;i<data_acq.total_sensors;i++) {
    fscanf(data_acq.file_handle,"%d", &data_acq.sensor_num[i]);
    if(data_acq.sensor_num[i] >= system.total_sensors) {
        IssueWarning("A sensor number exceeds the allowable number.", 3.0);
        return 1;
    }
}

/* reconstruct sensor string */
dash_flag[0] = 1;
for(i = 1; i<data_acq.total_sensors; i++)
    if(data_acq.sensor_num[i] == (data_acq.sensor_num[i-1]+1))
        dash_flag[i] = 1;
    else
        dash_flag[i] = 0;

itoa(data_acq.sensor_num[0], num_string, 10);
strcpy(sens_string, num_string);
for(i=1;i<data_acq.total_sensors-1; i++) {
    if(dash_flag[i] == 1)
        if(dash_flag[i+1] == 0) {
            strcat(sens_string, "-");
            itoa(data_acq.sensor_num[i], num_string, 10);
            strcat(sens_string, num_string);
        }
    if(dash_flag[i] == 0) {
        strcat(sens_string, ",");
        itoa(data_acq.sensor_num[i], num_string, 10);
        strcat(sens_string, num_string);
    }
}

if(dash_flag[data_acq.total_sensors-1] == 0) /* take care of last sensor */
    strcat(sens_string, ",");
else if(dash_flag[data_acq.total_sensors-1] == 1)
    strcat(sens_string, "-");
itoa(data_acq.sensor_num[data_acq.total_sensors-1], num_string, 10);
strcat(sens_string, num_string);

strcpy(data_acq.sensor_string, sens_string);

fscanf(data_acq.file_handle, "%s %s", temp, temp); /* reject "Calib Factor" */
for(i=0;i<data_acq.total_sensors;i++)

```

```

    fscanf(data_acq.file_handle, "%f", &sensor[data_acq.sensor_num[i]].calib_factor);

    fscanf(data_acq.file_handle, "%s %s", temp, temp);    /* reject "Abs Zero" */
    for(i=0;i<data_acq.total_sensors;i++)
        fscanf(data_acq.file_handle, "%f", &sensor[data_acq.sensor_num[i]].abs_zero);

    fscanf(data_acq.file_handle, "%s %s", temp, temp);    /* reject "Float Zero" */
    for(i=0;i<data_acq.total_sensors;i++)
        fscanf(data_acq.file_handle, "%f", &sensor[data_acq.sensor_num[i]].zero);

    fgets(temp, 512, data_acq.file_handle);    /* reject carriage return */
    fgets(temp, 512, data_acq.file_handle);    /* reject sensor units */
    fgets(temp, 512, data_acq.file_handle);    /* reject sensor names */

    /* finally we are ready for computing data points */
    for(j=0;j<data_acq.readings_taken;j++) {
        fscanf(data_acq.file_handle, "%f", &calander.elapsed_time);
        for(i=0;i<data_acq.total_sensors;i++) {
            fscanf(data_acq.file_handle, "%f", &involt);
            sensor_num = data_acq.sensor_num[i];
            sensor[sensor_num].abs_reading = sensor[sensor_num].calib_factor*(involt -
            sensor[sensor_num].abs_zero);
            sensor[sensor_num].reading = sensor[sensor_num].calib_factor*(involt -
            sensor[sensor_num].zero);
        }
        UserCompute();
    }

    data_acq.setup_flag = 1;    /* set data acquisition setup flag */

    SetMenuBarAttribute(MENU_DATA_STAR, 6, 1);    /* enable Start menu item */
    SetMenuBarAttribute(MENU_DATA_CLO, 6, 1);    /* enable Close menu item */
    SetMenuBarAttribute(MENU_DATA_INT, 6, 1);    /* enable Interval menu item */
    SetMenuBarAttribute(MENU_DATA_NEW, 6, 0);    /* disable New menu item */
    SetMenuBarAttribute(MENU_DATA_OPEN, 6, 0);    /* disable Open menu item */

    return 0;
} /* function DataAcqOpen ends here */

```

```

/*-----
| This function activates an already set-up or suspended          |
| data acquisition task. The task can be suspended again         |
| using DataAcqStop function.                                    |
|-----*/
int DataAcqRun(void)
{
    float interval;

```

```

if(control.system_run_flag == 0) {
    IssueWarning("System is not running.", 2.0);
    return 1;
}

AcquireData(); /* take first set of readings */

interval = data_acq.interval*1000.0; /* start the data acq thread */
control.thread_num[4] = t_sched_set(AcquireData, (unsigned long) interval, T_RESTART);
if(control.thread_num[4] == T_FAIL) /* check if allocation failed for the thread */
    IssueWarning("Unable to schedule a thread", -1.0);

data_acq.run_flag = 1; /* set data acquisition run flag */

SetMenuBarAttribute(MENU_DATA_STAR, 6, 0); /* disable Start menu item */
SetMenuBarAttribute(MENU_DATA_CLO, 6, 0); /* disable Close menu item */
SetMenuBarAttribute(MENU_DATA_STOP, 6, 1); /* enable Stop menu item */

return 0;
} /* function DataAcqRun ends here */

```

```

/*-----
| This function temporarily suspends a previously running
| data acquisition task. The task can be started again
| using DataAcqRun function.
|-----*/

```

```

int DataAcqStop(void)
{

    t_sched_delete(control.thread_num[4]); /* delete data acquisition thread */

    data_acq.run_flag = 0; /* clear data acquisition run flag */

    SetMenuBarAttribute(MENU_DATA_STAR, 6, 1); /* enable Start menu item */
    SetMenuBarAttribute(MENU_DATA_CLO, 6, 1); /* enable Close menu item */
    SetMenuBarAttribute(MENU_DATA_STOP, 6, 0); /* disable Stop menu item */

    return 0;
} /* function DataAcqStop ends here */

```

```

/*-----
| This function installs a window and allows the user to
| change time interval and maximum number of readings for
| currently running data acquisition task.
|-----*/

```

```

int DataAcqInterval(void)
{
    int ctrl_event = -1, exit_flag=0;
    int pan_handle, status;

    pan_handle = LoadPanel("FLEXCAT.UIR", DATACQ);
    SetCtrlVal(pan_handle, DATACQ_NAME, data_acq.file_name);
    SetCtrlVal(pan_handle, DATACQ_INTERVAL, data_acq.interval);
    SetCtrlVal(pan_handle, DATACQ_SENSNUM, data_acq.sensor_string);
    SetCtrlVal(pan_handle, DATACQ_MAXREAD, data_acq.max_readings);
    SetCtrlVal(pan_handle, DATACQ_READTAKE, data_acq.readings_taken);
    SetCtrlAttribute(pan_handle, DATACQ_NAME, 15, 0);
    SetCtrlAttribute(pan_handle, DATACQ_SENSNUM, 15, 0);
    InstallPopup(pan_handle);

    while(!exit_flag)
    {
        GetPopupEvent(0,&ctrl_event);
        switch (ctrl_event) {
            case DATACQ_CAN:
                exit_flag = 1;
                break;
            case DATACQ_INTERVAL:
                GetCtrlVal(pan_handle, DATACQ_INTERVAL, &data_acq.interval);
                break;
            case DATACQ_MAXREAD:
                GetCtrlVal(pan_handle, DATACQ_MAXREAD, &data_acq.max_readings);
                break;
            case DATACQ_DONE:
                exit_flag = 1;
                break;
        }
    }
    RemovePopup(0);
    UnloadPanel(pan_handle);

    return 0;
} /* function DataAcqInterval ends here */

```

```

/*-----
| This function performs house cleaning operations associated |
| with closing a data acquisitions task. It is essential to  |
| stop and close a task before quitting the application.    |
|-----*/

```

```

int DataAcqClose(void)
{

    /* update the main header */

```

```

fseek(data_acq.file_handle, 0L, SEEK_SET);
fprintf(data_acq.file_handle, "%s\n", "SPC:");
fprintf(data_acq.file_handle, "\n%s\ " %-2d %-4d ", data_acq.file_name, data_acq.total_sensors,
data_acq.max_readings);
fprintf(data_acq.file_handle, "%-8.3f %-4d \n", data_acq.interval, data_acq.readings_taken);

fclose(data_acq.file_handle); /* close the data acq file */
data_acq.setup_flag = 0; /* clear data acquisition setup flag */

SetMenuBarAttribute(MENU_DATA_NEW, 6, 1); /* enable New menu item */
SetMenuBarAttribute(MENU_DATA_OPEN, 6, 1); /* enable Open menu item */
SetMenuBarAttribute(MENU_DATA_STAR, 6, 0); /* disable Start menu item */
SetMenuBarAttribute(MENU_DATA_INT, 6, 0); /* disable interval menu item */
SetMenuBarAttribute(MENU_DATA_STOP, 6, 0); /* disable Stop menu item */
SetMenuBarAttribute(MENU_DATA_CLO, 6, 0); /* disable Close menu item */

return 0;
} /* function DataAcqClose ends here */

```

```

/*-----
| This function does the actual data acquisition. It runs in      |
| the background a thread #5 of the scheduler. Function          |
| UserCompute is also run in through this task.                  |
|-----*/

```

```

void AcquireData(void)
{
    int i, dup_handle;
    int card_num, channel_num;
    float involt;

    /* check if maximum limit is exceeded */
    if(data_acq.readings_taken >= data_acq.max_readings) {
        error.error_flag = 5;
        return;
    }

    UserCompute(); /* perform user computations */

    fseek(data_acq.file_handle, 0L, SEEK_END);
    fprintf(data_acq.file_handle, "%-15.3f ", calander.elapsed_time); /* print time */

    for(i=0; i<data_acq.tctal_sensors-1; i++) {
        card_num = sensor[data_acq.sensor_num[i]].card_num;
        channel_num = sensor[data_acq.sensor_num[i]].channel_num;
        involt = adcard[card_num].involt[channel_num]; /* obtain latest volt reading */
        fprintf(data_acq.file_handle, "%-+15.6f ", involt); /* write the volt reading to file */
    }
    i = data_acq.total_sensors-1; /* do same for the last sensor */

```

```

card_num = sensor[data_acq.sensor_num[i]].card_num;
channel_num = sensor[data_acq.sensor_num[i]].channel_num;
involt = adcard[card_num].involt[channel_num];
fprintf(data_acq.file_handle, "%-+15.6f\n", involt);

calander.elapsed_time = calander.elapsed_time + data_acq.interval; /* update current time */

data_acq.readings_taken ++; /* increment the reading counter */

fflush(data_acq.file_handle); /* flush the file buffer */
dup_handle = dup(fileno(data_acq.file_handle)); /* make a duplicate file handle */
close(dup_handle); /* close the duplicate handle to flush DOS buffer */

return;
} /* function AcquireData ends here */

```

```

/*-----
| This function displays a window and obtains from user the |
| A/D card parameters stored in structure adcard.           |
|-----*/

```

```

int AdcardSetup(void)
{
    int ctrl_event = -1, exit_flag = 0, i = 0;
    int pan_handle, max_num, status;

    max_num = system.total_adcards;
    pan_handle = LoadPanel("FLEXCAT.UIR", ADSET);
    SetCtrlVal(pan_handle, ADSET_SERNUM, i);
    SetCtrlVal(pan_handle, ADSET_BASADR, adcard[i].base_adress);
    SetCtrlVal(pan_handle, ADSET_RESL, adcard[i].resolution);
    SetCtrlVal(pan_handle, ADSET_INTIME, adcard[i].integration_time);
    SetCtrlVal(pan_handle, ADSET_CALTIME, adcard[i].calibration_time);
    SetCtrlVal(pan_handle, ADSET_TOTALCH, adcard[i].total_channel);
    SetCtrlVal(pan_handle, ADSET_BCAL, adcard[i].background_cal_flag);
    SetCtrlVal(pan_handle, ADSET_ENULL, adcard[i].electronic_null_flag);
    InstallPopup(pan_handle);
    if(max_num == 1)
        SetCtrlAttribute(pan_handle, ADSET_NEXT, 15, 0);

    while(!exit_flag)
    {
        GetCtrlVal(pan_handle, ADSET_BASADR, &adcard[i].base_adress);
        GetCtrlVal(pan_handle, ADSET_RESL, &adcard[i].resolution);
        GetCtrlVal(pan_handle, ADSET_INTIME, &adcard[i].integration_time);
        GetCtrlVal(pan_handle, ADSET_CALTIME, &adcard[i].calibration_time);
        GetCtrlVal(pan_handle, ADSET_TOTALCH, &adcard[i].total_channel);
        GetCtrlVal(pan_handle, ADSET_BCAL, &adcard[i].background_cal_flag);
        GetCtrlVal(pan_handle, ADSET_ENULL, &adcard[i].electronic_null_flag);
    }
}

```

```

GetPopupEvent(0,&ctrl_event);
switch (ctrl_event) {
    case ADSET_NEXT:
        i = i+1;
        SetCtrlVal(pan_handle, ADSET_SERNUM, i);
        SetCtrlVal(pan_handle, ADSET_BASADR, adcard[i].base_adress);
        SetCtrlVal(pan_handle, ADSET_RESL, adcard[i].resolution);
        SetCtrlVal(pan_handle, ADSET_INTIME, adcard[i].integration_time);
        SetCtrlVal(pan_handle, ADSET_CALTIME, adcard[i].calibration_time);
        SetCtrlVal(pan_handle, ADSET_TOTALCH, adcard[i].total_channel);
        SetCtrlVal(pan_handle, ADSET_BCAL, adcard[i].background_cal_flag);
        SetCtrlVal(pan_handle, ADSET_ENULL, adcard[i].electronic_null_flag);
        if (i>0)
            SetCtrlAttribute(pan_handle, ADSET_PREV, 15, 1);
        if (i >= max_num - 1)
            SetCtrlAttribute(pan_handle, ADSET_NEXT, 15, 0);
        break;
    case ADSET_PREV:
        i = i-1;
        SetCtrlVal(pan_handle, ADSET_SERNUM, i);
        SetCtrlVal(pan_handle, ADSET_BASADR, adcard[i].base_adress);
        SetCtrlVal(pan_handle, ADSET_RESL, adcard[i].resolution);
        SetCtrlVal(pan_handle, ADSET_INTIME, adcard[i].integration_time);
        SetCtrlVal(pan_handle, ADSET_CALTIME, adcard[i].calibration_time);
        SetCtrlVal(pan_handle, ADSET_TOTALCH, adcard[i].total_channel);
        SetCtrlVal(pan_handle, ADSET_BCAL, adcard[i].background_cal_flag);
        SetCtrlVal(pan_handle, ADSET_ENULL, adcard[i].electronic_null_flag);
        if (i<1)
            SetCtrlAttribute(pan_handle, ADSET_PREV, 15, 0);
        if (i<max_num-1)
            SetCtrlAttribute(pan_handle, ADSET_NEXT, 15, 1);
        break;
    case ADSET_DEF:
        adcard[i].resolution = 12;
        adcard[i].integration_time = 4;
        adcard[i].calibration_time = 5;
        adcard[i].background_cal_flag = 1;
        adcard[i].electronic_null_flag = 1;

        SetCtrlVal(pan_handle, ADSET_RESL, adcard[i].resolution);
        SetCtrlVal(pan_handle, ADSET_INTIME, adcard[i].integration_time);
        SetCtrlVal(pan_handle, ADSET_CALTIME, adcard[i].calibration_time);
        SetCtrlVal(pan_handle, ADSET_BCAL, adcard[i].background_cal_flag);
        SetCtrlVal(pan_handle, ADSET_ENULL, adcard[i].electronic_null_flag);
        break;
    case ADSET_DONE:
        exit_flag = 1;
        break;
}
}
RemovePopup(0);
UnloadPanel(pan_handle);

```



```

for(i=0; i<system.total_adcards; i++) /* initialize all cards */
AdcardInitialize(i);

return 0;
} /* Function AdcardSetup ends here */

/*-----
| This function checks if selected A/D card is ready to respond. |
| It returns 1 if the card is not ready and otherwise computes |
| internal parameters and programs the card for selected |
| values of integration, calibration times, data format, |
| background calibration and electronic nulling. |
|-----*/
int AdcardInitialize(int card_num)
{
    unsigned long interval;
    int adress, status;
    char *message;
    char *number;

    adcard[card_num].max_count = pow(2, (double) (adcard[card_num].resolution+8));

    /* compute reading frequency from integration time */
    switch (adcard[card_num].integration_time) {
        case 0:
            interval = 10;
            break;
        case 1:
            interval = 16.7;
            break;
        case 2:
            interval = 20;
            break;
        case 3:
            interval = 100;
            break;
        case 4:
            interval = 166.7;
            break;
        case 5:
            interval = 300;
            break;
    }
    /* this extra 4 ms. is time delay required for task switch */
    adcard[card_num].time_count = interval + 4L;

    adress = adcard[card_num].base_adress;

```

```

    outportb(address, CNV);    /* check if A/D convertor is ready or not */
    status = Wait(address, 1, 1);
    if(status == 1) {
        strcpy(message, "A/D convertor not ready/present at address ");
        itoa(address, number, 10);
        strcat(message, number);
        IssueWarning(message, 2.0);
        return 1;
    }

    outportb(address + 1, adcard[card_num].resolution+1); /* load resolution format */
    Wait(address, 1, 1);

    outportb(address, SDF); /* set resolution format */
    Wait(address, 1, 1);

    /* set background calibration capability */
    if (adcard[card_num].background_cal_flag == 1) {
        outportb(address, SDC+adcard[card_num].calibration_time+1);
        Wait(address, 1, 1);
        outportb(address, CALEN);
    }
    else
        outportb(address, CALDI);
    Wait(address, 1, 1);

    outportb(address, SDI+adcard[card_num].integration_time+1); /* set integration time */
    Wait(address, 1, 1);

    if (adcard[card_num].electronic_null_flag == 1) /* set electronic null capability */
        outportb(address, NULEN);
    else
        outportb(address, NULDI);
    Wait(address, 1, 1);

    return 0;
} /* function AdcardInitialize ends here */

```

```

/*-----
| This function displays a window and presents a menu to      |
| provide access to various A/D card calibration functions.   |
|-----*/

```

```

int AdcardCalibrate(void)
{
    int ctrl_event = -1, exit_flag = 0;
    int pan_handle, card_num, max_value, status;
    char option[80];

    max_value = system.total_adcards - 1;

```

```

pan_handle = LoadPanel("FLEXCAT.UIR", ADCAL);
InstallPopup(pan_handle);
SetCtrlAttribute(pan_handle, ADCAL_CARDNUM, 13, max_value);
while (!exit_flag) {
    status = GetCtrlVal(pan_handle, ADCAL_CARDNUM, &card_num);
    if (status == -14) {
        IssueWarning("Invalid A/D card number.", 2.0);
        SetCtrlVal(pan_handle, ADCAL_CARDNUM, 0);
    }

    GetPopupEvent(0,&ctrl_event);

    if(ctrl_event == ADCAL_OPTION) { /* process the event */
        GetCtrlVal(pan_handle, ADCAL_OPTION, option);
        switch (option[2]) {
            case 'n':
                SingleCalibration(card_num);
                break;
            case 'f':
                OffsetCorrection(card_num);
                break;
            case 'i':
                GainCorrection(card_num);
                break;
            case 'v':
                SaveAll(card_num);
                break;
        }
    }
    if (ctrl_event == ADCAL_DONE)
        exit_flag = 1;
}
RemovePopup(0);
UnloadPanel(pan_handle);
return 0;
} /* Function AdcardCalibrate ends here */

```

```

/*-----
| This function performs a single calibration cycle using
| integration time of 300 ms. and then restores the intergation
| time to previous value.
|-----*/

```

```

int SingleCalibration(card_num)
{
    int ctrl_event = -1, exit_flag = 0;
    int pan_handle, adress;
    char *message = "\n This function will perform a single \n"
        " cycle of calibration using 300 ms. \n"
        " calibration time and then restore \n"

```

```

        " the original calibration time. ";
address = adcard[card_num].base_address;
pan_handle = LoadPanel("FLEXCAT.UIR", OKCAN);
InstallPopup(pan_handle);
SetPanelAttribute(pan_handle, 0, "Single Calibration");
SetCtrlVal(pan_handle, OKCAN_TEXT, message);
while(!exit_flag) {
    GetPopupEvent(0,&ctrl_event);
    switch(ctrl_event) {
        case OKCAN_OK:

            outportb(address, SDC+6); /* set calibration time to 300 ms. */
            Wait(address, 1, 1);

            outportb(address, SCAL); /* perform single calibration cycle */
            Wait(address, 1, 1);

            /* restore original calibration time */
            outportb(address, SDC+adcard[card_num].calibration_time+1);
            exit_flag = 1;
            break;
        case OKCAN_CANC:
            exit_flag = 1;
            break;
    }
}
RemovePopup(0);
UnloadPanel(pan_handle);
return 0;
} /* function SingleCalibration ends here */

```

```

/*-----
| This function nullifies any offset voltage by performing an
| electronic offset correction operation. Channel 0 of the
| selected card must be shorted before the operation is
| performed. A separate Save operation is required to save the
| correction value permanently.
|-----*/

```

```

int OffsetCorrection(card_num)
{
    int ctrl_event = -1, exit_flag = 0;
    int pan_handle, address;
    float involt;

    char *message = " This function performs a digital null \n"
                   " operation and stores the offset value \n"
                   " in volatile memory. It is essential to \n"
                   " short channel 0 of selected card before \n"
                   " executing this function. Execute \n"
                   " \"Save Parameters\" command to store \n"

```

```

        " offset value in non-volatile memory. ";

address = adcard[card_num].base_address;

outportb(address, SDI+6); /* set integration time of 300 ms. */
Wait(address, 1, 1);

outportb(address, NULDI); /* disable electronic null capability */
Wait(address, 1, 1);

pan_handle = LoadPanel("FLEXCAT.UIR", OFFGAIN);
InstallPopup(pan_handle);
SetPanelAttribute(pan_handle, 0, "Offset Correction");
SetCtrlVal(pan_handle, OFFGAIN_TEXT, message);
while(!exit_flag) {
    involt = AdcardRead(card_num, 0);
    SetCtrlVal(pan_handle, OFFGAIN_CH0, involt);

    GetPopupEvent(0,&ctrl_event);
    switch(ctrl_event) {
        case OFFGAIN_OK:
            outportb(address, DONULL);
            Wait(address, 1, 1);
            outportb(address, NULEN);
            Wait(address, 1, 1);
            SetCtrlAttribute(pan_handle, OFFGAIN_OK, 15, 0);
            break;
        case OFFGAIN_CANC:
            exit_flag = 1;
            break;
    }
}

/* restore old integration time */
outportb(address, SDI+adcard[card_num].integration_time+1);
Wait(address, 1, 1);
/* restore old electronic null state */
if (adcard[card_num].electronic_null_flag == 1)
    outportb(address, NULEN);
else
    outportb(address, NULDI);
Wait(address, 1, 1);

RemovePopup(0);
UnloadPanel(pan_handle);
return 0;
} /* function OffsetCorrection ends here */

```

```

/*-----
| This function adjusts the full scale range by performing an |

```

```

| electronic gain correction operation. +5 V must be present on
| channel 0 of the selected card before the operation is
| performed. A separate Save operation is required to save the
| correction value permanently.
|-----*/
int GainCorrection(card_num)
{
    int ctrl_event = -1, exit_flag = 0;
    int pan_handle, address;
    float involt;
    char *message = " Perform offset correction before any \n"
                   " gain correction. Present +5 V source \n"
                   " to channel 0 of the selected card \n"
                   " before executing this function. \n"
                   " Execute \"Save Parameters\" command \n"
                   " to store the value of gain \n"
                   " correction in non-volatile memory. ";

    address = adcard[card_num].base_address;

    outportb(address, SDI+6); /* set integration time of 300 ms. */
    Wait(address, 1, 1);

    outportb(address, NULEN); /* enable electronic null capability */
    Wait(address, 1, 1);

    pan_handle = LoadPanel("FLEXCAT.UIR", OFFGAIN);
    InstallPopup(pan_handle);
    SetPanelAttribute(pan_handle, 0, "Gain Correction");
    SetCtrlVal(pan_handle, OFFGAIN_TEXT, message);
    while(!exit_flag) {
        involt = AdcardRead(card_num, 0);
        SetCtrlVal(pan_handle, OFFGAIN_CH0, involt);

        GetPopupEvent(0,&ctrl_event);
        switch(ctrl_event) {
            case OFFGAIN_OK:
                outportb(address, ECAL);
                Wait(address, 1, 1);
                SetCtrlAttribute(pan_handle, OFFGAIN_OK, 15, 0);
                break;
            case OFFGAIN_CANC:
                exit_flag = 1;
                break;
        }
    }
    RemovePopup(0);
    UnloadPanel(pan_handle);
    /* restore old integration time */
    outportb(address, SDI+adcard[card_num].integration_time+1);
    Wait(address, 1, 1);
    /* restore old electronic null state */
}

```

```

if (adcard[card_num].electronic_null_flag == 1)
    outportb(address, NULEN);
else
    outportb(address, NULDI);
Wait(address, 1, 1);

return 0;
} /* function GainCorrection ends here */

/*-----
| This function saves the values of integration/calibration time
| data format, and offset gain correction into N/V RAM on board
| AD1170. Maximum 1000 such operations are allowed.
|-----*/

int SaveAll(card_num)
{
    int ctrl_event = -1, exit_flag = 0;
    int pan_handle, address;
    char *message = "\n This function will save current values \n"
        " of integration/calibration times, \n"
        " data format, offset/gain correction \n"
        " values in on-chip non-volatile memory. \n"
        " Maximum 1000 such write \n"
        " operations are allowed. ";

    address = adcard[card_num].base_address;
    pan_handle = LoadPanel("FLEXCAT.UIR", OKCAN);
    InstallPopup(pan_handle);
    SetPanelAttribute(pan_handle, 0, "Save Parameters");
    SetCtrlVal(pan_handle, OKCAN_TEXT, message);
    while(!exit_flag) {
        GetPopupEvent(0, &ctrl_event);
        switch(ctrl_event) {
            case OKCAN_OK:
                outportb(address, SAVA);
                Wait(address, 1, 1);
                exit_flag = 1;
                break;
            case OKCAN_CANC:
                exit_flag = 1;
                break;
        }
    }
    RemovePopup(0);
    UnloadPanel(pan_handle);
    return 0;
} /* function SaveAll ends here */

```

```

/*-----
| This function displays a pop up window and continuously
| shows all the channel readings of selected A/D card.
|-----*/
int AdcardTest(void)
{
    int ctrl_event = -1, exit_flag = 0;
    int pan_handle, card_num = 0;
    int i;

    /* load voltmeter panel and display results */
    pan_handle = LoadPanel("FLEXCAT.UIR", ADTEST);
    InstallPopup(pan_handle);
    while (!exit_flag) {
        GetPopupEvent(0,&ctrl_event);

        for (i=0; i<adcard[card_num].total_channel; i++)
            SetCtrlVal(pan_handle, ADTEST_CH0+i, adcard[card_num].involt[i]);

        if (ctrl_event == ADTEST_CARDNUM) {
            GetCtrlVal(pan_handle, ADTEST_CARDNUM, &card_num);
            if (card_num > system.total_adcards - 1) {
                IssueWarning("Invalid A/D card number.", 2.0);
                card_num = 0;
                SetCtrlVal(pan_handle, ADTEST_CARDNUM, card_num);
            }
        }
        if (ctrl_event == ADTEST_DONE)
            exit_flag = 1;
    }
    RemovePopup(0);
    UnloadPanel(pan_handle);
    return 0;
} /* Function AdcardTest ends here */

```

```

/*-----
| This function displays a pop up window and obtains from user
| values of parameters contained in structure dacard.
|-----*/
int DacardSetup(void)
{
    int ctrl_event = -1, exit_flag = 0, i = 0;
    int pan_handle, max_num;
    int number, ch_number = 0;

```



```

max_num = system.total_dacards;
pan_handle = LoadPanel("FLEXCAT.UIR", DASET);
InstallPopup(pan_handle);
SetCtrlVal(pan_handle, DASET_SERNUM, i);
SetCtrlVal(pan_handle, DASET_CHNUM, ch_number);
SetCtrlVal(pan_handle, DASET_BASADR, dacard[i].base_adress);
SetCtrlVal(pan_handle, DASET_RES, dacard[i].resolution);
SetCtrlVal(pan_handle, DASET_RAN, dacard[i].range[ch_number]);
SetCtrlVal(pan_handle, DASET_UNI, dacard[i].polarity_code[ch_number]);
if(max_num == 1)
    SetCtrlAttribute(pan_handle, DASET_NEXT, 15, 0);

while(!exit_flag)
{
    GetCtrlVal(pan_handle, DASET_BASADR, &dacard[i].base_adress);
    GetCtrlVal(pan_handle, DASET_RES, &dacard[i].resolution);
    GetCtrlVal(pan_handle, DASET_CHNUM, &number);
    if(number != ch_number) {
        SetCtrlVal(pan_handle, DASET_RAN, dacard[i].range[number]);
        SetCtrlVal(pan_handle, DASET_UNI, dacard[i].polarity_code[number]);
        ch_number = number;
    }
    GetCtrlVal(pan_handle, DASET_RAN, &dacard[i].range[ch_number]);
    GetCtrlVal(pan_handle, DASET_UNI, &dacard[i].polarity_code[ch_number]);

    GetPopupEvent(0,&ctrl_event);
    switch (ctrl_event) {
        case DASET_NEXT:
            i = i+1;
            ch_number = 0;
            SetCtrlVal(pan_handle, DASET_SERNUM, i);
            SetCtrlVal(pan_handle, DASET_BASADR, dacard[i].base_adress);
            SetCtrlVal(pan_handle, DASET_RES, dacard[i].resolution);
            SetCtrlVal(pan_handle, DASET_CHNUM, ch_number);
            SetCtrlVal(pan_handle, DASET_RAN, dacard[i].range[ch_number]);
            SetCtrlVal(pan_handle, DASET_UNI, dacard[i].polarity_code[ch_number]);
            if (i>0)
                SetCtrlAttribute(pan_handle, DASET_PREV, 15, 1);
            if (i>=max_num-1)
                SetCtrlAttribute(pan_handle, DASET_NEXT, 15, 0);
            break;
        case DASET_PPREV:
            i = i-1;
            ch_number = 0;
            SetCtrlVal(pan_handle, DASET_SERNUM, i);
            SetCtrlVal(pan_handle, DASET_BASADR, dacard[i].base_adress);
            SetCtrlVal(pan_handle, DASET_RES, dacard[i].resolution);
            SetCtrlVal(pan_handle, DASET_CHNUM, ch_number);
            SetCtrlVal(pan_handle, DASET_RAN, dacard[i].range[ch_number]);
            SetCtrlVal(pan_handle, DASET_UNI, dacard[i].polarity_code[ch_number]);
            if (i<1)
                SetCtrlAttribute(pan_handle, DASET_PREV, 15, 0);
    }
}

```

```

        if (i<max_num-1)
            SetCtrlAttribute(pan_handle, DASET_NEXT, 15, 1);
        break;
    case DASET_DONE:
        exit_flag = 1;
        break;
    }
}
for(i=0; i<max_num; i++)
    dacard[i].max_count = pow(2, (double) dacard[i].resolution) - 1;
RemovePopup(0);
UnloadPanel(pan_handle);

return 0;
} /* Function DacardSetup ends here */

/*-----
| This function displays a pop up window and allows the user to |
| send voltage to specific channel(s) on selected D/A card. |
|-----*/
int DacardTest(void)
{
    int ctrl_event = -1, exit_flag = 0;
    int pan_handle, status;
    int card_num = 0, channel_num = 0, max_card_num;
    float volt, min_value, max_value;

    max_card_num = system.total_dacards - 1;
    pan_handle = LoadPanel("FLEXCAT.UIR", DATEST);
    SetCtrlVal(pan_handle, DATEST_CNUM, card_num);
    SetCtrlVal(pan_handle, DATEST_CHNUM, channel_num);
    InstallPopup(pan_handle);

    while (!exit_flag) {
        GetCtrlVal(pan_handle, DATEST_CNUM, &card_num);

        if (card_num > max_card_num) {
            IssueWarning("Invalid D/A card number.", 2.0);
            card_num = 0;
            SetCtrlVal(pan_handle, DATEST_CNUM, card_num);
        }

        GetPopupEvent(0,&ctrl_event);

        switch(ctrl_event) {
            case DATEST_SEND:
                GetCtrlVal(pan_handle, DATEST_CHNUM, &channel_num);
                /* compute the values of minimum and maximum output voltage for the selected channel */
                if (dacard[card_num].polarity_code[channel_num] == 1) {

```

```

        min_value = 0;
        max_value = dacard[card_num].range[channel_num];
    }
    else {
        max_value = dacard[card_num].range[channel_num]/2.0;
        min_value = -(max_value);
    }

    GetCtrlVal(pan_handle, DATEST_VOUT, &volt); /* get and echo the output volt value */
    SetCtrlVal(pan_handle, DATEST_VOUT, volt);

    if ((volt>max_value)||(volt<min_value)) { /* perform limit check on the value of volt */
        IssueWarning("Voltage value outside the range.", 2.0);
        break;
    }
    DacardWrite(card_num, channel_num, volt); /* send voltage to D/A card */
    break;
    case DATEST_DONE:
        exit_flag = 1;
        break;
    }
}

RemovePopup(0);
UnloadPanel(pan_handle);

return 0;
} /* function DacardTest ends here */

```

```

/*-----
| This function displays a pop up window and obtains from user |
| values of parameters contained in structure sensor.          |
|-----*/

```

```

int SensorSetup(void)
{
    int ctrl_event = -1, exit_flag = 0, i = 0;
    int pan_handle, max_num;

    max_num = system.total_sensors;

    pan_handle = LoadPanel("FLEXCAT.UIR", SENSET);
    InstallPopup(pan_handle);
    SetCtrlVal(pan_handle, SENSET_SERNUM, i);
    SetCtrlVal(pan_handle, SENSET_NAME, sensor[i].name);
    SetCtrlVal(pan_handle, SENSET_UNIT, sensor[i].unit);
    SetCtrlVal(pan_handle, SENSET_CARNUM, sensor[i].card_num);
    SetCtrlVal(pan_handle, SENSET_CHANUM, sensor[i].channel_num);
    SetCtrlVal(pan_handle, SENSET_CALFAC, sensor[i].calib_factor);
}

```

```

SetCtrlVal(pan_handle, SENSET_MAXRED, sensor[i].max_reading);
if(max_num == 1)
    SetCtrlAttribute(pan_handle, SENSET_NEXT, 15, 0);

while(!exit_flag)
{
    GetCtrlVal(pan_handle, SENSET_NAME, sensor[i].name);
    GetCtrlVal(pan_handle, SENSET_UNIT, sensor[i].unit);
    GetCtrlVal(pan_handle, SENSET_CARNUM, &sensor[i].card_num);
    GetCtrlVal(pan_handle, SENSET_CHANUM, &sensor[i].channel_num);
    GetCtrlVal(pan_handle, SENSET_CALFAC, &sensor[i].calib_factor);
    GetCtrlVal(pan_handle, SENSET_MAXRED, &sensor[i].max_reading);

    GetPopupEvent(0,&ctrl_event);
    switch (ctrl_event) {
        case SENSET_NEXT:
            i = i+1;
            SetCtrlVal(pan_handle, SENSET_SERNUM, i);
            SetCtrlVal(pan_handle, SENSET_NAME, sensor[i].name);
            SetCtrlVal(pan_handle, SENSET_UNIT, sensor[i].unit);
            SetCtrlVal(pan_handle, SENSET_CARNUM, sensor[i].card_num);
            SetCtrlVal(pan_handle, SENSET_CHANUM, sensor[i].channel_num);
            SetCtrlVal(pan_handle, SENSET_CALFAC, sensor[i].calib_factor);
            SetCtrlVal(pan_handle, SENSET_MAXRED, sensor[i].max_reading);
            if (i>0)
                SetCtrlAttribute(pan_handle, SENSET_PREV, 15, 1);
            if (i>= max_num-1)
                SetCtrlAttribute(pan_handle, SENSET_NEXT, 15, 0);
            break;
        case SENSET_PREV:
            i = i-1;
            SetCtrlVal(pan_handle, SENSET_SERNUM, i);
            SetCtrlVal(pan_handle, SENSET_NAME, sensor[i].name);
            SetCtrlVal(pan_handle, SENSET_UNIT, sensor[i].unit);
            SetCtrlVal(pan_handle, SENSET_CARNUM, sensor[i].card_num);
            SetCtrlVal(pan_handle, SENSET_CHANUM, sensor[i].channel_num);
            SetCtrlVal(pan_handle, SENSET_CALFAC, sensor[i].calib_factor);
            SetCtrlVal(pan_handle, SENSET_MAXRED, sensor[i].max_reading);
            if (i<1)
                SetCtrlAttribute(pan_handle, SENSET_PREV, 15, 0);
            if (i<max_num-1)
                SetCtrlAttribute(pan_handle, SENSET_NEXT, 15, 1);
            break;
        case SENSET_DONE:
            exit_flag = 1;
            break;
    }
}
RemovePopup(0);
UnloadPanel(pan_handle),

```

```

return 0;

} /* function SensorSetup ends here */

/*-----
| This function displays a popup window and shows volt, absolute |
| and relative reading for sensor specified by sensor_num. It   |
| also allows user to change zero values of that sensor.       |
|-----*/
int SensorTest()
{
    int ctrl_event = -1, exit_flag = 0;
    int pan_handle, index;
    int sensor_num = 0, prev_sensor_num = 0, max_sensor_num;

    max_sensor_num = system.total_sensors - 1;
    pan_handle = LoadPanel("FLEXCAT.UIR", SENTEST);
    SetCtrlVal(pan_handle, SENTEST_NAME, sensor[sensor_num].name);
    SetCtrlVal(pan_handle, SENTEST_UNIT, sensor[sensor_num].unit);
    InstallPopup(pan_handle);

    while (!exit_flag) {
        SensorRead(sensor_num);
        SetCtrlVal(pan_handle, SENTEST_VIN, sensor[sensor_num].volt);
        SetCtrlVal(pan_handle, SENTEST_RDG, sensor[sensor_num].reading);
        SetCtrlVal(pan_handle, SENTEST_ABSRDG, sensor[sensor_num].abs_reading);

        GetPopupEvent(0,&ctrl_event);
        if(ctrl_event == SENTEST_SNUM) {
            GetCtrlVal(pan_handle, SENTEST_SNUM, &sensor_num);
            if (sensor_num > max_sensor_num) {
                IssueWarning("Invalid sensor number.", 2.0);
                sensor_num = prev_sensor_num;
                SetCtrlVal(pan_handle, SENTEST_SNUM, sensor_num);
            }
            prev_sensor_num = sensor_num;

            SetCtrlVal(pan_handle, SENTEST_NAME, sensor[sensor_num].name);
            SetCtrlVal(pan_handle, SENTEST_UNIT, sensor[sensor_num].unit);
        }
        if(ctrl_event == SENTEST_ZER) {
            GetCtrlVal(pan_handle, SENTEST_ZER, &index);
            switch(index) {
                case 0:
                    SensorTakeZero(sensor_num);
                    break;
                case 1:
                    SensorRestoreZero(sensor_num);
                    break;
            }
        }
    }
}

```

```

        case 2:
            SensorTakeAbsZero(sensor_num);
            break;
        }
    }
    if(ctrl_event == SENCAL_EST_DONE)
        exit_flag = 1;

} /* while loop ends here */

RemovePopup(0);
UnloadPanel(pan_handle);

return 0;
} /* function SensorTest ends here */

/*-----
| This function displays a popup window and shows volt reading |
| for selected sensor and accepts reference values from user. |
| It then computes calibration factor from data and shows it in |
| graphical form. It also produces a file copy of calibration session. |
|-----*/
int SensorCalibrate()
{
    int ctrl_event = -1, exit_flag = 0;
    int pan_handle, indx = 0;
    int res_ctrl_event = -1, res_exit_flag = 0;
    int res_pan_handle, direction_flag = 0;
    int sensor_num = 0, prev_sensor_num = 0, max_sensor_num;
    char unit[15];
    float reference[30], signal[30], input_volt = 0.0, increment = 0.0;
    float calib_factor, r_squared, intercept;
    result_type result; /* user defined type */

    max_sensor_num = system.total_sensors - 1;
    pan_handle = LoadPanel("FLEXCAT.UIR", SENCAL);
    SetCtrlVal(pan_handle, SENCAL_NAME, sensor[sensor_num].name);
    SetCtrlVal(pan_handle, SENCAL_UNIT, sensor[sensor_num].unit);
    InstallPopup(pan_handle);

    while (!exit_flag) {
        SensorRead(sensor_num);
        SetCtrlVal(pan_handle, SENCAL_SIG, sensor[sensor_num].volt);
        GetPopupEvent(0, &ctrl_event);
        switch (ctrl_event) {
            case SENCAL_SNUM:
                GetCtrlVal(pan_handle, SENCAL_SNUM, &sensor_num);
                if (sensor_num > max_sensor_num) {
                    IssueWarning("Invalid sensor number.", 2.0);
                }
            }
        }
    }
}

```

```

        sensor_num = prev_sensor_num;
        SetCtrlVal(pan_handle, SENCAL_SNUM, sensor_num);
    }
    SetCtrlVal(pan_handle, SENCAL_NAME, sensor[sensor_num].name);
    SetCtrlVal(pan_handle, SENCAL_UNIT, sensor[sensor_num].unit);
    strcat(strcpy(unit, sensor[sensor_num].unit), "/V/V");
prev_sensor_num = sensor_num;
break;
case SENCAL_READ:
    GetCtrlVal(pan_handle, SENCAL_INC, &increment);
if(indx == 0) {
    reference[indx] = increment;
}
else {
    if(direction_flag == 0)
        reference[indx] = reference[indx-1] + increment;
    else
        reference[indx] = reference[indx-1] - increment;
}
signal[indx] = sensor[sensor_num].volt;
SetCtrlVal(pan_handle, SENCAL_REF, reference[indx]);
indx++;
    SetCtrlVal(pan_handle, SENCAL_RNUM, indx);
break;
case SENCAL_DEL:
    indx--;
    if(indx < 0)
        indx = 0;
    if(direction_flag == 0)
        reference[indx] = reference[indx-1] - increment;
    else
        reference[indx] = reference[indx-1] + increment;
    SetCtrlVal(pan_handle, SENCAL_REF, reference[indx]);
    SetCtrlVal(pan_handle, SENCAL_RNUM, indx);
break;
case SENCAL_VIEW:
    strcpy(plot.name, sensor[sensor_num].name);
    strcpy(plot.x_title, sensor[sensor_num].unit);
    strcpy(plot.y_title, "Volt");
    plot.min_x = plot.max_x = 0;
    plot.min_y = plot.max_y = 0;
    plot.x_logscale = plot.y_logscale = 0;
    plot.x_array[0] = reference;
    plot.y_array[0] = signal;
    plot.total_plots = 1;
    plot.total_points[0] = indx;
    GraphDisplay();
break;
case SENCAL_COMP:
    if(indx < 10) {
        IssueWarning("Not enough input data available.", 2.0);
        break;
    }

```

```

    }
    GetCtrlVal(pan_handle, SENCAL_VIN, &input_volt);
    if(input_volt == 0.0) {
        IssueWarning("Provide the value of input voltage.", 2.0);
        break;
    }
    if((input_volt < 0.0) || (input_volt > 15.0)) {
        IssueWarning("Invalid value for input voltage.", 2.0);
        break;
    }

    LinearRegression(reference, signal, indx, &result);

    if(result.slope == 0) { /* check if the slope is zero */
        IssueWarning("Invalid set of readings, try again!", -1.0);
        break;
    }
    calib_factor = input_volt/result.slope;
    r_squared = result.r_squared;
    intercept = result.intercept;
    res_pan_handle = LoadPanel("FLEXCAT.UIR", CALRES);
    SetCtrlVal(res_pan_handle, CALRES_CF, calib_factor);
    SetCtrlVal(res_pan_handle, CALRES_UNIT, unit);
    SetCtrlVal(res_pan_handle, CALRES_R2, r_squared);
    SetCtrlVal(res_pan_handle, CALRES_INT, intercept);
    InstallPopup(res_pan_handle);
    while (!res_exit_flag) {
        GetPopupEvent(0, &res_ctrl_event);
        if(res_ctrl_event == CALRES_OK)
            res_exit_flag = 1;
    }
    res_exit_flag = 0;
    RemovePopup(0);
    UnloadPanel(res_pan_handle);
    break;
case SENCAL_DIR:
    GetCtrlVal(pan_handle, SENCAL_DIR, &direction_flag);
    if(direction_flag == 0)
        SetCtrlAttribute(pan_handle, SENCAL_INC, 0, "Increment");
    else
        SetCtrlAttribute(pan_handle, SENCAL_INC, 0, "Decrement");
    break;
case SENCAL_DONE:
    exit_flag = 1;
    break;
}
} /* while loop ends here */
RemovePopup(0);
UnloadPanel(pan_handle);

return 0;
} /* function SensorCalibrate ends here */

```



```

/*-----
| This function performs linear regression on x and y arrays |
| each having n elements and returns the resulting slope, intercept |
| and correlation coefficient r squared through structure result. |
|-----*/
int LinearRegression(float *x, float *y, int n, result_type *result)
{
    register int i;
    double sum_x = 0, sum_x2 = 0;
    double sum_y = 0, sum_y2 = 0;
    double sum_xy = 0;
    double A = 0, B = 0, r2;
    double num_A, num_B, den_1, den_2, den_3;

    for(i=0; i<n; i++) {
        sum_x = sum_x + (double) x[i];
        sum_y = sum_y + (double) y[i];
        sum_x2 = sum_x2 + (double) (x[i]*x[i]);
        sum_y2 = sum_y2 + (double) (y[i]*y[i]);
        sum_xy = sum_xy + (double) (x[i]*y[i]);
    }

    den_1 = sum_x2 - sum_x*sum_x/n;
    den_2 = sum_y2 - sum_y*sum_y/n;
    num_B = sum_xy - sum_x*sum_y/n;
    num_A = (sum_x2*sum_y - sum_x*sum_xy)/n;

    B = num_B/den_1;
    A = num_A/den_1;

    den_3 = sqrt(den_1*den_2);
    r2 = num_B/den_3;

    result->slope = (float) B;
    result->intercept = (float) A;
    result->r_squared = (float) r2;

    return 0;
} /* function LinearRegression ends here */

```

```

/*-----
| This function takes a reading of A/D card and updates the |
| values of volt, reading and abs_reading for sensor specefied |
| by sensor_num in its argument. |
|-----*/

```

```

int SensorRead(int sensor_num)
{
    int card_num, channel_num;
    float involt;

    /* if the sensor number is an invalid number then return immediately */
    if(sensor_num >= system.total_sensors) {
        error.error_code = sensor_num;
        error.error_flag = 1;
        return 1;
    }
    card_num = sensor[sensor_num].card_num;
    channel_num = sensor[sensor_num].channel_num;
    involt = adcard[card_num].involt[channel_num];

    sensor[sensor_num].volt = involt;
    sensor[sensor_num].reading = sensor[sensor_num].calib_factor*(involt -
sensor[sensor_num].zero);
    sensor[sensor_num].abs_reading = sensor[sensor_num].calib_factor*(involt -
sensor[sensor_num].abs_zero);

    /* check if sensor limit has been exceeded */
    if((sensor[sensor_num].abs_reading > sensor[sensor_num].max_reading) || \
(sensor[sensor_num].abs_reading < -(sensor[sensor_num].max_reading))) {
        error.error_code = sensor_num;
        error.error_flag = 2;
        return 2;
    }
    return 0;
} /* function SensorRead ends here */

```

```

/*-----
| This function takes a reading and assigns it as zero reading      |
| for the sensor specified by sensor_num.                          |
|-----*/

```

```

int SensorTakeZero(int sensor_num)
{
    int card_num, channel_num;
    float involt;

    /* if the sensor number is an invalid number then return immediately */
    if(sensor_num >= system.total_sensors) {
        error.error_code = sensor_num;
        error.error_flag = 1;
        return 1;
    }
    card_num = sensor[sensor_num].card_num;
    channel_num = sensor[sensor_num].channel_num;
    involt = adcard[card_num].involt[channel_num];

```

```

    sensor[sensor_num].zero = involt;

    return 0;
} /* function SensorTakeZero ends here */

/*-----
| This function takes a reading and assigns it as absolute zero   |
| and zero for the sensor specified by sensor_num.                |
|-----*/
int SensorTakeAbsZero(int sensor_num)
{
    int card_num, channel_num;
    float involt;

    /* if the sensor number is an invalid number then return immediately */
    if(sensor_num >= system.total_sensors) {
        error.error_code = sensor_num;
        error.error_flag = 1;
        return 1;
    }
    card_num = sensor[sensor_num].card_num;
    channel_num = sensor[sensor_num].channel_num;
    involt = adcard[card_num].involt[channel_num];
    sensor[sensor_num].abs_zero = involt;
    sensor[sensor_num].zero = involt;

    return 0;
} /* function SensorTakeAbsZero ends here */

/*-----
| This function restores the value of zero to absolute zero for   |
| the sensor specified by sensor_num.                             |
|-----*/
int SensorRestoreZero(int sensor_num)
{
    /* if the sensor number is an invalid number then return immediately */
    if(sensor_num >= system.total_sensors) {
        error.error_code = sensor_num;
        error.error_flag = 1;
        return 1;
    }
    sensor[sensor_num].zero = sensor[sensor_num].abs_zero;
    return 0;
} /* function SensorRestoreZero ends here */

```

```

/*-----
| This function takes absolute zeros for all the sensors in |
| the system. |
|-----*/

```

```

int SensorZeroAll(void)
{
    int ctrl_event = -1, exit_flag = 0, i = 0;
    int pan_handle;
    char *message = "\n"
        " This will take a set of \n"
        " readings and assign them \n"
        " as absolute zeros for all \n"
        " sensors. ";

    pan_handle = LoadPanel("FLEXCAT.UIR", ZEROALL);
    SetCtrlVal(pan_handle, ZEROALL_TEXT, message);
    InstallPopup(pan_handle);

    while (!exit_flag) {
        GetPopupEvent(0, &ctrl_event);

        if (ctrl_event == ZEROALL_OK) {
            for(i=0; i<system.total_sensors; i++)
                SensorTakeAbsZero(i);
            exit_flag = 1;
        }

        if (ctrl_event == ZEROALL_CAN)
            exit_flag = 1;
    }

    RemovePopup(0);
    UnloadPanel(pan_handle);

    return 0;
}

```

```

/*-----
| This set of functions form the interface between user and |
| system modules and help talk with various sensors. |
|-----*/

```

```

float SensorGetVolt(int sensor_num)
{
    return sensor[sensor_num].voit;
}

```

```
float SensorGetReading(int sensor_num)
{
    return sensor[sensor_num].reading;
}
```

```
float SensorGetAbsReading(int sensor_num)
{
    return sensor[sensor_num].abs_reading;
}
```

```
/*-----
| This function displays a pop up window and obtains from user |
| values of parameters contained in structure servo.           |
|-----*/
```

```
int ServoSetup(void)
{
    int ctrl_event = -1, exit_flag = 0, i = 0;
    int pan_handle, max_num;

    max_num = system.total_servos;

    pan_handle = LoadPanel("FLEXCAT.UIR", SERSET);
    InstallPopup(pan_handle);
    SetCtrlVal(pan_handle, SERSET_SERNUM, i);
    SetCtrlVal(pan_handle, SERSET_NAME, servo[i].name);
    SetCtrlVal(pan_handle, SERSET_CARNUM, servo[i].card_num);
    SetCtrlVal(pan_handle, SERSET_CHANUM, servo[i].channel_num);
    SetCtrlVal(pan_handle, SERSET_RESET, servo[i].reset_flag);
    SetCtrlVal(pan_handle, SERSET_MAXVOL, servo[i].max_volt);
    SetCtrlVal(pan_handle, SERSET_MINVOL, servo[i].min_volt);
    if(max_num == 1)
        SetCtrlAttribute(pan_handle, SERSET_NEXT, 15, 0);

    while(!exit_flag)
    {
        GetCtrlVal(pan_handle, SERSET_NAME, servo[i].name);
        GetCtrlVal(pan_handle, SERSET_CARNUM, &servo[i].card_num);
        GetCtrlVal(pan_handle, SERSET_CHANUM, &servo[i].channel_num);
        GetCtrlVal(pan_handle, SERSET_RESET, &servo[i].reset_flag);
        GetCtrlVal(pan_handle, SERSET_MAXVOL, &servo[i].max_volt);
        GetCtrlVal(pan_handle, SERSET_MINVOL, &servo[i].min_volt);

        GetPopupEvent(0, &ctrl_event);
        switch (ctrl_event) {
            case SERSET_NEXT:
                i = i+1;
                SetCtrlVal(pan_handle, SERSET_SERNUM, i);
        }
    }
}
```

```

        SetCtrlVal(pan_handle, SERSET_NAME, servo[i].name);
        SetCtrlVal(pan_handle, SERSET_CARNUM, servo[i].card_num);
        SetCtrlVal(pan_handle, SERSET_CHANUM, servo[i].channel_num);
    SetCtrlVal(pan_handle, SERSET_RESET, servo[i].reset_flag);
        SetCtrlVal(pan_handle, SERSET_MAXVOL, servo[i].max_volt);
        SetCtrlVal(pan_handle, SERSET_MINVOL, servo[i].min_volt);
    if (i>0)
        SetCtrlAttribute(pan_handle, SERSET_PREV, 15, 1);
    if (i>=max_num-1)
        SetCtrlAttribute(pan_handle, SERSET_NEXT, 15, 0);
    break;
    case SERSET_PREV:
        i = i-1;
        SetCtrlVal(pan_handle, SERSET_SERNUM, i);
        SetCtrlVal(pan_handle, SERSET_NAME, servo[i].name);
        SetCtrlVal(pan_handle, SERSET_CARNUM, servo[i].card_num);
        SetCtrlVal(pan_handle, SERSET_CHANUM, servo[i].channel_num);
    SetCtrlVal(pan_handle, SERSET_RESET, servo[i].reset_flag);
        SetCtrlVal(pan_handle, SERSET_MAXVOL, servo[i].max_volt);
        SetCtrlVal(pan_handle, SERSET_MINVOL, servo[i].min_volt);
    if (i<1)
        SetCtrlAttribute(pan_handle, SERSET_PREV, 15, 0);
    if (i<max_num-1)
        SetCtrlAttribute(pan_handle, SERSET_NEXT, 15, 1);
    break;
    case SERSET_DONE:
        exit_flag = 1;
        break;
    }
}
RemovePopup(0);
UnloadPanel(pan_handle);
return 0;
} /* function ServoSetup ends here */

```

```

/*-----
| This function displays a pop up window and allow the user to |
| manipulate servo plant directly. |
-----*/

```

```

int ServoTest(void)
{
    int ctrl_event = -1, exit_flag = 0;
    int pan_handle;
    int servo_num = 0, prev_servo_num = 0, max_servo_num;
    float vout = 0.0;

    max_servo_num = system.total_servos - 1;
    pan_handle = LoadPanel("FLEXCAT.UIR", SERTEST);

```

```

SetCtrlVal(pan_handle, SERTEST_SERNO, servo_num);
SetCtrlVal(pan_handle, SERTEST_NAME, servo[servo_num].name);
InstallPopup(pan_handle);
while (!exit_flag) {
    GetPopupEvent(0,&ctrl_event);
    if(ctrl_event == SERTEST_SERNO) {
        GetCtrlVal(pan_handle, SERTEST_SERNO, &servo_num);
        if(servo_num > max_servo_num) {
            IssueWarning("Invalid servo number.", 2.0);
            servo_num = prev_servo_num;
            SetCtrlVal(pan_handle, SERTEST_SERNO, servo_num);
        }
        SetCtrlVal(pan_handle, SERTEST_NAME, servo[servo_num].name);
        prev_servo_num = servo_num;
    }
    if(ctrl_event == SERTEST_STOP) {
        vout = 0.0;
        ServoWrite(servo_num, vout);
        SetCtrlVal(pan_handle, SERTEST_VOUT, vout);
    }
    if(ctrl_event == SERTEST_UP) {
        vout = vout + 0.1;
        if(vout > servo[servo_num].max_volt)
            vout = servo[servo_num].max_volt;
        ServoWrite(servo_num, vout);
        SetCtrlVal(pan_handle, SERTEST_VOUT, vout);
    }
    if(ctrl_event == SERTEST_DN) {
        vout = vout - 0.1;
        if(vout < servo[servo_num].min_volt)
            vout = servo[servo_num].min_volt;
        ServoWrite(servo_num, vout);
        SetCtrlVal(pan_handle, SERTEST_VOUT, vout);
    }
    if(ctrl_event == SERTEST_DONE)
        exit_flag = 1;
}

RemovePopup(0);
UnloadPanel(pan_handle);
return 0;
} /* function ServoTest ends here */

```

```

/*-----
| This function writes voltage value to the servo specified by |
| servo_num. |
|-----*/

```

```

int ServoWrite(int servo_num, float volt)
{
    /* if the servo number is an invalid number then return immediately */
    if(servo_num >= system.total_servos) {
        error.error_code = servo_num;
        error.error_flag = 3;
        return 1;
    }
    /* check for the limits */
    if((volt>servo[servo_num].max_volt) || (volt<servo[servo_num].min_volt)) {
        error.error_code = servo_num;
        error.error_flag = 4;
        return 2;
    }

    DacardWrite(servo[servo_num].card_num, servo[servo_num].channel_num, volt);

    return 0;
} /* function ServoWrite ends here */

```

```

/*-----
| This function writes zero volt to the servo specified by      |
| servo_num.                                                    |
|-----*/

```

```

int ServoReset(int servo_num)
{
    /* if the servo number is an invalid number then return immediately */
    if(servo_num >= system.total_servos) {
        error.error_code = servo_num;
        error.error_flag = 3;
        return 1;
    }

    ServoWrite(servo_num, 0.0); /* send zero volt */

    return 0;
} /*function servo reset ends here */

```

```

/*-----
| This function writes zero volt to all servos which have their |
| reset flag set to 1.                                          |
|-----*/

```

```

int ServoResetAll(void)
{

```



```

int i;

for(i = 0; i < system.total_servos; i++)
    if(servo[i].reset_flag == 1)
        ServoReset(i);

return 0;
}

/*-----
| This function displays a pop up window and obtains from user |
| values of parameters contained in structure loop.           |
|-----*/
int LoopSetup(void)
{
    int ctrl_event = -1, exit_flag = 0, i = 0;
    int pan_handle, max_num;

    max_num = system.total_loops;
    pan_handle = LoadPanel("FLEXCAT.UIR", LOOPSET);
    SetCtrlVal(pan_handle, LOOPSET_SERNUM, loop[i].serial_num);
    SetCtrlVal(pan_handle, LOOPSET_NAME, loop[i].name);
    SetCtrlVal(pan_handle, LOOPSET_SENNUM, loop[i].sensor_num);
    SetCtrlVal(pan_handle, LOOPSET_SERVNUM, loop[i].servo_num);
    SetCtrlVal(pan_handle, LOOPSET_CYCNT, loop[i].cycle_count);
    SetCtrlVal(pan_handle, LOOPSET_KP, loop[i].kp);
    SetCtrlVal(pan_handle, LOOPSET_KV, loop[i].kv);
    SetCtrlVal(pan_handle, LOOPSET_KI, loop[i].ki);

    InstallPopup(pan_handle);

    if(max_num == 1)
        SetCtrlAttribute(pan_handle, LOOPSET_NEXT, 15, 0);

    while(!exit_flag)
    {
        GetCtrlVal(pan_handle, LOOPSET_SENNUM, &loop[i].sensor_num);
        if (loop[i].sensor_num > (system.total_sensors-1)) {
            IssueWarning("Invalid sensor number.", 2.0);
            loop[i].sensor_num = 0;
            SetCtrlVal(pan_handle, LOOPSET_SENNUM, loop[i].sensor_num);
        }

        GetCtrlVal(pan_handle, LOOPSET_SERVNUM, &loop[i].servo_num);
        if (loop[i].servo_num > (system.total_servos - 1)) {
            IssueWarning("Invalid servo number.", 2.0);
            loop[i].servo_num = 0;
            SetCtrlVal(pan_handle, LOOPSET_SERVNUM, loop[i].servo_num);
        }
    }
}

```

```

GetCtrlVal(pan_handle, LOOPSET_NAME, loop[i].name);
GetCtrlVal(pan_handle, LOOPSET_CYCNT, &loop[i].cycle_count);
GetCtrlVal(pan_handle, LOOPSET_KP, &loop[i].kp);
GetCtrlVal(pan_handle, LOOPSET_KV, &loop[i].kv);
GetCtrlVal(pan_handle, LOOPSET_KI, &loop[i].ki);
loop[i].serial_num = i;

GetPopupEvent(0,&ctrl_event);
switch (ctrl_event) {
    case LOOPSET_NEXT:
        i = i+1;
        SetCtrlVal(pan_handle, LOOPSET_SERNUM, i);
        SetCtrlVal(pan_handle, LOOPSET_NAME, loop[i].name);
        SetCtrlVal(pan_handle, LOOPSET_SENNUM, loop[i].sensor_num);
        SetCtrlVal(pan_handle, LOOPSET_SERVNUM, loop[i].servo_num);
        SetCtrlVal(pan_handle, LOOPSET_CYCNT, loop[i].cycle_count);
        SetCtrlVal(pan_handle, LOOPSET_KP, loop[i].kp);
        SetCtrlVal(pan_handle, LOOPSET_KV, loop[i].kv);
        SetCtrlVal(pan_handle, LOOPSET_KI, loop[i].ki);
        if (i>0)
            SetCtrlAttribute(pan_handle, LOOPSET_PREV, 15, 1);
        if (i>=max_num-1)
            SetCtrlAttribute(pan_handle, LOOPSET_NEXT, 15, 0);
        break;
    case LOOPSET_PREV:
        i = i-1;
        SetCtrlVal(pan_handle, LOOPSET_SERNUM, i);
        SetCtrlVal(pan_handle, LOOPSET_SENNUM, loop[i].sensor_num);
        SetCtrlVal(pan_handle, LOOPSET_NAME, loop[i].name);
        SetCtrlVal(pan_handle, LOOPSET_SERVNUM, loop[i].servo_num);
        SetCtrlVal(pan_handle, LOOPSET_CYCNT, loop[i].cycle_count);
        SetCtrlVal(pan_handle, LOOPSET_KP, loop[i].kp);
        SetCtrlVal(pan_handle, LOOPSET_KV, loop[i].kv);
        SetCtrlVal(pan_handle, LOOPSET_KI, loop[i].ki);
        if (i<1)
            SetCtrlAttribute(pan_handle, LOOPSET_PREV, 15, 0);
        if (i<max_num-1)
            SetCtrlAttribute(pan_handle, LOOPSET_NEXT, 15, 1);
        break;
    case LOOPSET_DONE:
        exit_flag = 1;
        break;
}
}
RemovePopup(0);
UnloadPanel(pan_handle);
return 0;
} /* function LoopSetup ends here */

```

```

/*-----
| This function displays sensor readings and servo output for |
| selected loop and allows experimental determination of loop |
| calibration coefficient kp. |
|-----*/
int LoopCalibrate(void)
{
    int ctrl_event = -1, exit_flag = 0;
    int pan_handle, status, index;
    int loop_num = 0, max_loop_num;
    float vout = 0, reading;

    max_loop_num = system.total_loops - 1;
    pan_handle = LoadPanel("FLEXCAT.UIR", LOOPCAL);
    SetCtrlVal(pan_handle, LOOPCAL_LOOPNUM, loop_num);
    SetCtrlVal(pan_handle, LOOPCAL_NAME, loop[loop_num].name);
    SetCtrlVal(pan_handle, LOOPCAL_SERNAME, servo[loop[loop_num].servo_num].name);
    SetCtrlVal(pan_handle, LOOPCAL_SENNAME, sensor[loop[loop_num].sensor_num].name);
    SetCtrlVal(pan_handle, LOOPCAL_UNIT, sensor[loop[loop_num].sensor_num].unit);
    SetCtrlAttribute(pan_handle, LOOPCAL_LOOPNUM, 13, max_loop_num);
    InstallPopup(pan_handle);

    while (!exit_flag) {
        reading = sensor[loop[loop_num].sensor_num].abs_reading;
        SetCtrlVal(pan_handle, LOOPCAL_RDG, reading);
        GetPopupEvent(0, &ctrl_event);
        switch(ctrl_event) {
            case LOOPCAL_LOOPNUM:
                status = GetCtrlVal(pan_handle, LOOPCAL_LOOPNUM, &loop_num);
                if (status == -14) {
                    IssueWarning("Invalid loop number.", 2.0);
                    loop_num = 0;
                    SetCtrlVal(pan_handle, LOOPCAL_LOOPNUM, loop_num);
                }
                SetCtrlVal(pan_handle, LOOPCAL_NAME, loop[loop_num].name);
                SetCtrlVal(pan_handle, LOOPCAL_SERNAME,
servo[loop[loop_num].servo_num].name);
                SetCtrlVal(pan_handle, LOOPCAL_SENNAME,
sensor[loop[loop_num].sensor_num].name);
                SetCtrlVal(pan_handle, LOOPCAL_UNIT, sensor[loop[loop_num].sensor_num].unit);
                break;
            case LOOPCAL_RUN:
                GetCtrlVal(pan_handle, LOOPCAL_VOUT, &vout);
                if ((vout > servo[loop[loop_num].servo_num].max_volt) \
|| (vout < servo[loop[loop_num].servo_num].min_volt)) {
                    IssueWarning("Invalid output voltage.", 2.0);
                    break;
                }
                ServoWrite(loop[loop_num].servo_num, vout);
                SetCtrlVal(pan_handle, LOOPCAL_VOUT, vout);
        }
    }
}

```

```

break;
case LOOPCAL_STOP:
    vout = 0;
    ServoWrite(loop[loop_num].servo_num, vout);
    SetCtrlVal(pan_handle, LOOPCAL_VOUT, vout);
break;
case LOOPCAL_DONE:
    exit_flag = 1;
break;
} /* switch statement ends here */

} /* while loop ends here */

RemovePopup(0);
UnloadPanel(pan_handle);
return 0;
} /* function LoopCalibrate ends here */

```

```

/*-----
| This function saves the data structure of all objects on hard |
| disk which can be later retrived by function DatabaseLoad |
|-----*/

```

```

int DatabaseSave(void)
{
    FILE *data_file;
    register int i;

    data_file = fopen("FLEXCAT.DBS", "wb"); /* open database file for writing */

    if (data_file == NULL) { /* if error opening the file on hard disk */
        IssueWarning("Error saving data file on Hard Disk.", 2.0);
        return 1;
    }

    fwrite(&system, sizeof(system), 1, data_file);
    for (i=0; i<system.total_adcards; i++)
        fwrite(&adcard[i], sizeof(adcard), 1, data_file);
    for (i=0; i<system.total_dacards; i++)
        fwrite(&dacard[i], sizeof(dacard), 1, data_file);
    for (i=0; i<system.total_sensors; i++)
        fwrite(&sensor[i], sizeof(sensor), 1, data_file);
    for (i=0; i<system.total_servos; i++)
        fwrite(&servo[i], sizeof(servo), 1, data_file);
    for (i=0; i<system.total_loops; i++)
        fwrite(&loop[i], sizeof(loop), 1, data_file);
    for (i=0; i<system.total_graphs; i++)
        fwrite(&graph[i], sizeof(graph), 1, data_file);
}

```

```

flush(data_file);    /* flush the data stream */

fclose(data_file);  /* close the file */

return 0;

} /* end function DatabaseSave */

```

```

/*-----
| This function retrieves the data structure of all objects from      |
| hard disk saved earlier by function DatabaseSave.                  |
|-----*/

```

```

int DatabaseLoad(void)
{
    FILE *data_file;
    register int i;

    data_file = fopen("FLEXCAT.DBS", "rb"); /* open database file for reading */
    if (data_file == NULL) /* if error opening the file on the hard disk */
        return 1;

    fread(&system, sizeof(system), 1, data_file);

    for (i=0; i<system.total_adcards; i++)
        fread(&adcard[i], sizeof(adcard), 1, data_file);
    for (i=0; i<system.total_dacards; i++)
        fread(&dacard[i], sizeof(dacard), 1, data_file);
    for (i=0; i<system.total_sensors; i++)
        fread(&sensor[i], sizeof(sensor), 1, data_file);
    for (i=0; i<system.total_servos; i++)
        fread(&servo[i], sizeof(servo), 1, data_file);
    for (i=0; i<system.total_loops; i++)
        fread(&loop[i], sizeof(loop), 1, data_file);
    for (i=0; i<system.total_graphs; i++)
        fread(&graph[i], sizeof(graph), 1, data_file);

    fclose(data_file);

    return 0;

} /* end function DatabaseLoad */

```

```

/*-----
| This function displays a box containing an error message,        |
| generates a beep and waits for user to press OK button.        |
|-----*/

```

```

-----*/
int IssueWarning(char *message, float interval)
{
    int pan_handle, ctrl_event;
    int exit_flag = 0;
    float init_time = 0.0, elapsed_time = 0.0;

    pan_handle = LoadPanel("FLEXCAT.UIR", WARN);
    SetCtrlVal(pan_handle, WARN_MESSAGE, message);
    InstallPopup(pan_handle); /* display the warning message */
    init_time = GetSystemTime();
    sound(510); /* generate sound at 510 Hz */
    while (!exit_flag) {
        elapsed_time = GetSystemTime() - init_time;
        if(elapsed_time >= 0.1)
            nosound(); /* turn off the sound */
        if(interval >= 0.0)
            if(elapsed_time >= interval)
                exit_flag = 1;
        GetPopupEvent(0,&ctrl_event);
        if (ctrl_event == WARN_OK) {
            nosound(); /* just in case timer fails */
            exit_flag = 1;
        }
    }
    RemovePopup(0);
    UnloadPanel(pan_handle);
    return 0;
} /* function IssueWarning ends here */

```

```

/*-----
| This function simulates BASIC statement 'WAIT', however it has |
| a maximum limit of 1 sec after which it returns with a time out |
| error. |
-----*/

```

```

int Wait(int adress, unsigned char i, unsigned char j)
{
    unsigned char testbyte;
    long start_time;

    start_time = biostime(0, 0); /* obtain initial time */
    do {
        if ((biostime(0, 0) - start_time) > 18) /* exit if elapsed time > 1 sec */
            return(1);
        testbyte = inportb(adress); /* read test byte */
    } while (((testbyte^j)&i) == 0); /* test for the bit pattern */
    return(0);
} /* function Wait ends here */

```

```

/*-----
| This is substitute function for LabWindows keyboard handler |
| It is executed every 150 ms in the background.             |
|-----*/
void LWKbdHandler(void)
{
    union RECS regs;

    int86(0x45, &regs, &regs); /* execute software interrupt# hex 45 */

    return;
}

/*-----
| This function returns time in seconds elapsed since the    |
| last time the system was started.                         |
|-----*/
float GetSystemTime(void)
{
    long unsigned timer_count;

    timer_count = t_check_timer(calander.timer_num, T_TOTAL);

    return ((float)timer_count)/1000.0;;
}

float AdcardRead(int card_num, int channel_num)
{
    int adres;
    unsigned int byte1, byte2, byte3;
    unsigned long int count;
    float involt;

    adres = adcard[card_num].base_adres;

    /* switch multiplexer to channel 0 */
    outportb(adres + 4, channel_num);

    /* start conversion */
    outportb(adres, CNV);
    Wait(adres, 1, 1);

    byte1 = inportb(adres + 1);
    byte2 = inportb(adres + 2);
    byte3 = inportb(adres + 3);
}

```

```

    outportb(adres, EOI);    /* clear data ready flag */

    count = byte1 + 256*byte2 + 65536*byte3;
    involt = (float) count*10.0/adcard[card_num].max_count - 5.0;
    return involt;
}

void AdcardDrive0(void)
{
    unsigned int byte1, byte2, byte3;
    unsigned long count;

    static int previous_channel = 0;
    static int channel_num = 0;

    /* read the card and store values in structure reading */
    byte1 = inportb(adcard[0].base_address + 1);
    byte2 = inportb(adcard[0].base_address + 2);
    byte3 = inportb(adcard[0].base_address + 3);

    outportb(adcard[0].base_address, EOI);    /* clear data ready flag */

    count = byte1 + 256*byte2 + 65536*byte3;
    adcard[0].involt[previous_channel] = (float)(count*10.0/adcard[0].max_count - 5.0);

    /* switch multiplexer and start conversion */

    outportb(adcard[0].base_address + 4, channel_num);
    outportb(adcard[0].base_address, CNV);

    /* increment channel number */
    previous_channel = channel_num;
    channel_num++;

    /* reset the channel number if it exceeds the max. value */
    if (channel_num > adcard[0].total_channel-1)
        channel_num = 0;

} /* Function AdcardDrive0 ends here */

void AdcardDrive1(void)
{
    unsigned int byte1, byte2, byte3;
    unsigned long count;

    static int previous_channel = 0;
    static int channel_num = 0;

```



```

/* read the card and store values in structure reading */
byte1 = inportb(adcard[1].base_address + 1);
byte2 = inportb(adcard[1].base_address + 2);
byte3 = inportb(adcard[1].base_address + 3);

outportb(adcard[1].base_address, EOI); /* clear data ready flag */

count = byte1 + 256*byte2 + 65536*byte3;
adcard[1].involt[previous_channel] = (float)(count*10.0/adcard[1].max_count - 5.0);

/* switch multiplexer and start conversion */
outportb(adcard[1].base_address + 4, channel_num);
outportb(adcard[1].base_address, CNV);

/* increment channel number */
previous_channel = channel_num;
channel_num++;

/* reset the channel number if it exceeds the max. value */
if (channel_num > adcard[1].total_channel-1)
    channel_num = 0;
} /* Function AdcardDrive1 ends here */

```

```

void AdcardDrive2(void)
{
    unsigned int byte1, byte2, byte3;
    unsigned long count;

    static int previous_channel = 0;
    static int channel_num = 0;

    /* read the card and store values in structure reading */
    byte1 = inportb(adcard[2].base_address + 1);
    byte2 = inportb(adcard[2].base_address + 2);
    byte3 = inportb(adcard[2].base_address + 3);

    outportb(adcard[2].base_address, EOI); /* clear data ready flag */

    count = byte1 + 256*byte2 + 65536*byte3;
    adcard[2].involt[previous_channel] = (float)(count*10.0/adcard[2].max_count - 5.0);

    /* switch multiplexer and start conversion */
    outportb(adcard[2].base_address + 4, channel_num);
    outportb(adcard[2].base_address, CNV);

    /* increment channel number */
    previous_channel = channel_num;
}

```

```

channel_num++;

/* reset the channel number if it exceeds the max. value */
if (channel_num > adcard[2].total_channel-1)
    channel_num = 0;

} /* Function AdcardDrive2 ends here */

int DacardWrite(int card_num, int channel_num, float volt)
{
    int address;
    float range;
    unsigned int vbits;

    address = dacard[card_num].base_address;
    range = dacard[card_num].range[channel_num];

    /* Check for unipolar/bipolar Setting */
    if(dacard[card_num].polarity_code[channel_num] == 0)
        vbits = (volt + range/2.0)*dacard[card_num].max_count/range;
    else
        vbits = volt*dacard[card_num].max_count/range;

    output(address, vbits);          /* Send voltage */
    outputb(address + 2, channel_num); /* Select channel */

    return 0;
}

int SetUserFlag(void)
{
    control.user_run_flag++;
    return control.user_run_flag;
}

int ClearUserFlag(void)
{
    control.user_run_flag--;
    return control.user_run_flag;
}

int GetEventPanel(void)
{
    return control.event_panel;
}

```

```
int GetEventControl(void)
{
    return control.event_control;
}
```

```
int GetSystemStatus(void)
{
    return control.system_run_flag;
}
```

```
int ScreenPrint(void)
{
    int status;
    char message[40], number[5];

    status = ConfigurePrinter("psfile", 1, 5.0, 4.0, 1);
    if(status < 0) {
        strcpy(message, "Error printing screen ");
        itoa(status, number, 10);
        strcat(message, number);
        IssueWarning(message, 2.0);
        return 1;
    }
    status = OutputScreen(0, "screen.dmp");
    if(status < 0) {
        strcpy(message, "Error printing screen ");
        itoa(status, number, 10);
        strcat(message, number);
        IssueWarning(message, 2.0);
        return 2;
    }
    return 0;
}
```



#### D.4.4. File APSR.C

```
/*-----  
| This is APSR module. In conjunction with the SYSTEM  
| module,it controls all feedback loops in APSR test,  
| dispalys data in numerical and graphical form, and  
| performs data acquisition.  
| Copy right Samir Chauhan, MIT Geotechnical Laboratory, 1994. |  
-----*/  
  
#include "support.h" /* this file must be included for using system services */  
#include "apsr.h" /* GUI header file */  
  
#define AIR_PR 0 /* these definations are used as loop indices */  
#define PIST_1 1  
#define PIST_2 2  
#define PIST_3 3  
#define PIST_4 4  
#define TOP_WALL 5  
#define BOT_WALL 6  
#define REINFORCE 7  
  
static int ProcessSigma3(int event_control_id); /* all functions local to this module are declared  
here */  
static int ProcessSigma2(int event_control_id);  
static int ProcessSigma1(int event_control_id);  
static int ProcessReinf(int event_control_id);  
static int ProcessGraph(int event_control_id);  
static int ProcessErrors(void);  
  
static int ControlAirpressure(void);  
static int ControlPlatforms(void);  
static int ControlSidewalls(void);  
static int ControlReinforcement(void);  
static int ExitStrainLoop(void);  
static int SaveResults(char *filename);  
static int StopAllMotors(void);  
  
static int main_panel, sigma1_panel, sigma2_panel; /* all variables global to this module are  
declared here */  
static int sigma3_panel, reinf_panel, misc_panel;  
  
static int pressure_ctrl_flag = 0; /* these flags control the execution of feed back control routines */  
static int sidewall_ctrl_flag = 0;  
static int reinf_ctrl_flag = 0;  
static int strain_ctrl_flag = 0;  
static int global_error_flag = 0;  
static int result_file_flag = 0;  
  
static float target_airpressure, target_strainrate, actual_strainrate;  
static float max_displacement, max_stress;
```

```

/* these global variables are used do hold data for graphing */
static float sigma1[200], sigma12[200], sigma2[200], sigma3[200];
static float epsilon1[200], epsilon2[200], epsilon3L[200], epsilon3C[200];
static float epsilon_vol[200], R_value[200], b_value[200], q[200];
static float load_cell[200], ir_sensor[200], time_array[200];
static int index = 0;

static const float sample_length = 570; /* dimensions of APSR sample */
static const float sample_width = 450;
static const float sample_height = 150;

```

```

/*-----
| This is the first user function to be executed. It is          |
| executed only once at the beginning of the program. It loads  |
| and displays all APSR panels from file APSR.UIR.              |
|-----*/

```

```

int MeFirst(void)
{
    main_panel = LoadPanel("APSR.UIR", MAIN); /* load all static panels */
    sigma1_panel = LoadPanel("APSR.UIR", SIGMA1);
    sigma2_panel = LoadPanel("APSR.UIR", SIGMA2);
    sigma3_panel = LoadPanel("APSR.UIR", SIGMA3);
    reinf_panel = LoadPanel("APSR.UIR", REINF);
    misc_panel = LoadPanel("APSR.UIR", MISC);
    DisplayPanel(main_panel); /* display all static panels */
    DisplayPanel(sigma1_panel);
    DisplayPanel(sigma2_panel);
    DisplayPanel(sigma3_panel);
    DisplayPanel(reinf_panel);
    DisplayPanel(misc_panel);

    return 0;
} /* function MeFirst ends here */

```

```

/*-----
| This function is part of the infinite loop the program spends |
| most of its time in. Therefore it gets executed all the time. |
| Any thing put in this function will illicit prompt user      |
| response but at a priority lower than that of UserBackground. |
|-----*/

```

```

int UserForeground(void)
{
    int event_panel_id, event_control_id;

```

```

float platform_pr1, platform_pr2;
float sigma2t, sigma2b, sigma1, sigma2, sigma3;
float disp11, disp12, strain1, strain2, strain3, R, b, q;
float reinf_disp, reinf_load;

/* compute and display all test parameters */
SetCtrlVal(sigma1_panel, SIGMA1_DCDT1, SensorGetAbsReading(1));
SetCtrlVal(sigma1_panel, SIGMA1_DCDT2, SensorGetAbsReading(2));
SetCtrlVal(sigma1_panel, SIGMA1_DCDT3, SensorGetAbsReading(3));
SetCtrlVal(sigma1_panel, SIGMA1_DCDT4, SensorGetAbsReading(4));

platform_pr1 = (SensorGetAbsReading(5) + SensorGetAbsReading(6))*0.19 - 5.5;
platform_pr2 = (SensorGetAbsReading(7) + SensorGetAbsReading(8))*0.19 - 5.5;
disp11 = 0.5*(SensorGetAbsReading(1) + SensorGetAbsReading(2));
disp12 = 0.5*(SensorGetAbsReading(3) + SensorGetAbsReading(4));
strain1 = 100.0*(disp11 + disp12)/sample_length;
SetCtrlVal(sigma1_panel, SIGMA1_AXSTRS1, platform_pr1);
SetCtrlVal(sigma1_panel, SIGMA1_AXSTRS2, platform_pr2);
SetCtrlVal(sigma1_panel, SIGMA1_STRUCTUREL, actual_strainrate);
SetCtrlVal(sigma1_panel, SIGMA1_AXSTRN, strain1);

sigma2t = SensorGetAbsReading(11);
sigma2b = SensorGetAbsReading(12);
strain2 = 100.0*(SensorGetReading(9) + SensorGetReading(10))/sample_height;
SetCtrlVal(sigma2_panel, SIGMA2_TOP, sigma2t);
SetCtrlVal(sigma2_panel, SIGMA2_BOT, sigma2b);
SetCtrlVal(sigma2_panel, SIGMA2_PSTRN, strain2);

strain3 = 50.0*(SensorGetReading(13)+SensorGetReading(14))/sample_width;
sigma3 = SensorGetAbsReading(0);

sigma1 = 0.5*(platform_pr1 + platform_pr2);
sigma2 = 0.5*(sigma2t + sigma2b);

if((sigma1-sigma3) != 0.0)          /* check for divide by zero */
    b = (sigma2 - sigma3)/(sigma1 - sigma3);
if((b>0.0)||(b<2.0))
    SetCtrlVal(sigma2_panel, SIGMA2_BVAL, b);

SetCtrlVal(sigma3_panel, SIGMA3_PACTUAL, sigma3);
SetCtrlVal(sigma3_panel, SIGMA3_LSTRN, strain3);
if(sigma3 != 0.0)          /* check for divide by zero */
    R = sigma1/(sigma3);
if((R>0.0)||(R<10.0))
    SetCtrlVal(sigma3_panel, SIGMA3_RVAL, R);

reinf_disp = SensorGetReading(16);
reinf_load = SensorGetAbsReading(17);
SetCtrlVal(reinf_panel, REINF_LOAD, reinf_load);
SetCtrlVal(reinf_panel, REINF_REFPOS, reinf_disp);

ProcessErrors();          /* process global errors */

```

```

event_panel_id = GetEventPanel();    /* get screen event specifications */
event_control_id = GetEventControl();

/* call event processing functions if event has occurred on a particular panel */
if(event_panel_id == sigma3_panel)
    ProcessSigma3(event_control_id);
if(event_panel_id == sigma2_panel)
    ProcessSigma2(event_control_id);
if(event_panel_id == sigma1_panel)
    ProcessSigma1(event_control_id);
if(event_panel_id == reinf_panel)
    ProcessReinf(event_control_id);
if(event_panel_id == misc_panel)
    ProcessGraph(event_control_id);

return 0;

} /* function UserForeground ends here */

```

```

/*-----
| This function runs in the foreground and respond to the status |
| of a global error flag. This error flag is used by the       |
| back ground functions to convey information to the user.     |
|-----*/

```

```

int ProcessErrors(void)
{
    switch (global_error_flag) {
        case 1:
            SetCtrlVal(sigma3_panel, SIGMA3_RUN, 0);
            IssueWarning("Cannot maintain air pressure !", -1.0);
            break;
        case 2:
            SetCtrlVal(sigma1_panel, SIGMA1_RUN, 0);
            SetCtrlAttribute(sigma1_panel, SIGMA1_MODE, 15, 1);
            IssueWarning("Limit has been reached.", -1.0);
            break;
        case 3:
            SetCtrlVal(sigma1_panel, SIGMA1_RUN, 0);
            SetCtrlAttribute(sigma1_panel, SIGMA1_MODE, 15, 1);
            IssueWarning("Piston 1 is stuck !", -1.0);
            break;
        case 4:
            SetCtrlVal(sigma1_panel, SIGMA1_RUN, 0);
            SetCtrlAttribute(sigma1_panel, SIGMA1_MODE, 15, 1);
            IssueWarning("Piston 2 is stuck !", -1.0);
            break;
        case 5:
            SetCtrlVal(sigma1_panel, SIGMA1_RUN, 0);

```



```

        SetCtrlAttribute(sigma1_panel, SIGMA1_MODE, 15, 1);
        IssueWarning("Piston 3 is stuck !", -1.0);
    break;
    case 6:
        SetCtrlVal(sigma1_panel, SIGMA1_RUN, 0);
        SetCtrlAttribute(sigma1_panel, SIGMA1_MODE, 15, 1);
        IssueWarning("Piston 4 is stuck !", -1.0);
    break;
    case 7:
        IssueWarning("The system is not running !", 2.0);
    break;
}

global_error_flag = 0; /* clear the error flag */

return 0;
} /* function ProcessErrors ends here */

```

```

/*-----
| This function processes any event that occurs on sigma_3
| panel. It obtains the value of target pressure from user
| and sets or clears the flag controlling the execution of
| air pressure control routine.
|-----*/

```

```

int ProcessSigma3(int event_control_id)
{
    int run_flag;

    if(event_control_id == SIGMA3_RUN) { /* check the cause of the event */
        GetCtrlVal(sigma3_panel, SIGMA3_RUN, &run_flag); /* get the status of run_flag */
        /* if starting the control task then get the target pressure value */
        if(run_flag == 1) {
            /* if system is not running then issue warning and quit */
            if(GetSystemStatus() == 0) {
                SetCtrlVal(sigma3_panel, SIGMA3_RUN, 0);
                global_error_flag = 7; /* raise the error flag */
                return 1;
            }
            GetCtrlVal(sigma3_panel, SIGMA3_PTARGET, &target_airpressure);
            if((target_airpressure<0) ||(target_airpressure>50.0)) {
                SetCtrlVal(sigma3_panel, SIGMA3_RUN, 0);
                IssueWarning("Invalid target pressure value", 2.0);
                return 1;
            }
        }
        else
            SetCtrlVal(sigma3_panel, SIGMA3_PTARGET, target_airpressure);

        pressure_ctrl_flag = 1;
    }
}

```

```

    SetUserFlag();
    loop[AIR_PR].counter = 0;
}

    if(run_flag == 0) {
        pressure_ctrl_flag = 0;
        ClearUserFlag();
        loop[AIR_PR].counter = 0;
    }
}

return 0;

} /* function ProcessSigma3 ends here */

```

```

/*-----
| This function processes any event that occurs on sigma_2
| panel. It rezeros side wall position targets and sets or
| clears the flag controlling the execution of side wall
| control routine.
|-----*/
int ProcessSigma2( int event_control_id)
{
    int run_flag;

    if(event_control_id == SIGMA2_RUN) {
        GetCtrlVal(sigma2_panel, SIGMA2_RUN, &run_flag); /* get the status of run_flag */
        /* if starting the control task then reset the sensor readings */
        if(run_flag == 1) {
            /* if system is not running then issue warning and quit */
            if(GetSystemStatus() == 0) {
                SetCtrlVal(sigma2_panel, SIGMA2_RUN, 0);
                global_error_flag = 7; /* raise the error flag */
                return 1;
            }
            /* otherwise set loop counters to zero and set execution flag */
            loop[TOP_WALL].counter = 0;
            loop[BOT_WALL].counter = 0;
            sidewall_ctrl_flag = 1;
            SetUserFlag();
        }
        /* if stopping the control task then stop the motors and clear the execution flag */
        if(run_flag == 0) {
            ServoReset(loop[TOP_WALL].servo_num);
            ServoReset(loop[BOT_WALL].servo_num);
            loop[TOP_WALL].counter = 0;
            loop[BOT_WALL].counter = 0;
            sidewall_ctrl_flag = 0;
        }
    }
}

```

```

        ClearUserFlag();
    }
}

return 0;

} /* function ProcessSigma2 ends here */

```

```

/*-----
| This function runs in the foreground and turns main platforms
| on or off depending on the status of run switch on the sigma 1
| panel.
|-----*/

```

```

int ProcessSigma1(int event_control_id)
{
    int run_flag;

    /* check the cause of event & get the status of run_flag */
    if(event_control_id == SIGMA1_RUN) {
        GetCtrlVal(sigma1_panel, SIGMA1_RUN, &run_flag);
        if(run_flag == 1) {
            /* if system is not running then issue warning and quit */
            if(GetSystemStatus() == 0) {
                SetCtrlVal(sigma1_panel, SIGMA1_RUN, 0);
                global_error_flag = 7; /* raise the error flag */
                return 1;
            }
            /* obtain test parameters */
            GetCtrlVal(sigma1_panel, SIGMA1_STRTARGET, &target_strainrate);
            if((target_strainrate > 2.3) || (target_strainrate < -2.3)) {
                SetCtrlVal(sigma1_panel, SIGMA1_RUN, 0);
                IssueWarning("Invalid value for strain rate.", 2.0);
                return 1;
            }
        }
        else
            SetCtrlVal(sigma1_panel, SIGMA1_STRTARGET, target_strainrate);

            GetCtrlVal(sigma1_panel, SIGMA1_DISPLIMIT, &max_displacement);
            if((max_displacement > 20.0) || (max_displacement < -20.0)) {
                SetCtrlVal(sigma1_panel, SIGMA1_RUN, 0);
                IssueWarning("Invalid value for displacement limit.", 2.0);
                return 1;
            }
        else
            SetCtrlVal(sigma1_panel, SIGMA1_DISPLIMIT, max_displacement);

            GetCtrlVal(sigma1_panel, SIGMA1_STRLIMIT, &max_stress);

```

```

    if((max_stress > 350.0) || (max_stress < -25.0)) {
        SetCtrlVal(sigma1_panel, SIGMA1_RUN, 0);
        IssueWarning("Invalid value for stress limit.", 2.0);
        return 1;
    }
    else
        SetCtrlVal(sigma1_panel, SIGMA1_STRLIMIT, max_stress);

        loop[PIST_1].counter = 0;
        loop[PIST_2].counter = 0;
        loop[PIST_3].counter = 0;
        loop[PIST_4].counter = 0;
        strain_ctrl_flag = 1;
    SetUserFlag();
        SetCtrlAttribute(sigma1_panel, SIGMA1_MODE, 15, 0);
}
/* if stopping the control task then kill the motors */
if(run_flag == 0) {
    ServoReset(loop[PIST_1].servo_num);
    ServoReset(loop[PIST_2].servo_num);
    ServoReset(loop[PIST_3].servo_num);
    ServoReset(loop[PIST_4].servo_num);
    loop[PIST_1].counter = 0;
    loop[PIST_2].counter = 0;
    loop[PIST_3].counter = 0;
    loop[PIST_4].counter = 0;
    actual_strainrate = 0;
    strain_ctrl_flag = 0;
    ClearUserFlag();
    SetCtrlAttribute(sigma1_panel, SIGMA1_MODE, 15, 1);
}
}
return 0;
} /* function ProcessSigma1 ends here */

```

```

/*-----
| This function runs in the foreground and turns reinforcement |
| control function on or off.                                  |
|-----*/
int ProcessReinf(int event_control_id)
{
    int run_flag;

    /* check the cause of event & get the status of run_flag */
    if(event_control_id == REINF_RUN) {

```

```

GetCtrlVal(reinf_panel, REINF_RUN, &run_flag);
/* if starting control task then reset the target */
if(run_flag == 1) {
    if(GetSystemStatus() == 0) {
        SetCtrlVal(reinf_panel, REINF_RUN, 0);
        global_error_flag = 7; /* raise the error flag */
        return 1;
    }
    loop[REINFORCE].counter = 0;
    reinf_ctrl_flag = 1;
    SetUserFlag();
}
/* if stopping the control task then kill the motor */
if(run_flag == 0) {
    ServoReset(loop[REINFORCE].servo_num);
    loop[REINFORCE].counter = 0;
    reinf_ctrl_flag = 0;
    ClearUserFlag();
}
}
return 0;
} /* function ProcessReinforment ends here */

/*-----
| This function runs in the foreground and plot the selected
| variables on a graph selected from the database. It then
| displays the graph.
|-----*/
int ProcessGraph(int event_control_id)
{
    int graph_type;

    if(event_control_id == MISC_GRAPH) {
        GetCtrlVal(misc_panel, MISC_GRAPH, &graph_type);

        switch (graph_type) { /* plot the selected graph */
        case 0:
            GraphPlot(0, 0, index, time_array, epsilon1);
            break;
        case 1:
            GraphPlot(1, 0, index, time_array, R_value);
            break;
        case 2:
            GraphPlot(2, 0, index, time_array, q);
            break;
        case 3:
            GraphPlot(3, 0, index, epsilon1, R_value);
            break;

```

```

    case 4:
        GraphPlot(4, 0, index, epsilon1, q);
        break;
    case 5:
        GraphPlot(5, 0, index, epsilon1, epsilon3L);
        GraphPlot(5, 1, index, epsilon1, epsilon3C);
        break;
    case 6:
        GraphPlot(6, 0, index, epsilon1, epsilon_vol);
        break;
    case 7:
        GraphPlot(7, 0, index, epsilon1, b_value);
        break;
    case 8:
        GraphPlot(8, 0, index, epsilon1, epsilon2);
        break;
    case 9:
        GraphPlot(9, 0, index, R_value, load_cell);
        break;
    case 10:
        GraphPlot(10, 0, index, R_value, ir_sensor);
        break;
} /* switch statement ends here */

GraphDisplay(); /* display the current graph */

} /* if statement ends here */

return 0;

} /* function ProcessGraph ends here */

/*-----
| This function is called by the "system" in background at an |
| interval set by system.cycle_time. All control tasks are |
| performed through this function. No LabWindows user interface |
| function should ever be called from background functions or |
| else the system will surely hang up or crash !!! |
|-----*/
int UserBackground(void)
{
    if(pressure_ctrl_flag == 1) /* perform control tasks */
        ControlAirpressure();
    if(strain_ctrl_flag == 1)
        ControlPlatforms();
    if(sidewall_ctrl_flag == 1)
        ControlSidewalls();
    if(reinf_ctrl_flag == 1)

```

```

ControlReinforcement();

return 0;
} /* function UserBackground ends here */

/*-----
| This function runs in the background and controls the air
| pressure. It returns an error code which must be interpreted
| through a global variable.
|-----*/

int ControlAirpressure(void)
{
    int i;
    float actual_airpressure;
    static float error[3], vout = 0.0;

    if((loop[AIR_PR].counter == 0) && (vout == 0.0)) { /* starting vary first time */
        for(i=0;i<3;i++)
            error[i] = 0.0;
        vout = 1.0 + target_airpressure/15.0; /* use Fairchild formula */
        ServoWrite(loop[AIR_PR].servo_num, vout);
        loop[AIR_PR].counter++;
        return 1;
    }
    actual_airpressure = SensorGetAbsReading(loop[AIR_PR].sensor_num);
    error[0] = target_airpressure - actual_airpressure;
    vout = vout + loop[AIR_PR].kp * (error[0] - error[1]) \
        + loop[AIR_PR].ki * error[0] \
        + loop[AIR_PR].kv * (error[0] - 2.0 * error[1] + error[2]);

    if(target_airpressure == 0.0) /* move into fast lane ! */
        vout = 0.0;

    if(vout > 9.0) {
        ServoWrite(loop[AIR_PR].servo_num, 3.0); /* apply nominal pressure so that sample does not
collapse */
        loop[AIR_PR].counter = 0;
        pressure_ctrl_flag = 0; /* quit the pressure control mode */
        ClearUserFlag();
        global_error_flag = 1; /* raise the global error flag */
    }
    else
        ServoWrite(loop[AIR_PR].servo_num, vout);

    error[2] = error[1];
    error[1] = error[0];
    loop[AIR_PR].counter++; /* increment the loop counter */
}

```

```

return 0;

} /* function ControlAirpressure ends here */

/*-----
| This function runs in the background and controls side wall |
| in order to maintain plane strain condition.                |
|-----*/
int ControlSidewalls(void)
{
    float difference, vout;

    if(loop[TOP_WALL].counter == 0) { /* if entered for the first time */
        SensorTakeZero(loop[TOP_WALL].sensor_num);
        SensorTakeZero(loop[BOT_WALL].sensor_num);
        ServoReset(loop[TOP_WALL].servo_num);
        ServoReset(loop[BOT_WALL].servo_num);
        loop[TOP_WALL].counter++; /* increment the loop counters */
        loop[BOT_WALL].counter++;
        return 1;
    }

    difference = SensorGetReading(loop[TOP_WALL].sensor_num);
    vout = loop[TOP_WALL].kp*difference;
    ServoWrite(loop[TOP_WALL].servo_num, vout);
    difference = SensorGetReading(loop[BOT_WALL].sensor_num);
    vout = loop[BOT_WALL].kp*difference;
    ServoWrite(loop[BOT_WALL].servo_num, vout);

    loop[TOP_WALL].counter++; /* increment the loop counters */
    loop[BOT_WALL].counter++;

    return 0;
} /* function ControlSidewalls ends here */

```

```

/*-----
| This function runs in the background and controls four pistons |
| to drive main platforms at constant rate of strain. It also   |
| checks for the limits and raises a global error flag when     |
| are reached.                                                  |
|-----*/
int ControlPlatforms(void)
{

```



```

static int limit_flag1, limit_flag2;
static float elapsed_time;
float sigma11, sigma12;
float disp1, disp2, disp3, disp4, disp12, disp34, target_disp, average_disp;
float velocity1, velocity2, velocity3, velocity4;
static float init_time, previous_disp1, previous_disp2, previous_disp3, previous_disp4;
static float vout1, vout2, vout3, vout4;

if(loop[PIST_1].counter == 0) {
    SensorTakeZero(loop[PIST_1].sensor_num);    /* rezero the sensors */
    SensorTakeZero(loop[PIST_2].sensor_num);
    SensorTakeZero(loop[PIST_3].sensor_num);
    SensorTakeZero(loop[PIST_4].sensor_num);
    /* compute output volts using servo calibration */
    vout1 = vout2 = vout3 = vout4 = 2.0 * target_strainrate;
    ServoWrite(loop[PIST_1].servo_num, vout1);    /* send voltages */
    ServoWrite(loop[PIST_2].servo_num, vout2);
    ServoWrite(loop[PIST_3].servo_num, vout3);
    ServoWrite(loop[PIST_4].servo_num, vout4);

    previous_disp1 = previous_disp2 = previous_disp3 = previous_disp4 = 0;

    loop[PIST_1].counter++;    /* increment the counters */
    loop[PIST_2].counter++;
    loop[PIST_3].counter++;
    loop[PIST_4].counter++;

    limit_flag1 = 0;    /* clear the limit flags */
    limit_flag2 = 0;

    init_time = GetSystemTime();    /* read the initial time */
    elapsed_time = 0.0;

    return 0;
}

elapsed_time = elapsed_time + system.cycle_time/60.0;    /* compute elapsed time in minute */
target_disp = target_strainrate*elapsed_time;

disp1 = SensorGetReading(loop[PIST_1].sensor_num);    /* take readings */
disp2 = SensorGetReading(loop[PIST_2].sensor_num);
disp3 = SensorGetReading(loop[PIST_3].sensor_num);
disp4 = SensorGetReading(loop[PIST_4].sensor_num);

if(limit_flag1 == 0) {    /* do for platform 1 */
    velocity1 = 60.0*(disp1 - previous_disp1)/system.cycle_time;
    velocity2 = 60.0*(disp2 - previous_disp2)/system.cycle_time;

    vout1 = vout1 + loop[PIST_1].kp*(target_disp - disp1) + \
        loop[PIST_1].kv*(target_strainrate - velocity1);
}

```

```

vout2 = vout2 + loop[PIST_2].kp*(target_disp - disp2) + \
    loop[PIST_2].kv*(target_strainrate - velocity2);

ServoWrite(loop[PIST_1].servo_num, vout1);
ServoWrite(loop[PIST_2].servo_num, vout2);

previous_disp1 = disp1;    /* reassign displacement values */
previous_disp2 = disp2;

loop[PIST_1].counter++;    /* increment the counters */
loop[PIST_2].counter++;

/* perform limit checks */
disp12 = 0.5*(SensorGetAbsReading(loop[PIST_1].sensor_num) +
SensorGetAbsReading(loop[PIST_2].sensor_num));
sigma11 = (SensorGetAbsReading(5) + SensorGetAbsReading(6))*0.19 - 5.5;
if(target_strainrate > 0) {
    if(disp12 >= max_displacement)
        limit_flag1 = 1;
    if(sigma11 >= max_stress)
        limit_flag1 = 1;
}
if(target_strainrate < 0) {
    if(disp12 <= max_displacement)
        limit_flag1 = 1;
    if(sigma11 <= max_stress)
        limit_flag1 = 1;
}

if(limit_flag1 == 1) {    /* stop motors if limit is reached */
    ServoReset(loop[PIST_1].servo_num);
    ServoReset(loop[PIST_2].servo_num);
}

}

if(limit_flag2 == 0) {    /* do the same for platform 2 */
    velocity3 = 60.0*(disp3 - previous_disp3)/system.cycle_time;
    velocity4 = 60.0*(disp4 - previous_disp4)/system.cycle_time;

    vout3 = vout3 + loop[PIST_3].kp*(target_disp - disp3) + \
        loop[PIST_3].kv*(target_strainrate - velocity3);
    vout4 = vout4 + loop[PIST_4].kp*(target_disp - disp4) + \
        loop[PIST_4].kv*(target_strainrate - velocity4);

    ServoWrite(loop[PIST_3].servo_num, vout3);
    ServoWrite(loop[PIST_4].servo_num, vout4);

    previous_disp3 = disp3;
    previous_disp4 = disp4;
}

```

```

loop[PIST_3].counter++;
loop[PIST_4].counter++;

/* perform limit checks */
disp34 = 0.5*(SensorGetAbsReading(loop[PIST_3].sensor_num) +
SensorGetAbsReading(loop[PIST_4].sensor_num));
sigma12 = (SensorGetAbsReading(7) + SensorGetAbsReading(8))*0.19 - 5.5;
if(target_strainrate > 0) {
    if(disp34 >= max_displacement)
        limit_flag2 = 1;
    if(sigma12 >= max_stress)
        limit_flag2 = 1;
}
if(target_strainrate < 0) {
    if(disp34 <= max_displacement)
        limit_flag2 = 1;
    if(sigma12 <= max_stress)
        limit_flag2 = 1;
}

if(limit_flag2 == 1) { /* stop motors if limit is reached */
    ServoReset(loop[PIST_3].servo_num);
    ServoReset(loop[PIST_4].servo_num);
}

}

if(limit_flag1 == 1) /* compute actual strain rate */
    actual_strainrate = 0.5*(disp3 + disp4)/elapsed_time;
else if(limit_flag2 == 1)
    actual_strainrate = 0.5*(disp1 + disp2)/elapsed_time;
else
    actual_strainrate = 0.25*(disp1 + disp2 + disp3 + disp4)/elapsed_time;

average_disp = 0.5*(disp1 + disp2); /* check if any piston is stuck */

if((average_disp - disp1) > 0.125) {
    ExitStrainLoop();
    global_error_flag = 3;
}
if((average_disp - disp2) > 0.125) {
    ExitStrainLoop();
    global_error_flag = 4;
}

average_disp = 0.5*(disp3 + disp4);

if((average_disp - disp3) > 0.125) {
    ExitStrainLoop();
    global_error_flag = 5;
}
}

```

```

if((average_disp - disp4) > 0.125) {
    ExitStrainLoop();
    global_error_flag = 6;
}

if((limit_flag1==1) && (limit_flag2==1)) { /* if limits have been reached */
    ExitStrainLoop();
    global_error_flag = 2; /* raise the global error flag */
}

return 0;

} /* function ControlPlatforms ends here */

```

```

/*-----
| This function performs all house cleaning operations associated |
| with the termination of platform control function and clears |
| strain_ctrl_flag so as to exit the control loop. |
|-----*/

```

```

int ExitStrainLoop(void)
{
    ServoReset(loop[PIST_1].servo_num); /* stop all motors */
    ServoReset(loop[PIST_2].servo_num);
    ServoReset(loop[PIST_3].servo_num);
    ServoReset(loop[PIST_4].servo_num);
    loop[PIST_1].counter = 0; /* reset loop counters */
    loop[PIST_2].counter = 0;
    loop[PIST_3].counter = 0;
    loop[PIST_4].counter = 0;
    actual_strainrate = 0;
    ClearUserFlag(); /* clear the user flag */
    strain_ctrl_flag = 0; /* quit the control function */

    return 0;

} /* function ExitStrainLoop ends here */

```

```

/*-----
| This function runs in the background and controls the |
| position of reinforcement. |
|-----*/

```

```

int ControlReinforcement(void)
{

```

```

float difference, vout;

if(loop[REINFORCE].counter == 0) { /* if entered for the first time */
  SensorTakeZero(loop[REINFORCE].sensor_num);
  ServoReset(loop[REINFORCE].servo_num);
  loop[REINFORCE].counter++;
  return 1;
}

difference = SensorGetReading(loop[REINFORCE].sensor_num);
vout = loop[REINFORCE].kp*difference;
ServoWrite(loop[REINFORCE].servo_num, vout);

loop[REINFORCE].counter++; /* increment the loop counter */

return 0;
} /* function ControlReinforcement ends here */

```

```

/*-----
| This function must be defined in a user program. It computes
| test parameters from sensor readings. These parameters are
| used for graph display and result file. This function is
| called in the background by the data acquisition module and
| it should not contain any user interface library routines.
|-----*/

```

```

int UserCompute(void)
{
  float disp11, disp12;
  float pp1, pp2, pp3, pp4, sigma1;
  float sigma2_t, sigma2_b, disp2_t, disp2_b;

  if(index == 0) { /* if entered for the first time */
    result_file_flag = 1; /* set the result file flag */
    epsilon1[0] = 0.0; /* initialize selected parameters */
    epsilon2[0] = 0.0;
    epsilon3L[0] = epsilon3C[0] = 0.0;
  }

  disp11 = 0.5*(SensorGetAbsReading(1) + SensorGetAbsReading(2));
  disp12 = 0.5*(SensorGetAbsReading(3) + SensorGetAbsReading(4));
  epsilon1[index] = (100.0*(disp11 + disp12)/sample_length) - epsilon1[0];
  pp1 = SensorGetAbsReading(5);
  pp2 = SensorGetAbsReading(6);
  pp3 = SensorGetAbsReading(7);
  pp4 = SensorGetAbsReading(8);
  sigma11[index] = (pp1 + pp2)*0.19 - 5.5;
  sigma12[index] = (pp3 + pp4)*0.19 - 5.5;
  sigma1 = 0.5*(sigma11[index] + sigma12[index]);

```

```

disp2_t = SensorGetReading(9);
disp2_b = SensorGetReading(10);
epsilon2[index] = (100.0*(disp2_t + disp2_b)/sample_height) - epsilon2[0];

sigma2_t = SensorGetAbsReading(11);
sigma2_b = SensorGetAbsReading(12);
sigma2[index] = 0.5*(sigma2_t + sigma2_b);

epsilon3C[index] = (100.0*SensorGetReading(13)/sample_width) - epsilon3C[0];
epsilon3L[index] = (100.0*SensorGetReading(14)/sample_width) - epsilon3L[0];

sigma3[index] = SensorGetAbsReading(0);

epsilon_vol[index] = epsilon1[index] + 0.5*(epsilon3L[index] + epsilon3C[index]);
q[index] = 0.5*(sigma1 - sigma3[index]);
if(sigma3[index] != 0.0)
    R_value[index] = sigma1/(sigma3[index]);
if(q[index] != 0.0)
    b_value[index] = (sigma2[index]-sigma3[index])/(sigma1 - sigma3[index]);

load_cell[index] = SensorGetAbsReading(17);
ir_sensor[index] = SensorGetReading(16);

time_array[index] = calander.elapsed_time/60.0;    /* increment index */

index++;    /* increment index */

return 0;

} /* end function UserCompute */

```

```

/*-----
| This is the last user function to be executed. It is
| executed only once at the end of the program. It unloads
| all APSR panels.
|-----*/

```

```

int MeLast(void)
{
    int ctrl_event = -1, exit_flag = 0;
    int pan_handle;
    char file_name[15];

    if(result_file_flag == 1) {
        pan_handle = LoadPanel("APSR.UIR", RESULT);
        InstallPopup(pan_handle);
        while (!exit_flag) {
            GetCtrlVal(pan_handle, RESULT_NAME, file_name);

```

```

GetPopupEvent(0,&ctrl_event);
if(ctrl_event == RESULT_OK) {
    SaveResults(file_name);
    RemovePopup(0);
    UnloadPanel(pan_handle);
    exit_flag = 1;
}
if (ctrl_event == RESULT_CAN) {
    RemovePopup(0);
    UnloadPanel(pan_handle);
    exit_flag = 1;
}
}
}

UnloadPanel(sigma1_panel);    /* unload other APSR panels */
UnloadPanel(sigma2_panel);
UnloadPanel(sigma3_panel);
UnloadPanel(reinf_panel);
UnloadPanel(misc_panel);
UnloadPanel(main_panel);

return 0;
} /* end function MeLast */

```

```

/*-----
| This function opens a file in the default directory and
| writes values of all test parameters. This file can later
| be inmorted into any spread sheet or graphing program.
|-----*/

```

```

int SaveResults(char *file_name)
{
    int i=0;
    FILE *file_handle;
    char temp[15];
    char *title[11] = { " Time ", \
        " Epsilon_1 ", " R_Value ", " q ", \
        " b_value ", " Epsilon_2 ", " Epsilon_3L ", \
        " Epsilon_3C ", " Epsilon_vol ", " Load_cell ", \
        " IR_Sensor " };

    /* open a new file */
    strcpy(temp, file_name);    /* attach .res extension */
    strcat(temp, ".res");
    file_handle = fopen(temp, "wt");
    if(file_handle == NULL) {
        IssueWarning("Error opening result file on disk.", 2.0);
        return 1;
    }
}

```

```

}

fp.printf(file_handle, "%s\n", file_name);      /* print file header */
fprintf(file_handle, "%s\n", calander.date_created);
fprintf(file_handle, "%s\n", calander.time_created);

fprintf(file_handle, "%12s ", title[0]);      /* print column titles */
for(i=1; i<10;i++)
    fprintf(file_handle, "%15s ", title[i]);
fprintf(file_handle, "%15s\n", title[10]);

for(i=0; i<index; i++) {
    fprintf(file_handle, "%12.3f %+15.6f %+15.6f %+15.6f %+15.6f %+15.6f ", time_array[i],
epsilon1[i], R_value[i], q[i], b_value[i], epsilon2[i]);
    fprintf(file_handle, "%+15.6f %+15.6f %+15.6f %+15.6f %+15.6f\n", epsilon3L[i],
epsilon3C[i], epsilon_vol[i], load_cell[i], ir_sensor[i]);
}

fflush(file_handle);
fclose(file_handle);

return 0;

} /* function SaveResults ends here */

/* end of APSR.C module */

```