

JAMES ERNEST RICKMAN
Captain, USAF

**Three-Dimensional Visualization
of Ozone Process Data**

1997

153 pages

**Master of Science,
Atmospheric Science**

North Carolina State University

EXEMPTION STATEMENT A

Approved for public release;
Distribution Unlimited

19970625 056

DTIC QUALITY INSPECTED 8

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188	
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.				
1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE 18 JUN 97	3. REPORT TYPE AND DATES COVERED		
4. TITLE AND SUBTITLE THREE-DIMENSIONAL VISUALIZATION OF OZONE PROCESS DATA			5. FUNDING NUMBERS	
6. AUTHOR(S) JAMES ERNEST RICKMAN				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) NORTH CAROLINA STATE UNIVERSITY			8. PERFORMING ORGANIZATION REPORT NUMBER 97-064	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) DEPARTMENT OF THE AIR FORCE AFIT/CI BLDG 125 2950 P STREET WRIGHT-PATTERSON AFB OH 45433-7765			10. SPONSORING/MONITORING AGENCY REPORT NUMBER	
11. SUPPLEMENTARY NOTES				
12a. DISTRIBUTION AVAILABILITY STATEMENT			12b. DISTRIBUTION CODE	
13. ABSTRACT (Maximum 200 words)				
14. SUBJECT TERMS			15. NUMBER OF PAGES 153	
			16. PRICE CODE	
17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT	

ABSTRACT

RICKMAN, JAMES ERNEST, Three-Dimensional Visualization of Ozone Process Data. (Under the direction of Viney Pal Aneja.)

In the last ten years, significant developments have occurred in the separate fields of three-dimensional visualization and air quality. The primary goal of this research is to find means and methods of coupling air quality data to three-dimensional visualization in order to enhance ozone process analysis. Process analysis data refers to the individual rates associated with the different processes (advection, diffusion, deposition, and chemical reaction) within the mass balance and integrated reaction rate calculations of air quality models. Generally discarded by models after computations are completed, a process analysis method retains these rates for analysis into which factors are most significantly impacting the accumulation of ozone in the troposphere.

The author, working with researchers from MCNC - North Carolina Supercomputing Center, adapted techniques for visualizing ozone data in three-dimensional volumes using the Application Visualization System and Vis-5D. They also made adaptations to Vis-5D to visually compare the contributions of the individual processes using small multiple techniques and cycle schematics.

Five case studies (two low diffusion events, a strong advection event, one stagnation event, and one event with low contribution by chemical processes) in the eastern United States, from the summer of 1995, suggest that these techniques, when used together, provide many insights (maximum versus minimum contribution, offsetting contributions, and meteorological versus chemical contributions) into the primary processes responsible for ozone accumulation events.

BIBLIOGRAPHY

- Advanced Visual Systems, Inc. (1993) *AVS User's Guide*. Advanced Visual Systems, Inc., Waltham, MA.
- Aneja, V. P. and Li, Z. (1992) Characterization of Ozone at High Elevation in the Eastern United States: Trends, Seasonal Variations, and Exposure. *Journal of Geophysical Research*. 97 (D9), 9873-9888.
- Boutell, T. (1995) *gd 1.2 Users Guide*. Quest Protein Database Center, Cold Spring Harbor Labs.
- Dennis, R.L., Byun, D.W., Novak, J.H., et al. (1996) The Next Generation of Integrated Air Quality Modeling: EPA's Models-3. *Atmospheric Environment*. 30 (12), 1925-1938.
- Fine, S.S. and Mathur, R. (1996) Applying Multivariate Analysis Techniques to Interpreting Results Produced by Chemistry-Transport Models. To appear in *Proceedings, Computing in Environmental Resource Management*. Air & Waste Management Association, Research Triangle Park.
- Finlayson-Pitts, B.J. and Pitts, J.N., Jr. (1986) *Atmospheric Chemistry: Fundamentals and Experimental Techniques*. John Wiley & Sons, Inc., New York.
- Finlayson-Pitts, B.J. and Pitts, J.N., Jr. (1993) Atmospheric Chemistry of Tropospheric Ozone Formation: Scientific and Regulatory Implications. *Journal of the Air and Waste Management Association*. 43, 1091-1100.
- Hibbard, B. and Santek, D. (1990) The VIS-5D System for Easy Interactive Visualization. *Proceedings of the First IEEE Conference on Visualization: Visualization '90*, pp 28-35. IEEE Computer Society, San Francisco.
- Hibbard, W. and Santek, D. (1989) Interactivity is the Key. *Chapel Hill Workshop on Volume Visualization: Conference Proceedings*, pp 39-43. Department of Computer Science, University of North Carolina at Chapel Hill.
- Hibbard, W.L., Paul, B.E., Santek, D.A., et al. (1994) Interactive Visualization of Earth and Space Science Computations. *Computer*. 27 (July 1994), 65-72.
- Jang, J.C., Jeffries, H.E., Byun, D., et al. (1995) Sensitivity of Ozone to Model Grid Resolution — I. Application of High-Resolution regional Acid Deposition Model. *Atmospheric Environment*. 29 (21), 3085-3100.
- Jang, J.C., Jeffries, H.E., and Tonnesen, S. (1995) Sensitivity of Ozone to Model Grid Resolution — II. Detailed Process Analysis for Ozone Chemistry. *Atmospheric Environment*. 29 (21), 3101-3114.
- Jeffries, H.E. and Tonnesen, S. (1994) A Comparison of Two Photochemical Reaction Mechanisms Using Mass Balance and Process Analysis. *Atmospheric Environment*. 28 (18), 2991-3003.
- Nielson, G.M., Foley, T.A., Hamann, B., et al. (1991) Visualizing and Modeling Scattered Multivariate Data. *IEEE Computer Graphics & Applications*. 11 (May), 47-55.

Odman, M.T. and Ingram, C.L. (1996) *Multiscale Air Quality Simulation Platform (MAQSIP): Source Code Documentation and Validation*, ENV-96TR002-v1.0. Environmental Programs, MCNC-North Carolina Supercomputing Center, Research Triangle Park.

Thorpe, S., Ambrosiano, J., Balat, R., et al. (1996) The Package for Analysis and Visualization of Environmental Data. *Proceedings: Thirty-Eighth Semi-Annual Cray User Group Meeting*, pp 46-50. Cray User Group, Inc., Charlotte, NC.

Tufte, E.R. (1990) *Envisioning Information*. Graphics Press, Cheshire, Connecticut.

Warneck, P. (1988) *Chemistry of the Natural Atmosphere*. Academic Press, San Diego.

ABSTRACT

RICKMAN, JAMES ERNEST, Three-Dimensional Visualization of Ozone Process Data. (Under the direction of Viney Pal Aneja.)

In the last ten years, significant developments have occurred in the separate fields of three-dimensional visualization and air quality. The primary goal of this research is to find means and methods of coupling air quality data to three-dimensional visualization in order to enhance ozone process analysis. Process analysis data refers to the individual rates associated with the different processes (advection, diffusion, deposition, and chemical reaction) within the mass balance and integrated reaction rate calculations of air quality models. Generally discarded by models after computations are completed, a process analysis method retains these rates for analysis into which factors are most significantly impacting the accumulation of ozone in the troposphere.

The author, working with researchers from MCNC - North Carolina Supercomputing Center, adapted techniques for visualizing ozone data in three-dimensional volumes using the Application Visualization System and Vis-5D. They also made adaptations to Vis-5D to visually compare the contributions of the individual processes using small multiple techniques and cycle schematics.

Five case studies (two low diffusion events, a strong advection event, one stagnation event, and one event with low contribution by chemical processes) in the eastern United States, from the summer of 1995, suggest that these techniques, when used together, provide many insights (maximum versus minimum contribution, offsetting contributions, and meteorological versus chemical contributions) into the primary processes responsible for ozone accumulation events.

**THREE-DIMENSIONAL VISUALIZATION
OF OZONE PROCESS DATA**

by

JAMES ERNEST RICKMAN

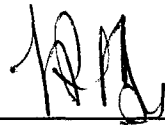
A thesis submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the Degree of
Master of Science

MARINE, EARTH AND ATMOSPHERIC SCIENCES

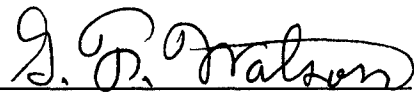
Raleigh

1997

APPROVED BY:



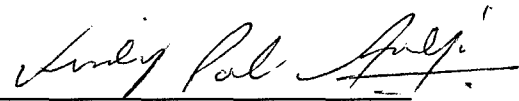
S. PAL ARYA



GERALD F. WATSON



STEVEN S. FINE



VINEY P. ANEJA
Chair of Advisory Committee

PERSONAL BIOGRAPHY

James E. Rickman, Captain, United States Air Force, [REDACTED]

[REDACTED]. He attended Centerville High School in Centerville, Ohio, where he was active in the marching and symphonic bands, the International Thespian Society, and the National Honor Society. He graduated from Centerville High School ranked number 18 in a class of 660, on June 15, 1985. He then attended the Florida State University in Tallahassee, Florida, on an Air Force Reserve Officer Training Corps, Four-Year Scholarship. He was very active in the 145th Cadet Group and served as the president of the Arnold Air Society. He graduated from the Florida State University with a Bachelor of Arts in Meteorology on April 29, 1989, and was commissioned a Second Lieutenant on May 1, 1989, receiving a Distinguished Graduate award.

In October 1989, Capt Rickman was assigned to Detachment 1, 4th Weather Wing at Falcon Air Force Base, Colorado, as an Atmospheric and Space Environmental Forecaster and Staff Weather Officer to the 2d Space Wing. After the 1991 wing reorganization, Captain Rickman was assigned to the 50th Operations Support Squadron, 50th Space Wing, Falcon AFB, CO. It was in this position in 1992 that he became the first weather officer selected as Company Grade Officer of the Quarter of a Space Squadron.

In November 1992, Captain Rickman was assigned to the Automated Weather Distribution System (AWDS) Assistance Team at the 7th Weather Squadron, Heidelberg Army Installation, Germany as an AWDS Transition and Exploitation Team Chief and Weather Programs Officer. The United States Air Forces in Europe (USAFE) recognized him for his outstanding technical support to the USAFE Weather Support Mission with the 1994 USAFE Merewether Award. After the weather reorganization of July 1994, he was assigned as the Future Requirements Officer for the 617th Weather Squadron, Heidelberg AIN, GE, where he was recognized as both a Company Grade Officer of the Quarter for the Squadron, and Company Grade Weather Officer of the Year for the 617th Air Support Operations Group for 1994.

In August 1995, Captain Rickman was selected for his current assignment, to attend the Air Force Institute of Technology's Civilian Institution Program at the North Carolina State University, Raleigh, North Carolina, to pursue a Master of Science in Meteorology, specializing in the use of Interactive Video Graphics.

Captain Rickman has earned the Air Force and the Army Commendation Medals, the Air Force Achievement Medal, and the National Defense Service Medal. He wears the Air Force Senior Meteorology Badge.

Captain Rickman is single. He is the son of Richard and Sarah Rickman of Centerville, Ohio.

ACKNOWLEDGEMENTS

I have to thank the entire team in the Environmental Programs Group at the MCNC - North Carolina Super Computing Center, for helping me and putting up with me for the last year. Ken Galluppi and Ed Bilicki, who got me started with the group when this process all began. Neil Wheeler who helped me focus my project and connect me with the right people to accomplish this work, not to mention oversaw my entrance into the study. Don Olerud and Rohit Mathur who helped me gather the data for some of the case studies and answer questions on the data itself. Steve Thorpe, who gave me my introduction to the visualization resources, and went on to answer a myriad of questions involving access to and use of the computer systems, data field formats, and programming in general. One of the biggest debts of gratitude goes out to Carey Jang who worked with me almost daily to accomplish this program, helping me understand the process analysis methods and what he needed to be able to continue his research. The other goes out to Steve Fine, who not only served on my committee, but also was the guiding force in keeping my work on track and my mind on the science.

Through all of these people, I want to acknowledge MCNC's contribution to this work, which was supported by EPA grant R825199-010 and NASA grant NAGW-4701.

I want to thank the NCSU Air Quality Group, Mita, Paul, Jim, Lisa, Regi, Li, John, JP, Lara, Daiwan, and Deug-Soo, for all of their support, both academically and personally. Especially to Jeff Brittig, my fellow Air Force guy, who went through all of those homework, mid-term, and final sessions with me, not to mention many evenings in a soybean field in eastern North Carolina.

Thanks to my NCSU committee members, Dr Pal Arya and Dr Jerry Watson. And especially my advisor, Dr Viney Aneja, who helped me to find a way to do the Interactive Video Graphics program exactly the way I had hoped I'd be able to do it.

Thanks to the United States Air Force for giving me the chance to work on this thesis and giving me a career that I love and look forward to continuing.

Finally, thanks to my family. Both my relations and my extended family at First Assembly, who loved me and supported me through these 22 months of school. And, of course, my loving Heavenly Father, and my Savior, Jesus Christ, who have made all the difference in my life.

TABLE OF CONTENTS

	Page
LIST OF FIGURES	viii
1. INTRODUCTION	1
1.1 Process Analysis	1
1.2 Three-Dimensional Visualization	2
2. BACKGROUND	3
2.1 Ozone	3
2.2 Visualization	7
2.2.1 Volumetric Data	8
2.2.2 The Use of Color	10
3. DEVELOPMENT	12
3.1 Selection of Visualization Technology	13
3.2 Designing New Techniques	23
4. CASE STUDIES	30
4.1 Houston, Texas, July 8, 1995 (54 km Resolution)	34
4.2 Baton Rouge, Louisiana, July 9, 1995 (54 km Resolution)	40
4.3 Atlanta, Georgia, July 10 - 11, 1995 (54 km Resolution)	47
4.4 St Louis, Missouri, July 12, 1995 (36 km Resolution)	54
4.5 Washington, DC, July 12, 1995 (36 km Resolution)	60
4.6 Comparison of Cases Using Small Multiple Plotting Map	66
5. CONCLUSIONS	69
6. LIST OF REFERENCES	72

7. APPENDICIES.....	74
7.1 Setup Vis-5D.....	75
7.1.1 gdfontsy.c.....	83
7.1.2 gdfontsy.h.....	108
7.1.3 rickman.c.....	109
7.1.4 rickmang.c.....	123
7.1.5 rickman.h.....	135
7.1.6 rickmang.h.....	136
7.1.7 patemplate.c.....	138
7.2 Data Conversion.....	145
7.3 User's Guide.....	149

LIST OF FIGURES

		Page
FIGURE 2.1	Hydroxyl Radical Cycle and Nitric Oxide Oxidation Cycle schematic	6
FIGURE 2.2	A cuberile grid where P=(2,4,5) is accented	9
FIGURE 3.1	Black and white version of the variation of the Tiny Cubes method	14
FIGURE 3.2	Value scale coloring of ozone concentration on a planar slice	17
FIGURE 3.3	0.05 ppmv interval contours of ozone concentration on a planar slice displaying the same data set as in Figure 3.2.....	18
FIGURE 3.4	Small multiple of process rates	25
FIGURE 3.5	Diamond plot small multiple representing the same information as in Figure 3.4	26
FIGURE 3.6	Computer generated cycle schematic	29
FIGURE 4.1	Map of case study region.....	31
FIGURE 4.2	Concentrations are calculated at the center of the hatched cell. u and v wind components are calculated at the centers of the four empty cells, at the base of the vectors.....	32
FIGURE 4.3	Map of cell reference grid over Houston, TX (38,10)	35
FIGURE 4.4	Small multiples for Houston, TX case	36
FIGURE 4.5	Planar slice of chemical process contours for the Houston, TX case.....	37
FIGURE 4.6	Planar slice of diffusion and deposition process contours for the Houston, TX case.....	39

FIGURE 4.7	Map of cell reference grid over Baton Rouge, LA (38,17)	41
FIGURE 4.8	Small multiples for Baton Rouge, LA case.....	42
FIGURE 4.9	Planar slice of chemical process contours for the Baton Rouge, LA case	44
FIGURE 4.10	Isosurface rendering of -20 ppbv/h diffusion and 0.12 ppmv ozone concentration for the Baton Rouge, LA case	46
FIGURE 4.11	Map of cell reference grid over Atlanta, GA (30,28)	48
FIGURE 4.12	Small multiples for 10/12 - 11/11, Atlanta, GA case	49
FIGURE 4.13	Small multiples for 11/12 - 12/10, Atlanta, GA case	50
FIGURE 4.14	Planar slice of chemical process contours for the Atlanta, GA case on July 10th.....	52
FIGURE 4.15	Planar slice of chemical process contours for the Atlanta, GA case on July 11th.....	53
FIGURE 4.16	Map of cell reference grid over St Louis, MO (30,27)	55
FIGURE 4.17	Small multiples for St Louis, MO case	56
FIGURE 4.18	Planar slice of chemical process contours with wind vectors for the St Louis, MO case	57
FIGURE 4.19	Isosurface rendering of 0.12 ppmv ozone concentration and total advection process contours for the St Louis, MO case	59
FIGURE 4.20	Map of cell reference grid over Washington, DC (27,58)	61
FIGURE 4.21	Small multiples for Washington, DC case	62
FIGURE 4.22	Planar slice of chemical process contours for the Washington, DC case	63

FIGURE 4.23	Volume imaging of chemical process rates for Washington, DC case	65
FIGURE 4.24	Small multiple plotting map comparing similar shapes near Houston, TX, and Baton Rouge, LA.....	66
FIGURE 4.25	Small multiple plotting map comparing similar shapes near Houston, TX, and Baton Rouge, LA.....	67
FIGURE 4.26	Small multiple plotting map comparing dissimilar shapes near St Louis, MO, and Washington, DC	68

1 INTRODUCTION

Ozone concentrations continue to exceed the National Ambient Air Quality Standard, 0.12 parts per million by volume (ppmv), averaged over one hour, in some areas of the U S. ¹ High ozone levels have become a major issue among atmospheric scientists and chemists in the field of air quality. This issue has become a public interest issue following the documentation and establishment of the impacts on vegetation and human respiratory functions. ^{2,3} Since photochemical processes are solely responsible for its formation in the atmosphere, ozone is a secondary pollutant. Those photochemical processes, as well as those of advection, diffusion, and deposition are predominantly responsible for the distribution of ozone in the atmosphere.

1.1 Process Analysis

Since many air quality studies today rely on atmospheric and chemical modeling using computers, some scientists have a tendency to treat models as black boxes into which numbers go, and different numbers come out. The whole process relies on automated algorithms and computations which are entirely invisible. Sometimes, especially in air quality modeling, these computations become so complex that we lose sight of the individual processes within the numerical simulation, and we do not fully understand

the relationship between the input and output of these systems. ⁴ Most air quality models generally output the calculated concentration values for each species they analyze and discard the rates of change for each species. They calculate these rates and concentrations for unit volumes throughout the simulated atmosphere. Each unit volume is located at a grid point, and is referred to as a cell. There is much to learn about model behavior by looking at those individual rates. A "process analysis" method saves the contributions of the individual chemical and physical processes as they are calculated instead of discarding them, so that analysts may use them in post processing to identify the contributions of the individual processes. ⁵ Similarly, a greater understanding of the contributions of advection, diffusion, and deposition to the concentration can give the analyst greater insight into the sources and sinks of ozone in a cell. Furthermore, studying the individual contributions to increases and decreases in ozone concentration can give analysts a better understanding of which mechanisms or processes combine to generate high ozone concentrations, and which offset each other to keep ozone concentrations low. The goal of this work is to study methods and tools for analyzing process data.

1.2 Three-Dimensional Visualization

This research takes the analyst into the realm of three-dimensional

visualization of data to provide insights into the process analysis data. In the last five to ten years, significant advancements in the field of three-dimensional data visualization have occurred. Research in the field of process analysis has developed in the same time frame. The author believes that these two relatively new fields of study can complement each other well, and that belief is the basis for this research.

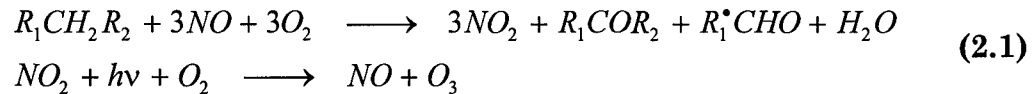
To see the results of the complementary use of these two fields, the author accomplished several case studies which involved retrospective computer simulations of high ozone events. The author used data from the **Multiscale Air Quality Simulation Platform (MAQSIP)** modeling system. MAQSIP is a modular comprehensive air quality modeling system which MCNC - North Carolina Supercomputing Center's Environmental Programs group is developing as a prototype of the U S Environmental Protection Agency's third-generation air quality modeling system, Models-3.^{6,7,8}

2 BACKGROUND

2.1 Ozone

In an unpolluted (remote) environment, the troposphere has a background ozone concentration which originates from two sources, the large reservoir in the "Ozone Layer" in the stratosphere, and from the chain of reactions initiated by the photodissociation of nitrogen dioxide in the

troposphere. Owing to enhanced emissions of ozone precursors (oxides of nitrogen and volatile organic compounds) in urban and some rural locations, ozone concentrations build-up based on a complex set of photochemical reactions. A general net mechanism for ozone production in the troposphere from the oxidation of a straight-chain alkane (for example) is as follows:^{9,10}



It is these processes which air quality modelers strive to incorporate.

On a basic level, air quality models use mass continuity equations to calculate the concentrations of the myriad of chemical species in the atmospheric reactions. These mass continuity equations generally take the following form:

$$\begin{aligned} \frac{\partial C_i}{\partial t} + \frac{\partial}{\partial x}(uC_i) + \frac{\partial}{\partial y}(vC_i) + \frac{\partial}{\partial z}(wC_i) + \\ \frac{\partial}{\partial x}\left(-K_x \frac{\partial C_i}{\partial x}\right) + \frac{\partial}{\partial y}\left(-K_y \frac{\partial C_i}{\partial y}\right) + \frac{\partial}{\partial z}\left(-K_z \frac{\partial C_i}{\partial z}\right) \\ = S_i - R_i + P_i - L_i C_i + Q_i \end{aligned} \quad (2.2)$$

where C_i is the concentration of species i ; u, v, w are the wind velocity components in the x, y, z direction, respectively; $K_x, K_y,$ and K_z are the turbulent diffusion coefficients; S_i is the production rate by emission sources

for species i ; $-R_i$ is the rate of removal of species i (for example, due to dry and wet deposition); P_i is the rate of chemical production of species i (which one can break down into smaller components); L_i is the loss rate of species i by chemical reactions; and Q_i is the rate of production or loss of species i by cloud processes. All terms on the right-hand side of the equation are expressed in the form of concentration per unit time. Generally, the models solve these equations and only output the concentrations, discarding the rates which might give insight into which physical and chemical processes are significantly impacting that concentration.² There are some other instances, such as during sensitivity analysis, when the model preserves data other than the concentration, as well.

One can also subdivide the chemical processes mentioned above into their constituent contributions. For instance, Jeffries and Tonnesen (1994) have demonstrated the utility of the Integrated Reaction Rate and Mass Balance method in understanding the chemical complexities of the formation of ozone. The method partitions the chemistry process into the Hydroxyl Radical Cycle and the Nitric Oxide Oxidation Cycle. These two sunlight-driven processes are dependent upon each other to link the mechanisms of initiation, propagation, and termination which drive the reactions of ozone and its precursors. Figure 2.1 shows a schematic of this process. The top

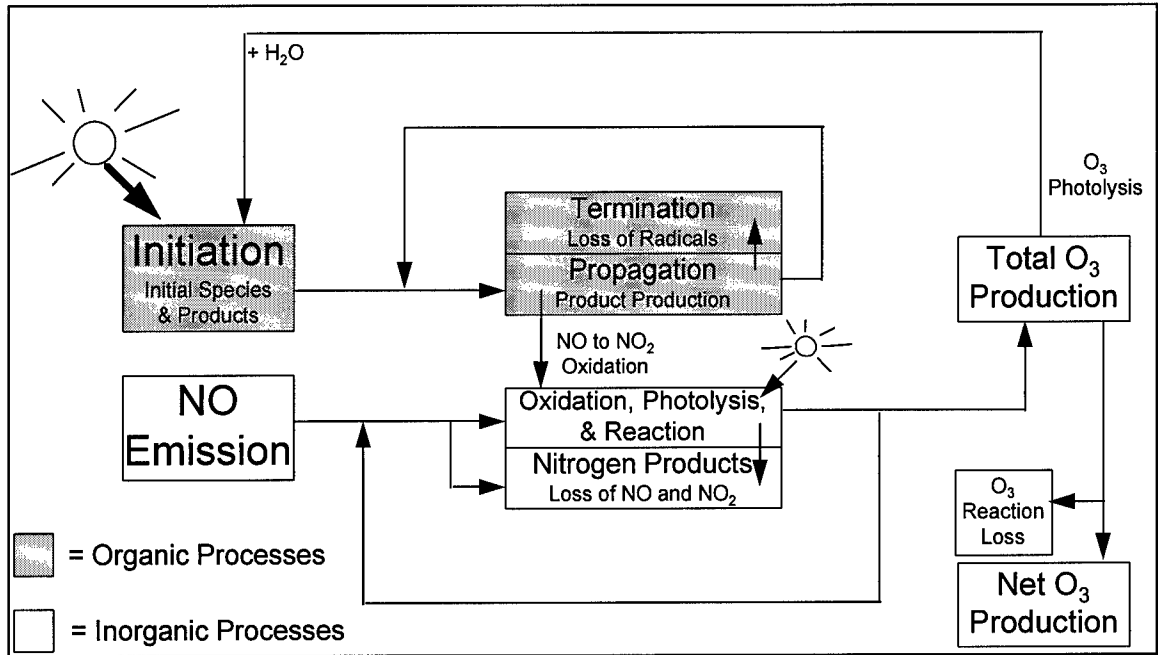


Figure 2.1: Hydroxyl Radical Cycle and Nitric Oxide Oxidation Cycle schematic from Jang, Jeffries, and Tonnesen (1995) ¹¹

half is the Hydroxyl Radical Cycle where volatile organic compounds react with hydroxyl radicals in a series of chain reactions that include initialization steps (hydroxyl and other radicals are generated by photolytic reactions), propagation steps (volatile organic compounds are oxidized and nitric oxide (NO) is converted to nitrogen dioxide (NO₂)), and termination steps (radicals are lost and combined into stable products). The bottom half is the Nitric Oxide Oxidation Cycle where nitric oxide is converted into nitrogen dioxide (reacting with peroxy radicals or oxidizing nitric oxide into nitrogen dioxide through a reaction with ozone (O₃)). Subsequently, nitrogen dioxide can be photolyzed back again to nitric oxide. Finally, oxides of

nitrogen are terminated through loss or combination into nitric acid, organic nitrates, or peroxy nitrates.¹¹ For details and derivation of the Integrated Reaction Rate and Mass Balance method, see Jeffries and Tonnesen (1994).⁵

It is through the understanding of all of these individual processes, and the analysis of their combined impacts, that one can better understand the factors which analysts must consider in designing effective control strategies.² The data the author is using in this study is the process analysis data extracted from the MAQSIP model. Using the MAQSIP prototype model, one is now able to look at the ozone data from the chemical cycles of the integrated reaction rates, as well as the advection, diffusion, and chemical data from the mass continuity processes.

2.2 Visualization

Whenever a computer graphics designer decides to visualize data, there is an essential concept that he/she must keep in mind. Tufte (1990) puts it this way, "... the essential dilemma of a computer display [is as follows]: at every screen are two powerful information-processing capabilities, human and computer. Yet all communication between the two must pass through the low-resolution, narrow-band video display terminal, which chokes off fast, precise, and complex communication."¹² To overcome this dilemma, the designer must understand the data and the visualization tools

which are available.

2.2.1 Volumetric Data

In order to display data, one must first understand what type of data it is. Volumetric data, which the author uses in this study, is the domain of a single dependent variable and the three independent variables upon which it depends, that is, equation (2.3).

$$A = F(x, y, z) \quad (2.3)$$

The three independent variables represent a point in three-dimensional space:

$$p = (x, y, z) \quad (2.4)$$

In total, a volume is defined by the domain of N distinct points:

$$p_i = (x_i, y_i, z_i) \forall i = 1, \dots, N \quad (2.5)$$

Subsequently, for every p_i there is a scalar value of A_i such that equation (2.6) follows:¹³

$$A_i = F(p_i) = F(x_i, y_i, z_i) \forall i = 1, \dots, N \quad (2.6)$$

In order to display volumetric data one defines the data over a cuberille grid. The data, therefore, is defined over N_x columns, N_y rows, and N_z vertical levels, and takes the form of equation (2.7).

$$(x_i, y_j, z_k, F_{ijk}) \forall i = 1, \dots, N_x, j = 1, \dots, N_y, k = 1, \dots, N_z \quad (2.7)$$

Figure 2.2 shows a 5 by 5 by 5 cuberille grid, and how to locate a point based on its x , y , and z coordinates. There are two primary ways for visualizing volumetric data on a cuberille grid: isovalue surface (or isosurfaces) and volume rendering.¹³

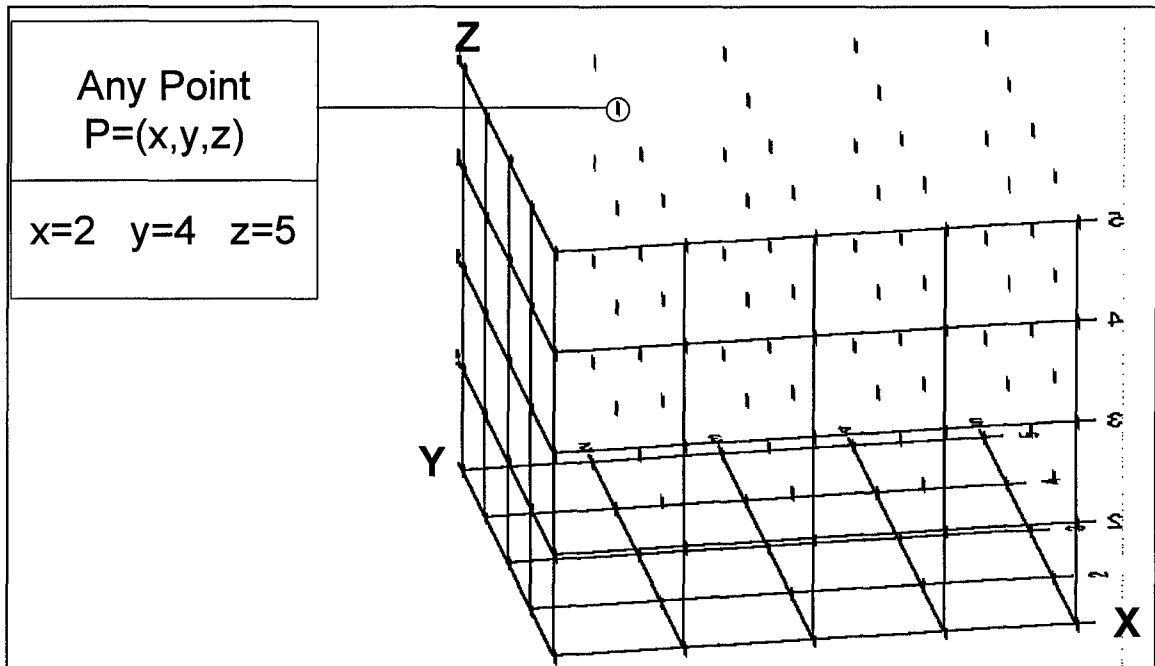


Figure 2.2: A cuberille grid where $P=(2,4,5)$ is accented

2.2.2 The Use of Color

In order to design the most effective displays possible, one must understand how to use combinations of colors effectively. One uses color primarily to accomplish the following four objectives: ¹²

1. To label a feature,
2. To measure or interpret the value of a quantity,
3. To represent or imitate reality, and
4. To enliven or decorate the presentation.

Item 4 is purely an esoteric feature of color and has little relevance to this work. As for item 1, labeling, designers accomplish this objective by displaying textual information on or near a feature. Drawing a background reference, such as a map to label the geographic location of a feature, is another example of using color to label.

When quantifying, item 2, color is a common means by which a designer can get the point across. Color is, at least perceptually, continuous in both value (ie red, blue, green, etc) and saturation, or brightness. It is also sufficiently distinct to use in comparisons of most measurements or calculations. ¹² That is not to say that the human eye can distinguish all of these distinct measurements. After all, a trained colorist may be able to distinguish between 1,000,000 colors, under the right conditions, but most people can only distinguish differences between about 20,000 colors. ¹² The

implication is that one has the technology to produce sufficiently distinct colors on a display to use in comparisons of most measurements; however, the human eye cannot perceive all of those distinct differences.

Value scales represent the primary use of color in this quantifying role. Initially a color scale could simply be a gray scale. After all, shades of gray are simply changes in saturation or intensity of the color white. Gray scales work well for defined quantitative differences; however, when one must detect small differences between non-neighboring regions, gray scales are not as useful. A scale defined by a rainbow of colors is much more distinct; however, since the human brain does not naturally recognize the rainbow color scale (red - orange - yellow - green - blue - violet, or for computer images red - yellow - green - cyan - blue - magenta) as an ordered system, these types of color scaling often require analysts to approach the scales as an encoding rather than a continuous function. In other words, an analyst first recognizes the color with which the feature is displayed, not the value that the color represents, then evaluates the represented value using a look up table or legend. Therefore, it is often necessary to use additional methods of conveying the quantitative measure when using color scales.¹²

In a video display, the primary uses of item 3, coloring for reality employ shading techniques or algorithms to present the illusion of depth and

the use of color to give features a familiar or easily recognized look, such as coloring water features blue.

When using color, one should consider certain concepts. Bright colors can be overwhelming when a designer uses them over large portions of a display or when there is very little contrast provided by dark features. It is best to use the bright colors sparingly and rely on the contrast between the feature and a dark or dull background. Color can also present perceived differences, or a lack thereof, because of interactive contextual effects (the analyst will often mistake one color, in the presence of another surrounding color, for something else).¹²

3 DEVELOPMENT

Dr Steven S. Fine and Dr J. Carey Jang, both of MCNC, and the author, hereafter referred to as the researchers, developed the visualization technique in three steps. First the researchers worked with three-dimensional visualization software to find a method to display the elements of the process analysis data. Secondly, the researchers developed a means of comparing the different contributions of the data elements on screen. Finally, the researchers incorporated a method to display the internal elements of the chemical process.

3.1 Selection Of Visualization Technology

In their initial work the researchers experimented with the Application Visualization System (AVS) software.¹⁴ They used MCNC's Package for Analysis and Visualization of Environmental data (PAVE) to pull the process analysis data sets out of the MAQSIP data files, then saved the data sets into AVS format using PAVE's built-in converter.¹⁵ These initial displays visualized the data as discrete packets.

The author used a variation on Nielson et al's (1991) Tiny Cubes method.¹³ This method plots the ozone process rate as a shaded cube with the color representing the magnitude of the rate. The total number of cubes displayed will be $N_x \times N_y \times N_z$, where, as before, N_x is the number of columns, N_y the number of rows, and N_z the number of vertical levels. The program plotted the cubes on a cuberille grid. In Nielson et al's original method, they gave each cube the same dimensions; however, the author found that by changing the cube to a sphere and tying the radius of each sphere to the magnitude of its respective rate, one could present information more efficiently and clearly.

The procedure produced a three-dimensional cuberille grid of spheres. The author made the spheres representing large decreases in ozone concentration very large and shaded them intense blue. The author also made the spheres representing large increases very large but shaded them

intense red. The author made those spheres which were closest to zero increase or decrease very small and shaded them intense green. The author graduated the shading and size of the spheres in between these values. In the black and white example in Figure 3.1, the large dark sphere in the foreground represents a large, positive magnitude, while the smaller spheres show smaller magnitudes. The value scale shows radius on the x-axis, and magnitude on the y-axis. The superimposed gray scale plots the sphere color against the magnitude, and the white arcs plot the radius against the magnitude. There are no large light gray spheres, which would represent large negative magnitudes, in Figure 3.1.

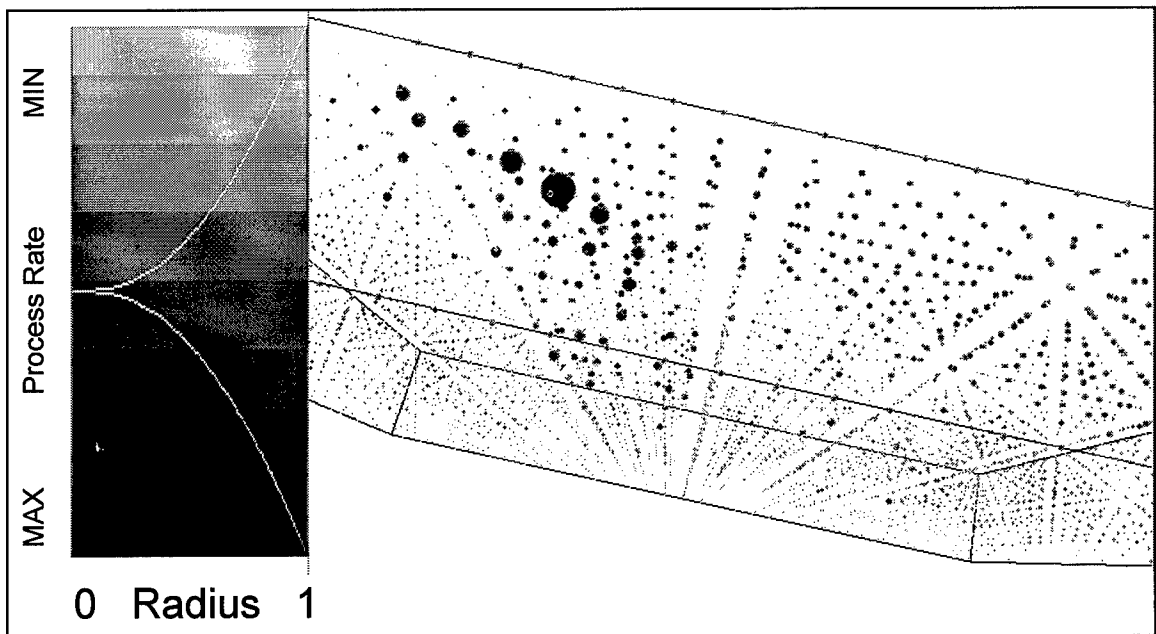


Figure 3.1: Black and white version of the variation of the Tiny Cubes method

The effect of tying the magnitude to two display features was an application of Tufte's point about color value scales. The researchers found that it worked well. The spheres which represented little change in concentration were small enough that they could easily see into the volume and identify those plots which represented large changes. And the color scale allowed them to identify the sign of the rate.

However, the AVS discrete displays did not visualize the pattern of the three-dimensional structure of the process well. These displays gave some initial feeling for the three-dimensional locations of maxima and minima, but failed to give a deeper understanding into the structure of the processes. There was no direct indication of how the values might have been changing between the discrete points. Consequently, there was no means by which to identify surfaces, or even lines, of constant magnitude, except, of course, by visual guessing. These surfaces would identify groups of points over volumes which corresponded to common sources or sinks. To find that information, the researchers had to find a means to interpolate the data into a continuous method.

The discrete data format would not allow direct interpolation of values between the given points. The description of volumetric data above, could suggest that a mathematical formula involving the independent variables produced the dependent variable. In the discrete data with which the

researchers worked, that was not the case. There was no mathematical formula. In order to interpolate the value of the dependent variable at any location other than at one of the given points, the computer interpolated a mathematical formula which could produce a value for the dependent variable from the independent variables. When a formula, or a collection of formulas defined over limits, exists which could interpolate the value for any point in the domain, the function is considered continuous.

AVS provided a method for continuous interpolation of discrete data. One of the visualization analysis methods available on AVS was planar slicing. The approach was to pass a plane through the volume of data. Then through an interpolation algorithm, the computer shaded the surface of the plane by interpolating the magnitude of the rate for every virtual point on the plane by inserting the x, y, z values for each point on that plane into the continuous function it created to represent the discrete data. The author then applied a color value scale similar to the blue-green-red scale he used for the Tiny Cubes method. In the end this display created a two-dimensional, value scaled picture, like the one in Figure 3.2. The old fashioned contour plot in Figure 3.3, showed much the same information. So, the researchers still had not found a representation of the three-dimensional structure.

Another visualization method the author tested used isovalue surfacing, or isosurfacing. Isosurfacing is the three variable function based



Figure 3.2: Value scale coloring of ozone concentration on a planar slice

equivalent of two variable function isoplething.¹³ The method used the continuous data function to locate all points, equation (2.4), for which the function is constant as in equation (3.1).

$$F(p) = \text{constant} \quad (3.1)$$

This method then drew small triangular planes between all adjacent points which satisfy the function. The user could then change the value of the *constant* to see the three-dimensional surfaces on which the rate was equal to

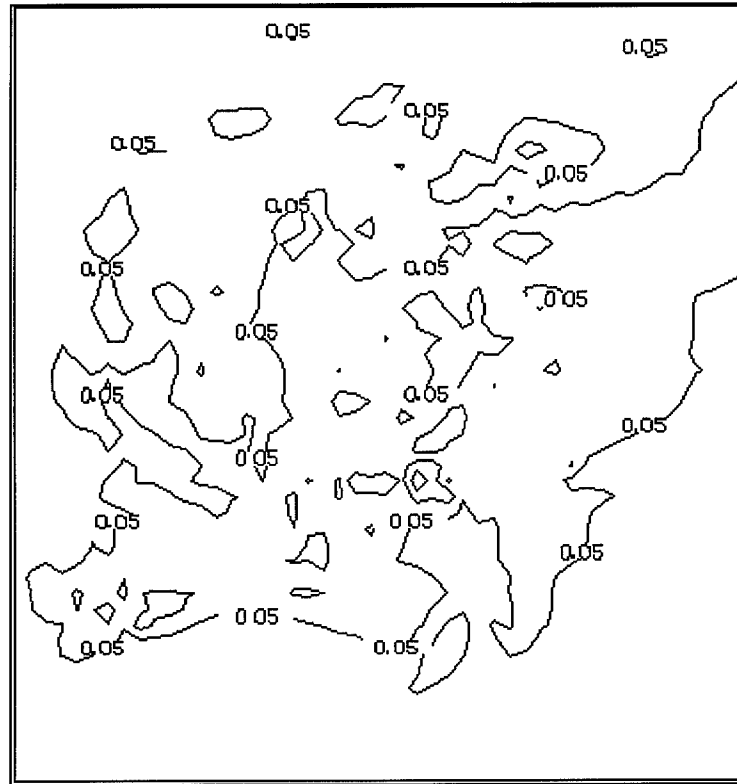


Figure 3.3: 0.05 ppmv interval contours of ozone concentration on a planar slice displaying the same data set as in Figure 3.2

the *constant*. The author used a few of these isosurfaces in the case studies discussed later, such as the one in Figure 4.10. This technique worked well; however, since the surfaces only represent one value, the changes between surfaces were not apparent. Additional surfaces could help; however, when one surface was contained within the bounds of another, the inner surface was obscured. Additionally, when one value produced a multitude of individual surfaces, the display became too complex to understand.

Modified Tiny Cubes, planar slices, and isosurfaces gave some new impressions of the structure, but they still were not sufficient tools to give a

complete structural visualization. The next logical step was a volume imaging method. Volume imaging depicted a three-dimensional scalar as a transparent fog, where it tied the transparency to the magnitude of the scalar. The method treats the entire volume as a series of flat surfaces stacked on top of each other. The computer produced a planar slice for each of these flat surfaces and included the transparency into the slice. Then it laid one slice on top of another, and used a mathematical algorithm to calculate the new transparency and color. The computer repeated this step for each successive layer from the one that was, virtually speaking, the farthest from the viewer, to the closest, until it rendered the volume.¹⁶ Once again, the author used this technique in the case studies, for example, in Figure 4.23.

At this point in time, the researchers were faced with a dilemma. The license to use AVS expired, and they had no idea how long it might take for the company to issue a new one. So, the researchers needed a different program. They wanted a software package which would allow them to duplicate many of the features they had already found in AVS. The researchers also wanted something which would be able to handle a volume imaging method for volume rendering. They found their answer in Vis-5D, a “literal and intuitive data” depiction program created and distributed as

freeware by the University of Wisconsin Space Sciences and Engineering Center.¹⁷

Vis-5D, now in version 4.2, created a virtual Earth environment from five-dimensional data sets. These sets were two-dimensional arrays of three-dimensional spatial grids. The two-dimensions in the array were time (temporal) and a scalar variable.⁴ In other words, it was much the same as volumetric data, but it included a fourth, independent, temporal dimension, t . The function would be similar to a volumetric one, equation (3.2).

$$A = F(x, y, z, t) \quad (3.2)$$

The fifth-dimension, which the name, Vis-5D, implies, is the choice of which function or field of data to present as three-dimensional vectors and polygons, and rendered volumes.⁴

The researchers extracted just the ozone data sets from the MAQSIP process analysis data using a Models-3 data extraction utility. Process analysis data on MAQSIP is limited to the lower troposphere, the first six sigma levels (approximately 20 - 300 m); therefore, the researchers extracted the ozone concentration data for only the lower six sigma levels of the concentrations data set using the same utility. They also extracted the u and v wind component data from the MAQSIP data set.

MAQSIP stored wind data differently than other data sets. In any given cell, the model located the process analysis and ozone concentration data in the center of that cell. On the other hand, it located the wind data at the corners of that cell. Since Vis-5D only accepted data sets defined on one grid, the researchers had to adjust the wind plotting locations. In order to fit the two types of plotting data together, they extracted all of the wind plots except for the extreme east column and the extreme north row, as well as only the lower six sigma levels.

The researchers accomplished the conversion of the MAQSIP data into Vis-5D format quite easily with the Models-3 to Vis-5D, *m3tov5d*, conversion routine developed by the U S Environmental Protection Agency's Visualization Lab. However, Vis-5D interpreted the location of a data point to be at the lower left-hand corner of a cell, while the MAQSIP located the data in the center. This transition introduced a spatial off-set of approximately 38 kilometers (km) southwest into the data fields of the 54 km horizontal grid spacing data sets the author used, and 25 km for the 36 km sets.

Because of a problem with one of the Vis-5D utility programs the author used, he had to edit the Vis-5D data files to adjust the vertical levels. The simplest adjustment was to assign a generic, equally spaced level to each of the sigma levels. The author used *v5dedit* to change the bottom sigma

level to 0, the next sigma level to 1, etc. The author then adjusted the Lambert Conformal projection coordinates of the wind data to align it with the process analysis and concentration data using *v5dedit*. This change adjusted the wind fields approximately 38 km northeast in the 54 km data sets, and 25 km in the 36 km sets. Note that the two spacial adjustments cancelled each other out.

To combine the three data sets into one Vis-5D file, the author used the *v5dimport* program. Once the author had all of the data sets aligned on one cuberille and temporal grid, he used *v5dedit* to adjust the vertical levels back to their equivalent heights. The author also used *v5dedit* to correct the spatial error that the converter introduced. The author added one half of a column length to the column of the north/south pole and one half of a row length to the row of the north/south pole. This adjustment negated the spatial off-set the converter introduced thus setting the process analysis and concentration data back to their proper locations; however, it reintroduced the off-set of approximately 38 km northeast to the wind fields of the 54 km data sets, and 25 km to the 36 km sets. The author documented the complete, data conversion process in Appendix 7.2.

Vis-5D provided many methods for viewing the process analysis data. The volume imaging offered a means to better visualize the structures of the areas of increases and decreases in ozone concentration. The vertical and

horizontal planar slices through the volume were also readily available (as both a color shaded plane surface and as contours) as well as the isosurfaces. Vis-5D also included a data probe which could interpolate the value of each data field at any given point in the four-dimensional space. What was missing was a graphical means by which the researchers could compare the impacts of each of the three major processes, advection, diffusion, and chemical reaction.

3.2 Designing New Techniques

Vis-5D offered many ways to display multiple fields of data simultaneously. However, the differences between the display modes themselves, and sometimes, even the similarities, made comparisons difficult. In addition, the differences in the display of different data sets within the same display mode could make comparisons difficult. The researchers could have displayed the different data sets using the same method in several different windows, even on different terminals; however, to be effective, they had to accomplish the comparison between different, yet interrelated data sets within the range of the user's eyespan.¹²

Comparison is the very essence of quantitative reasoning, and small multiple designs are often the best design solution for displaying comparison.¹² The small multiple concept is to use multiple iterations of the

same design with slight variations, representing the differences in the data. Each of the design iterations should be small enough for an average human to make the comparison with as little eye or body movement as possible. So, in order to provide a graphical evaluation of the relative contributions of each of the three processes, the researchers went with a small multiple format. Following Fine and Mathur (1996),⁶ the researchers used a star or diamond plot design which represented the magnitudes of different variables by lines radiating from a central point. The length of each line represented the magnitude of the variable it represents. Each of the lines, in themselves, was a small multiple. An analyst could compare the individual contributions of each process at a glance. A consistent display method put the emphasis of analysis on the differences in the common feature, the lines.¹²

For example, the lines in Figure 3.4 represent different rates. Most people would recognize Figure 3.4 as a bar graph. The difference between this bar graph and most is the location of the base value, or starting point. Here it is at -50, whereas, it would normally be at zero (0). The bar graph is a small multiple, in its most simplistic form. The point where the right hand end of the line ends marks the magnitude. Therefore, a very short line ending to the left of zero represents a negative value, and a very long line a positive value. Without even looking at the scale, one can determine that chemical processes contributed the most and advective processes removed the

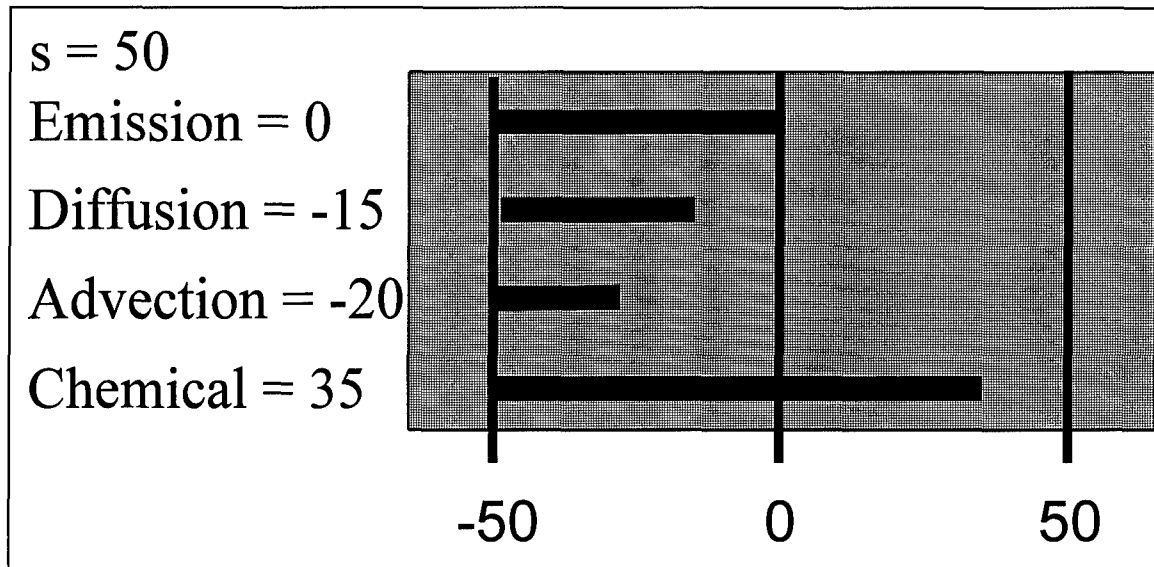


Figure 3.4: Small multiple of process rates

most.

The diamond small multiple plot the researchers designed, such as the one in Figure 3.5, used four lines within a diamond shaped figure to represent the contribution of the following processes:

- 1) Chemical processes (production rate) represented by the downward vertical line.
- 2) Advection processes (rate) represented by the right hand horizontal line. The magnitude is the sum of the MAQSIP process outputs from horizontal and vertical advection rates of ozone.

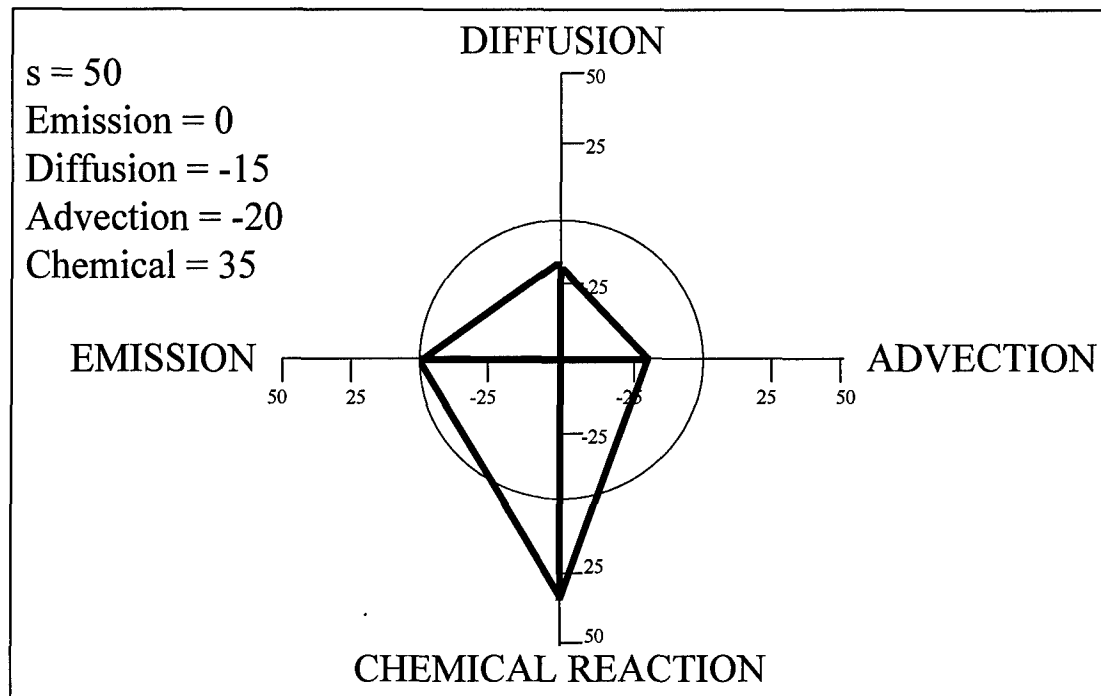


Figure 3.5 Diamond plot small multiple representing the same information as in Figure 3.4

- 3) Diffusion and deposition processes (rate) represented by the upward vertical line. The researchers combined the removal rate due to dry deposition with the rate of change in concentration due to the mixing effects of horizontal and vertical diffusion. However, the contribution of horizontal diffusion was very small compared to the other processes, as one would expect for a large scale grid model. They also added a correction factor, which included certain loss terms and was generally 10 orders of magnitude smaller than the other processes.
- 4) The contribution by source emissions represented by the left hand

horizontal line. In the study of ozone, this line always represented a magnitude of zero since ozone is a secondary pollutant and is, therefore, not emitted into the atmosphere. The researchers included this feature for use in future research which might study other compounds.

The scale on the diamond plots was radial. The center of the diamond represented the strongest negative contribution, which the radius of the circle drawn in the plot specified. The points on the circumference of the circle, where the various process lines intersect it, represented zero contribution. Lines extending beyond the circle represented positive contribution. Lastly, the researchers connected the extreme ends of each line to draw a diamond or quadrilateral. The pattern or shape of these quadrilaterals was the feature one could compare from one data point to another to understand the comparative contributions of the processes at different locations at different times.⁶ The small multiple in Figure 3.5 actually represents zero net change, or a steady state, in ozone concentration since the processes cancel each other out mathematically.

By manipulating the source code for Vis-5D and incorporating commands from a graphics drawing package, the author produced a small multiple Graphic Interchange File (GIF) image based on the process analysis factors at any given location of the Vis-5D probe. The author used gd 1.2, a

graphics library of calls to draw images using C code and saved them in a GIF file. This program was created and distributed by Quest Protein Database Center, Cold Spring Harbor Labs.¹⁸ The author's code appears in Appendices 7.1.1 - 7.1.7.

For an instantaneous evaluation of the processes at a given location, this feature was fine; however, comparison of features between different cells could provide more insight. Once again the eyespan factor came into play. The researchers needed a means to compare smaller scale versions of the diamond plot against other locations on the same screen. Therefore, the author modified the code again to be able to produce multiple diamond plots on a background map. The author also added a feature to allow the user to modify the scale of the diamond plots in order to extract more detail from the diagrams when the magnitudes of the processes are low. Figures 4.24, 4.25, and 4.26 are examples of the diagram produced from the results of some case studies. Julian date (D), time (T), and level (Z) are to the right of each plot.

The researchers still wanted to find a visualization method to display the individual process components of the integrated reaction rate and mass balance computations which contributed to the chemical reaction process rate. Computation of these components required an additional post-processor. Since the post-processor produces data for only user-specified locations, the researchers did not have volumetric data of these processes.

Developing the volumetric data would have taken considerable computation time, which the author did not have. Future research may benefit from producing volumetric versions of this data for analysis. The author used the Vis-5D data probe to send the user-specified time and location to the post-processor, and then used the gd graphics library to display the results in a schematic representation of the Hydroxyl Radical Cycle and Nitric Oxide Oxidation Cycle similar to Jang, Jeffries, and Tonnesen's (1995) diagram in Figure 2.1.¹¹ These diagrams, though individual, and independent, provided the analyst with the contributions from each of these cycles to the formation and destruction of ozone due to chemical reactions. Unfortunately, the post-

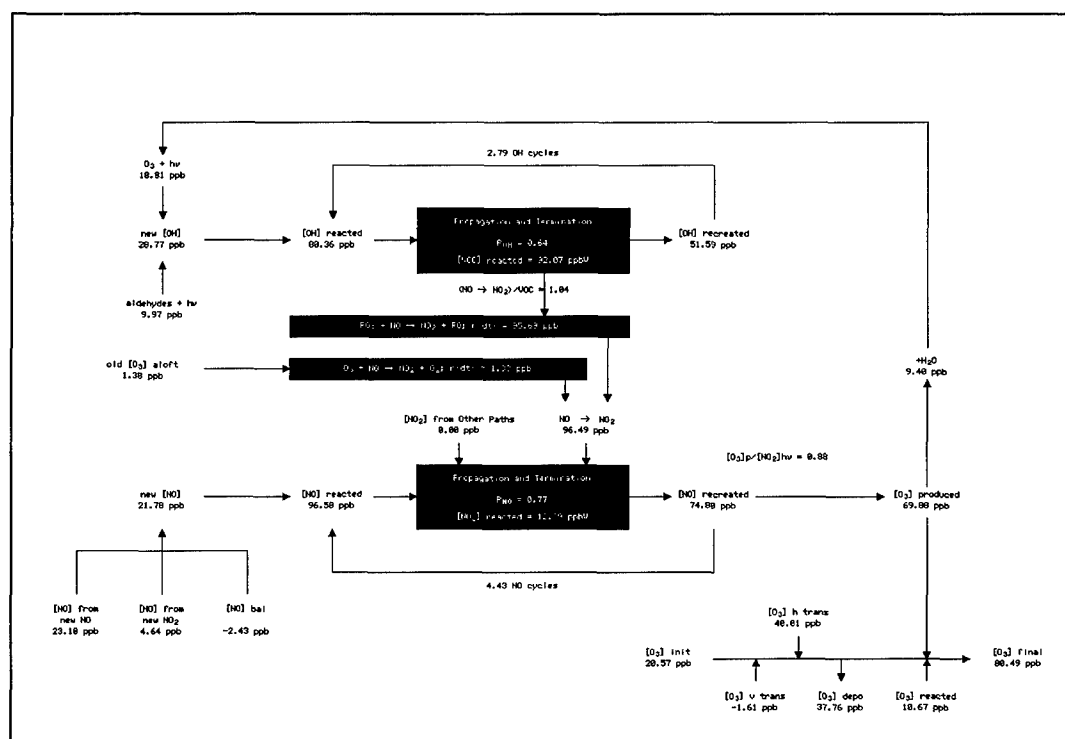


Figure 3.6: Computer generated cycle schematic

processor was not finished at publication; so, the only example was a canned data set product shown in Figure 3.6.

4 CASE STUDIES

To test the visualization methods and evaluate their effectiveness, the author studied five high ozone events in a period of seven days. The case study simulations are characterized by the following parameters of MAQSIP:

- 1) The horizontal spatial resolutions are 54 or 36 km. The area encompasses a Lambert Conformal projection of a region of the eastern United States, shown in Figure 4.1.
- 2) Calculation of the process analysis data is limited to the lowest six MAQSIP sigma (σ) layers. The equivalent heights are, approximately, 20 meters (m), 60 m, 100 m, 140 m, 200 m, and 296 m.
- 3) The temporal resolution of the output is 1 hour (h). In the 54 km data set, there were data from 7/1300 - 12/1200, and in the 36 km data set, there were data from 12/1200 - 13/1800.



Figure 4.1: Map of case study region

- 4) Meteorological data comes from the Mesoscale Model Version 5 (MM5). In order to make more accurate assessments of the advection terms, MAQSIP evaluates the west to east component of wind velocity (u) and south to north component (v) at grid locations

shifted such that if the process analysis values are located at the center $(1.0,1.0)$ of a grid cell, the wind components associated with that cell are located at the four corner points, which define the horizontal boundaries of the cell: $(0.5,0.5)$, $(0.5,1.5)$, $(1.5,1.5)$, $(1.5,0.5)$. See Figure 4.2.

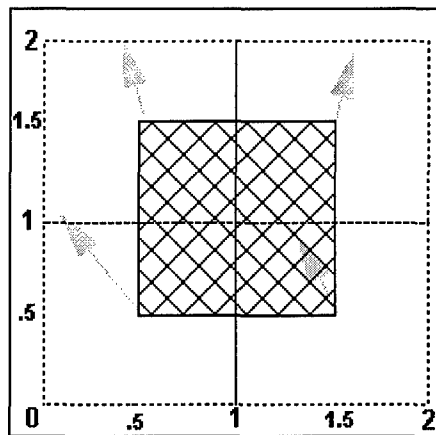


Figure 4.2: Concentrations are calculated at the center of the hatched cell $(1,1)$. u and v wind components are calculated at the centers of the four empty cells, at the base of the vectors $(.5,.5)$ $(1.5,.5)$ $(1.5,1.5)$ $(.5,1.5)$

- 5) The chemical initialization data set comes from a standard, initial data set of background concentrations.
- 6) The *m3tov5d* converter program had a problem converting the “horizontal diffusion” field at 13/1000 Coordinated Universal Time (UTC) in the 36 km resolution data set. To compensate, the author entered a field of all zeros for that one time step in that one field.

Since the order of magnitude of this field is generally significantly smaller than the other processes, this correction should have negligible effect on the display.

Three of the cases were on a 54 km grid size resolution; the other two had a 36 km resolution. In each of these cases, the model's predicted ozone concentration for at least one grid cell in the location of interest exceeded 0.12 ppmv for more than one, hour-long, time step. The cell references in these studies were in the following format: (*row,column*), where (1,1) was the cell in the most northwest corner of the grid. One should note at this point that this method was Vis-5D's grid reference. In the original data from MAQSIP, the cell (1,1) was in the most southwest corner of the grid. The *m3tov5d* conversion routine changed the cell references from MAQSIP's method of navigation to Vis-5D's method when it converted the data. To identify the MAQSIP cell reference, subtract the Vis-5D row number from the total number of rows (72 for 36 km resolution or 54 for 54 km resolution) and add one (1). In addition, the author will be using the term "vertical column" in these cases. These "vertical columns" refer to the stack of all cells over a single reference cell at the ground surface. The reader should not confuse these "vertical columns" with the term "column" the author used to refer to the second number in a cell reference. The author will give time references; if they are unlabeled, the reader should assume that they are in UTC.

4.1 Houston, Texas, July 8, 1995 (54 km Resolution)

In this case, the ozone concentration exceeded 0.12 ppmv from 8/1700 - 9/0200 UTC (8/1200 - 2100 Central Daylight Time (CDT)) and reached a maximum concentration of 0.179 ppmv. The area of ozone accumulation was actually southeast of Houston (38,9) over Galveston Bay (39,10). See Figure 4.3. The winds were primarily northeasterlies, and since the ozone exceeded the standard in several cells west of (39,10), the author labeled this case a weak downwind event.

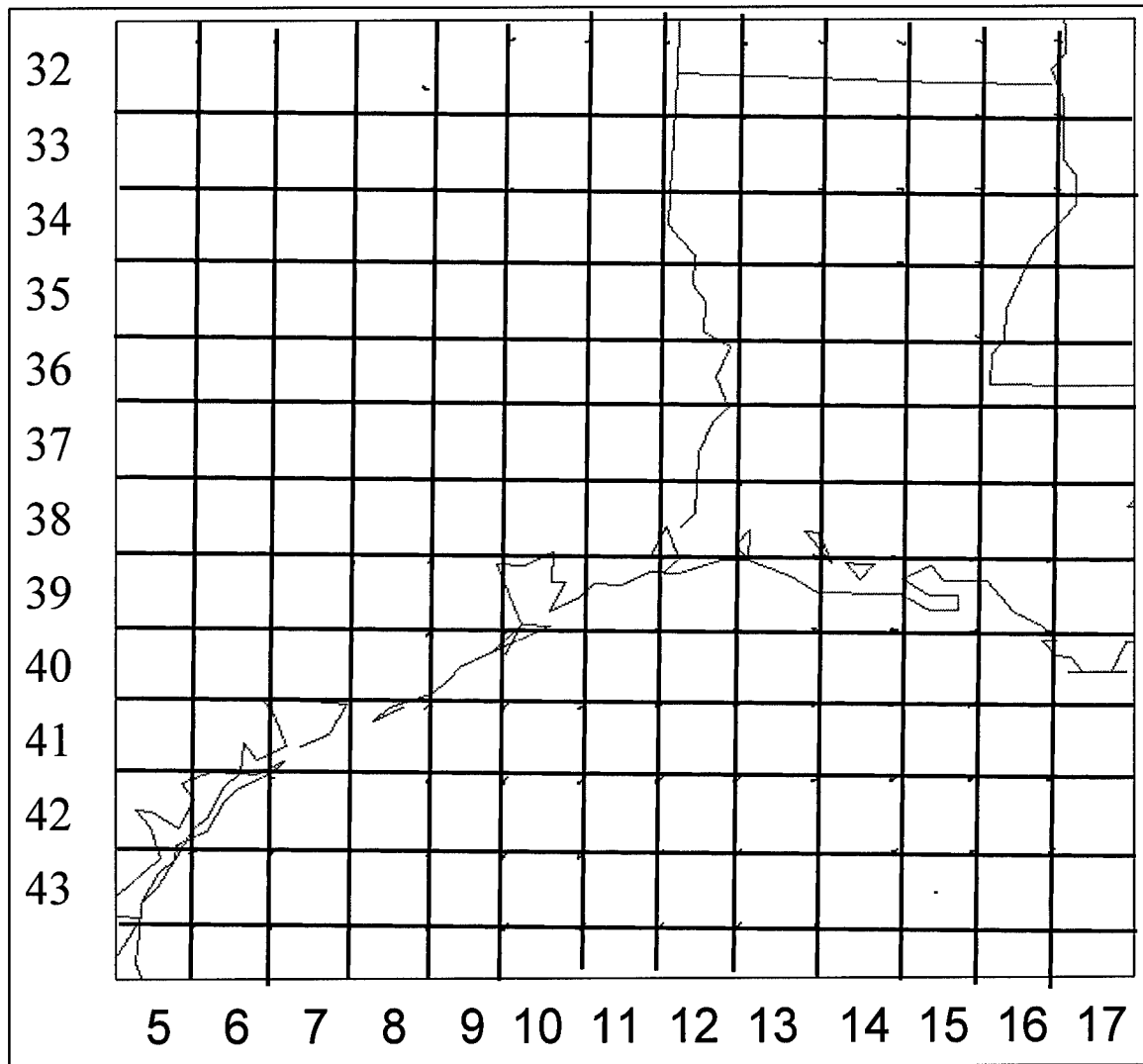


Figure 4.3: Map of cell reference grid over Houston, TX (38,10)

When the author compared the small multiples for the 20 m level at cell (38,9), (38,11), and (39,10) in Figure 4.4, he found that all processes were relatively equal early in the period. The notable exception of the diffusion and deposition contributions at (39,10) got the author's attention. From this basic information, the author moved into individual process analysis using

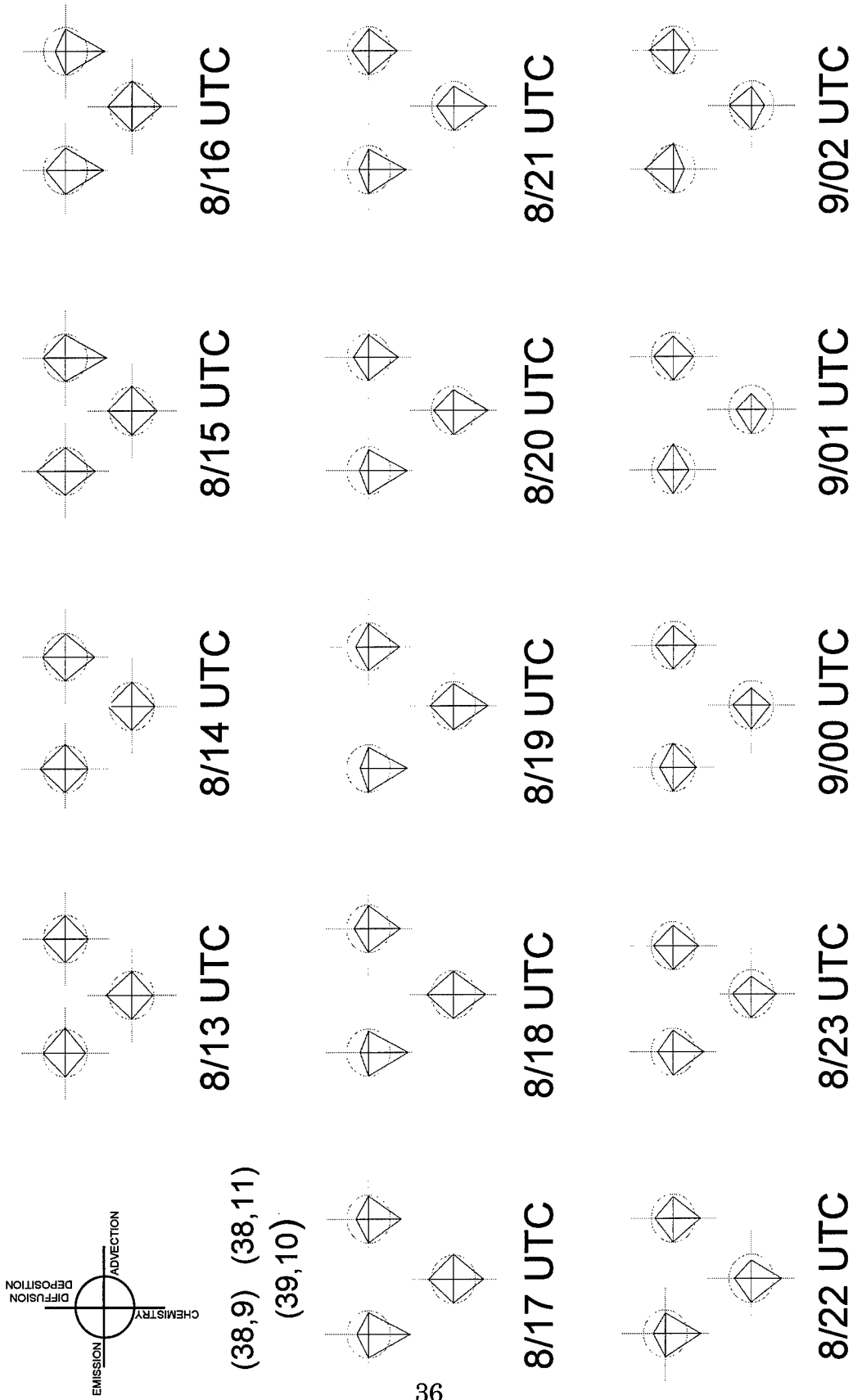


Figure 4.4: Small multiples for Houston, TX case. Radial Scale is -50 to +50 ppbv/h.

planar slices.

The chemical processes which produced the ozone for this case began early, around 1500, and became very strong. The cells with the highest rates of production were (38,9) and (38,11). They both exceeded 30 ppmv/h at 1600, as in Figure 4.5, where the contours are in increments of 10 ppmv/h. However, neither of these cells accumulated enough ozone to exceed the

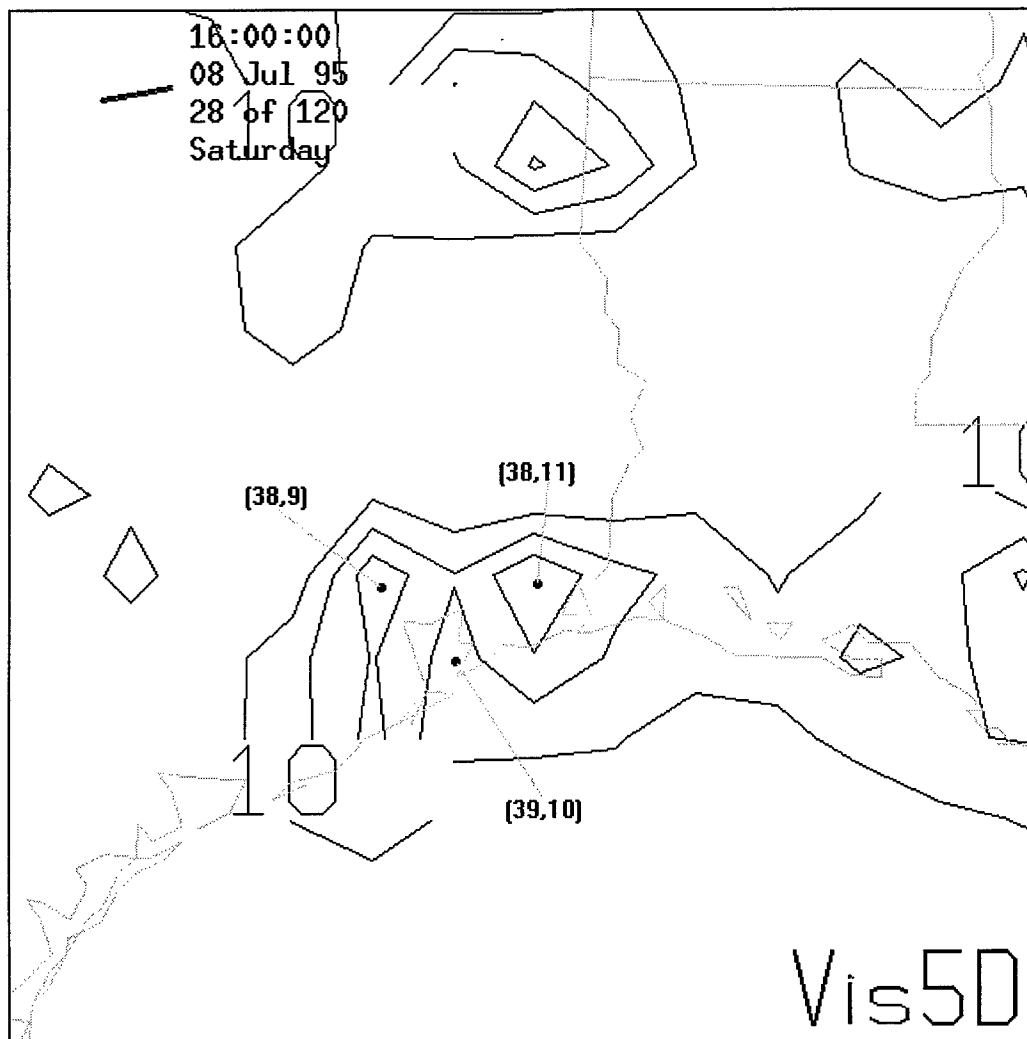


Figure 4.5: Planar slice of chemical process contours for the Houston, TX case

standard. Meanwhile, the chemical production in cell (39,10) was much lower than those of (38,9) and (38,11) early in the period (10-20 ppmv/h at 1600 in Figure 4.5), but became moderately strong by late afternoon.

The advection processes had little to do with the early rise in ozone concentration. The winds were light in the early morning and did not pick up until afternoon. In fact the advection rates were negligible until after the ozone had exceeded the standard. As the winds picked up in the afternoon, advection processes moving ozone out of (38,10) slowed the overall rate of ozone accumulation, indicated by the ozone concentration data. Downwind, the winds were the primary cause for high ozone concentrations along the coast, southwest of Houston, as advection moved the ozone from cell to cell until late in the evening when the chemical termination processes and dry deposition reduced the concentration.

The key factors in this event appeared to be vertical diffusion and dry deposition. In the vertical columns above each of these three key cells in this event, dry deposition was the main factor in the bottom layer (20m) and diffusion was the primary factor aloft. The combined rates of deposition and diffusion in cells (38,9) and (38,11), where the ozone production was highest, and, in fact, in all of the cells in the area surrounding Houston except (39,10), showed strong rates of ozone removal. At 1700, the light contours at 5 ppmv/h intervals in Figure 4.6 show the strong contribution of less than

-20 ppmv/h at (38,9) and (38,11). However, in that one cell, (39,10), where ozone reached its highest concentration of the day, the combined rate was much closer to zero during the early formation hours, and did not remove

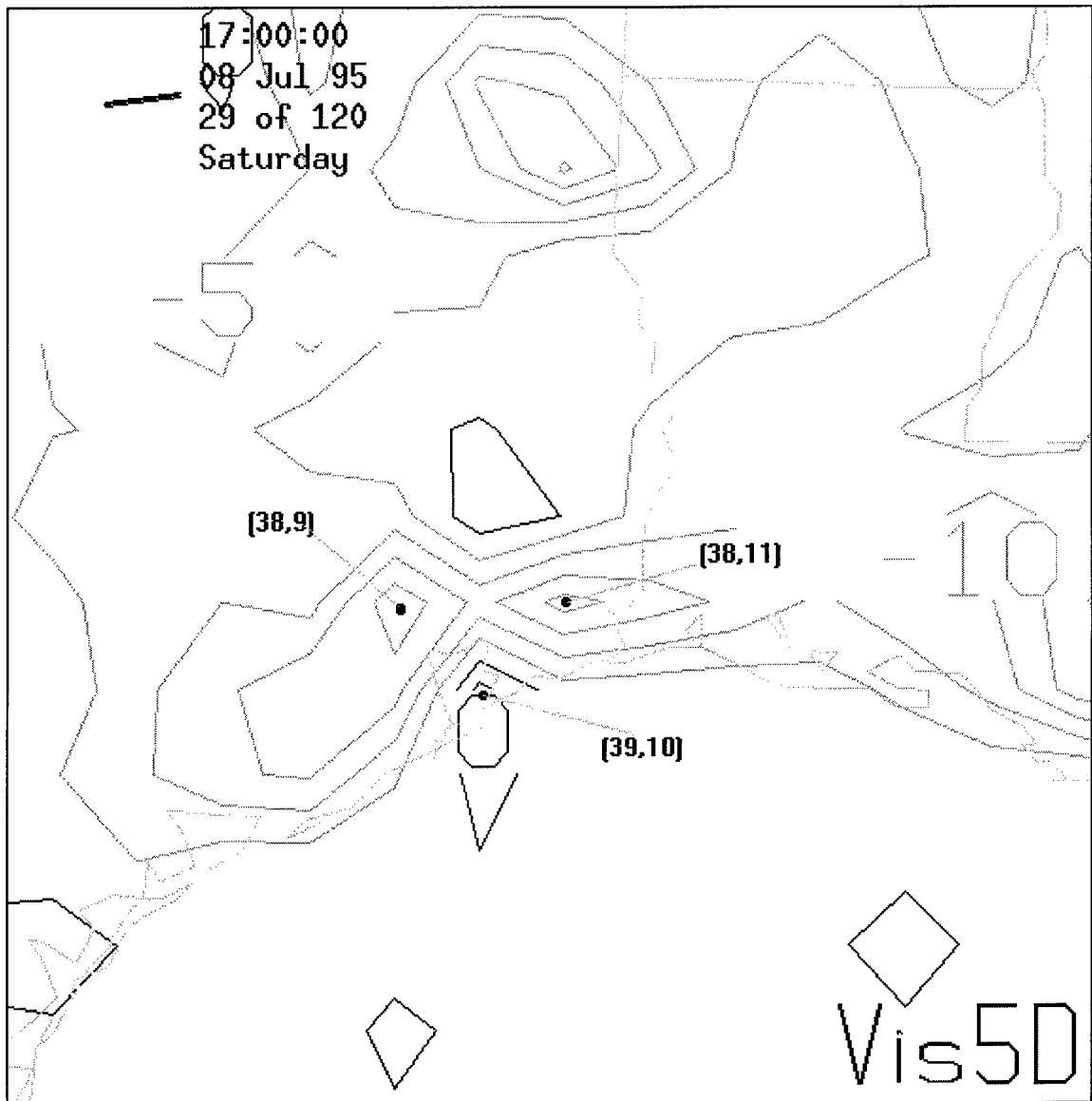


Figure 4.6: Planar slice of diffusion and deposition process contours for the Houston, TX case

significant amounts of ozone until late in the day. At 1700, the dark contours of ozone addition in Figure 4.6 were over (39,10). Most likely, since cell (39,10) was located directly over Galveston Bay, and had a landuse description of water, the lack of dry deposition and the less turbulent morning air over the water kept these removal processes low until the afternoon. The lack of removal by deposition and diffusion allowed ozone to accumulate above the standard.

The three-dimensional isosurfacing and volume imaging gave little additional information since the concentration and process analysis values were consistent from the bottom to the top of the study data.

4.2 Baton Rouge, Louisiana, July 9, 1995 (54 km Resolution)

In this case, the ozone concentration exceeded 0.12 ppmv from 9/1900 - 10/0000 UTC (9/1400 - 1900 CDT) and reached a maximum concentration of 0.176 ppmv. The area of ozone accumulation was actually east of Baton Rouge (38,17) over Lake Pontchartrain (38,18), which is just west of New Orleans (38,19), as shown in Figure 4.7. There was accumulation of ozone east of New Orleans (38,20-21) as well; however, its maximum concentration only reached 0.124 ppmv. The winds were primarily westerlies, and since the ozone exceeded the standard east of Baton Rouge and subsequently east of New Orleans, the author labeled this case a downwind event.

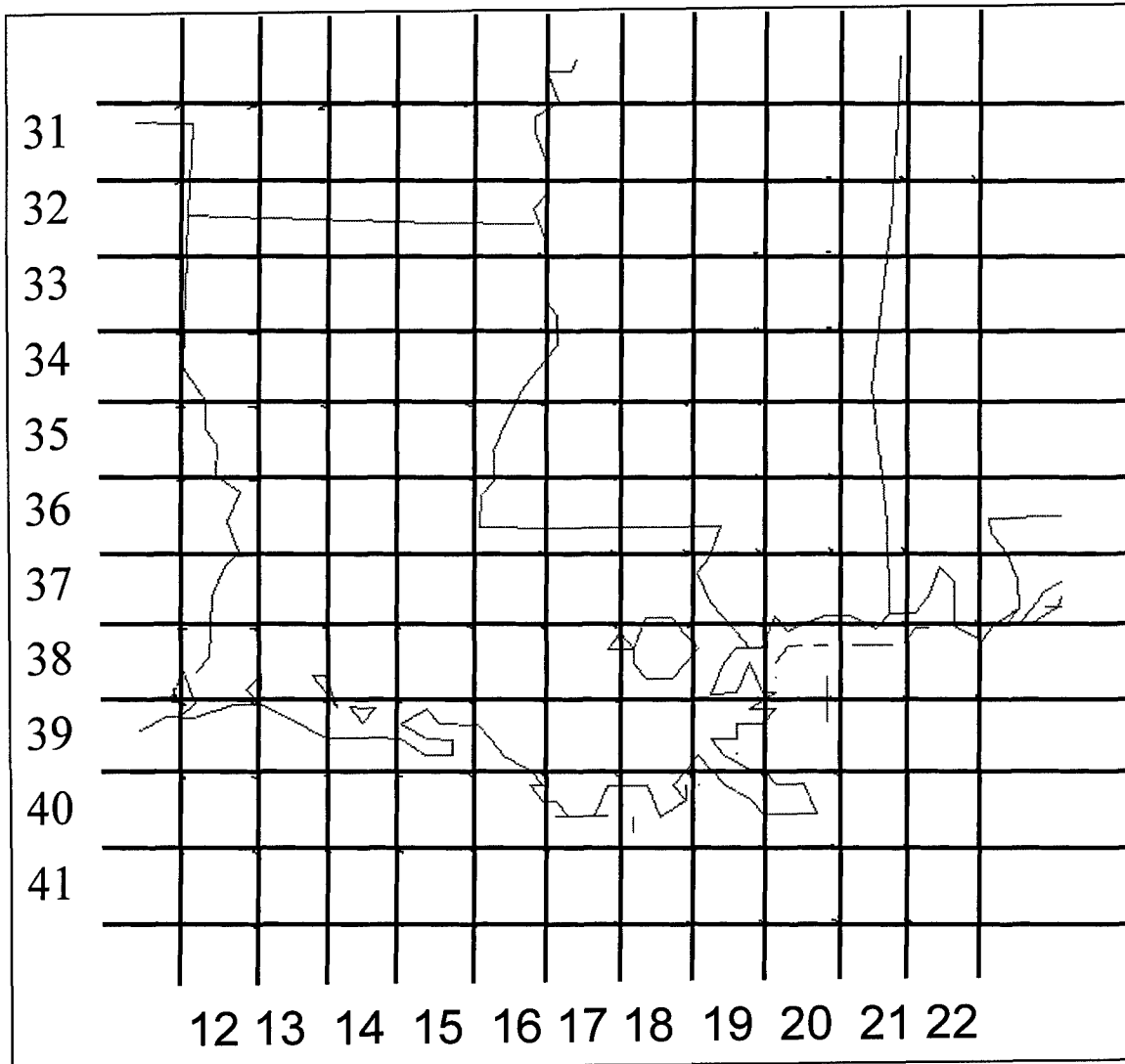
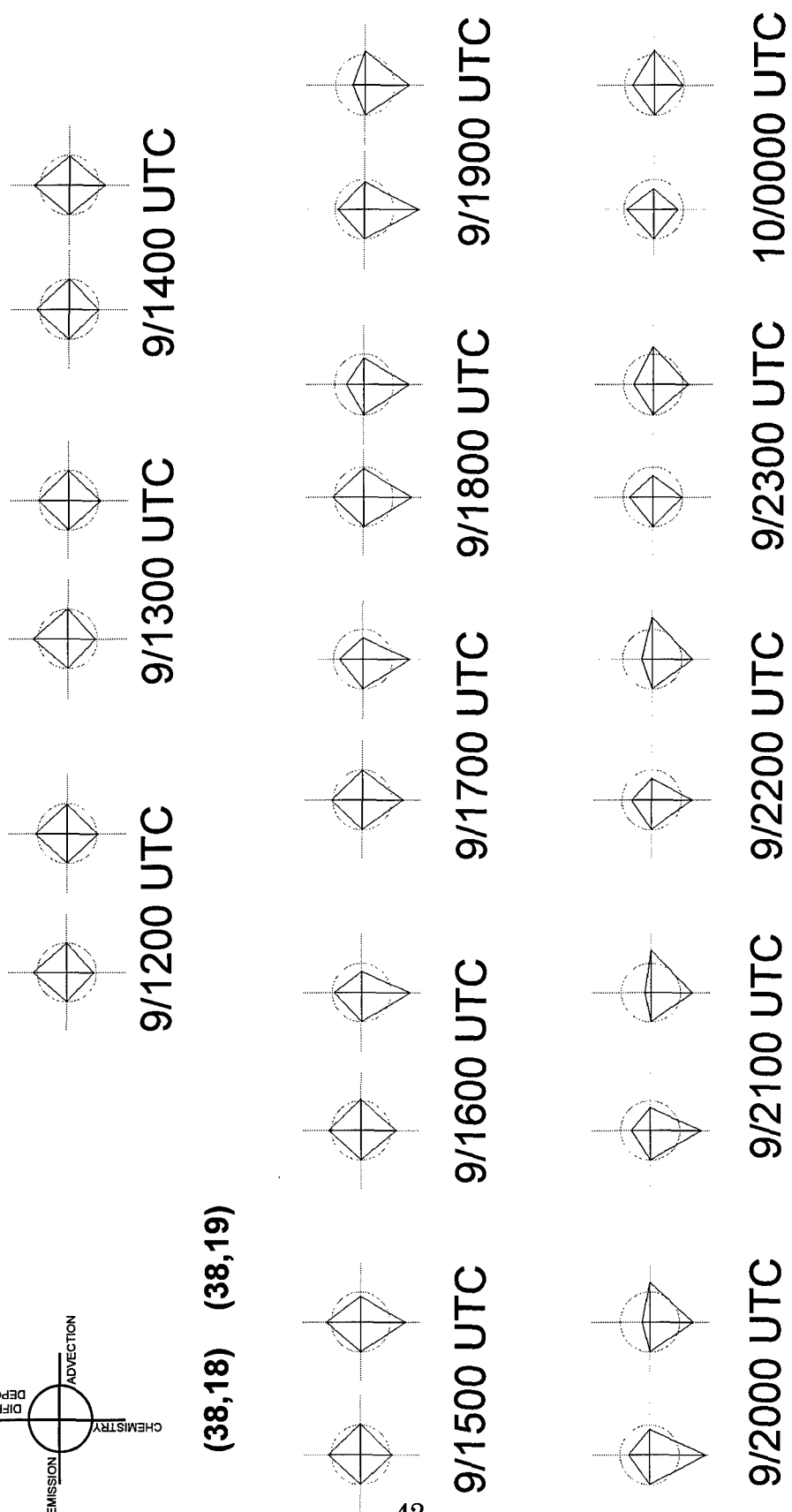
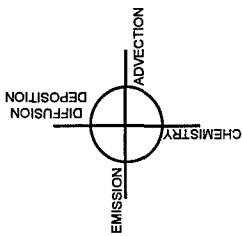


Figure 4.7: Map of cell reference grid over Baton Rouge, LA (38,17)

When the author compared the small multiples for the 20 m level at cell (38,18) and (38,19) in Figure 4.8, he found that chemical processes were dominant at (38,18) and diffusion and deposition were dominant at (38,19) especially after 1700. The surprising factor was the lack of contribution from advection, in the presence of the strong winds, until late in the period when



(38,18) (38,19)

Figure 4.8: Small multiples for Baton Rouge, LA case. Radial Scale is -50 to +50 ppbv/h.

advection in cell (38,19) changed from a removal term to a source term after 1900. Considering this basic information, the author moved into individual process analysis using planar slices.

Strong westerly winds, on the order of 4 meters per second (m/s), appeared to have advected many ozone precursors from Baton Rouge out over the lake (38,18) by 1700. These precursors fed the chemical processes which produced the ozone. The chemical processes became very strong by 1500 all around Baton Rouge, except on the north. By 1900, (38,18) was the heart of all chemical production of ozone in the area with a maximum rate of 40-50 ppmv/h directly overhead and slightly weaker (20-30 ppmv/h) in (38,19), as displayed in Figure 4.9.

Total advection of ozone in (38,18) was nearly zero until after 2000, and this apparent lack of advection led to the accumulation. Since the winds were a strong influence in this event, the author knew there had to be something more going on here. The author switched back to looking at the individual contributions of horizontal and vertical advection. The strong westerly winds were moving significant amounts of ozone out of the high ozone cells; however, vertical advection was bringing in a nearly equivalent amount of ozone canceling out the horizontal advection. Since total advection was moving little to no ozone out of cell (38,18), ozone accumulated.

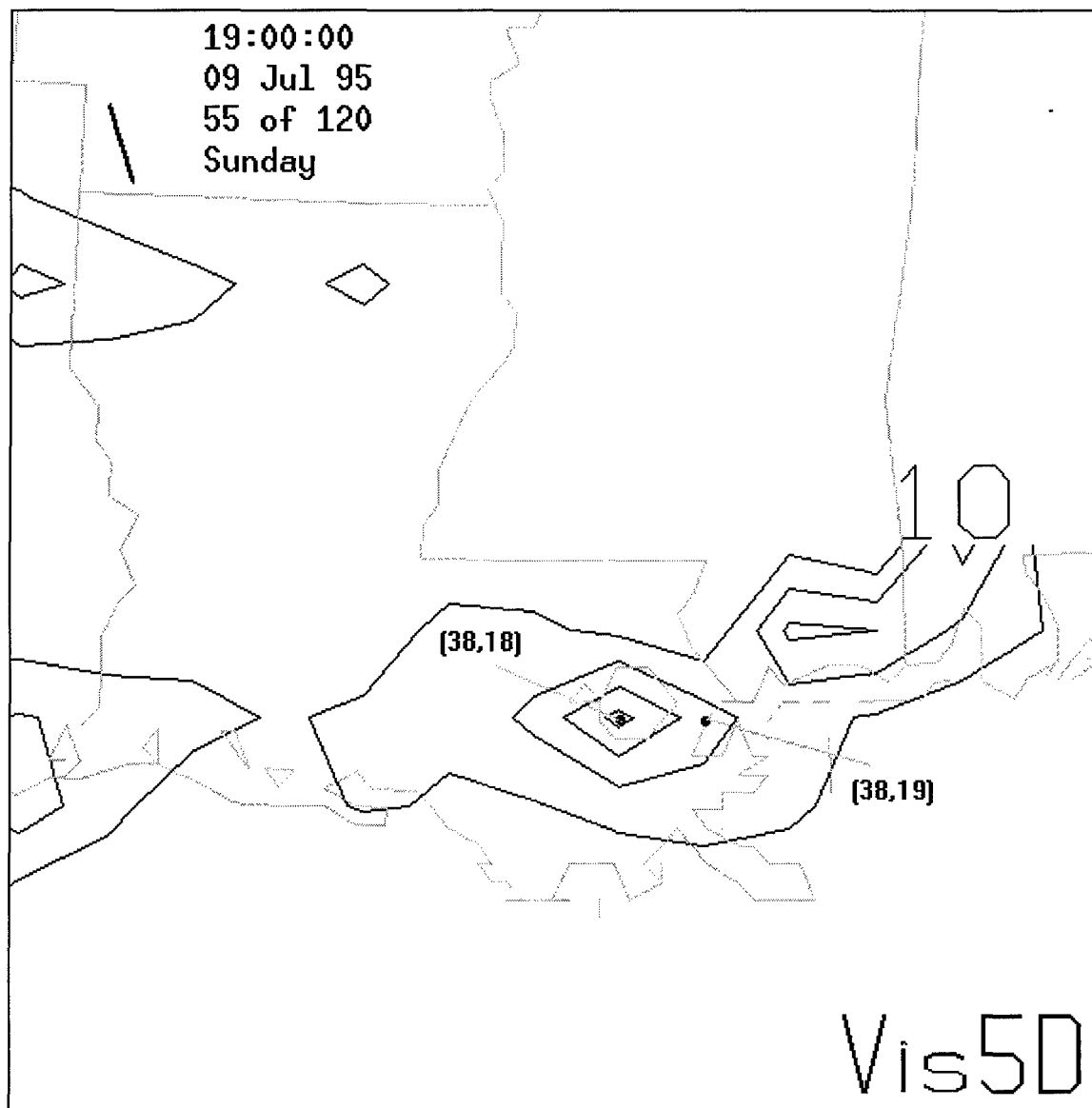


Figure 4.9: Planar slice of chemical process contours for the Baton Rouge, LA case

Once again, diffusion and deposition played key roles in the event. In the vertical columns of air above the cells, dry deposition was the main factor at 20 m and diffusion was the primary factor aloft. The combined rate of deposition and diffusion around Baton Rouge was strong early, except north

of the city, and in cell (38,18). The low contribution at (38,18), was similar to the Houston event. The landuse description over the lake was water. The lack of dry deposition and the less turbulent morning air over the water kept the diffusion and deposition low. This low contribution combined with the low advection and high chemical production, allowed the ozone to accumulate to its maximum in that cell. Cell (38,19), directly over New Orleans, had a landuse description of coniferous forest which, unlike the water coded cells, promoted diffusion and deposition. Interestingly enough, in cell (38,19), the diffusion and deposition rates were nearly twice the rate of the vertical advection, but of the opposite sign. Therefore, diffusion and deposition canceled out the vertical advection process, which the author mentioned earlier had canceled out the horizontal advection, creating a strong ozone sink directly over New Orleans. In fact the three-dimensional isosurfacing technique showed the blocking effect of the diffusion aloft in Figure 4.10. The light gray surface encloses the vertical column of ozone concentration greater than 0.12 ppmv, and the dark surface encloses the vertical column of diffusion contribution less than -20 ppmv/h. The stronger removal accounted for the lower ozone concentrations directly over New Orleans, while the vertical columns just east and west of the city had high ozone concentrations. Later in the period, around 2200 to 2300, when the diffusion rate halved

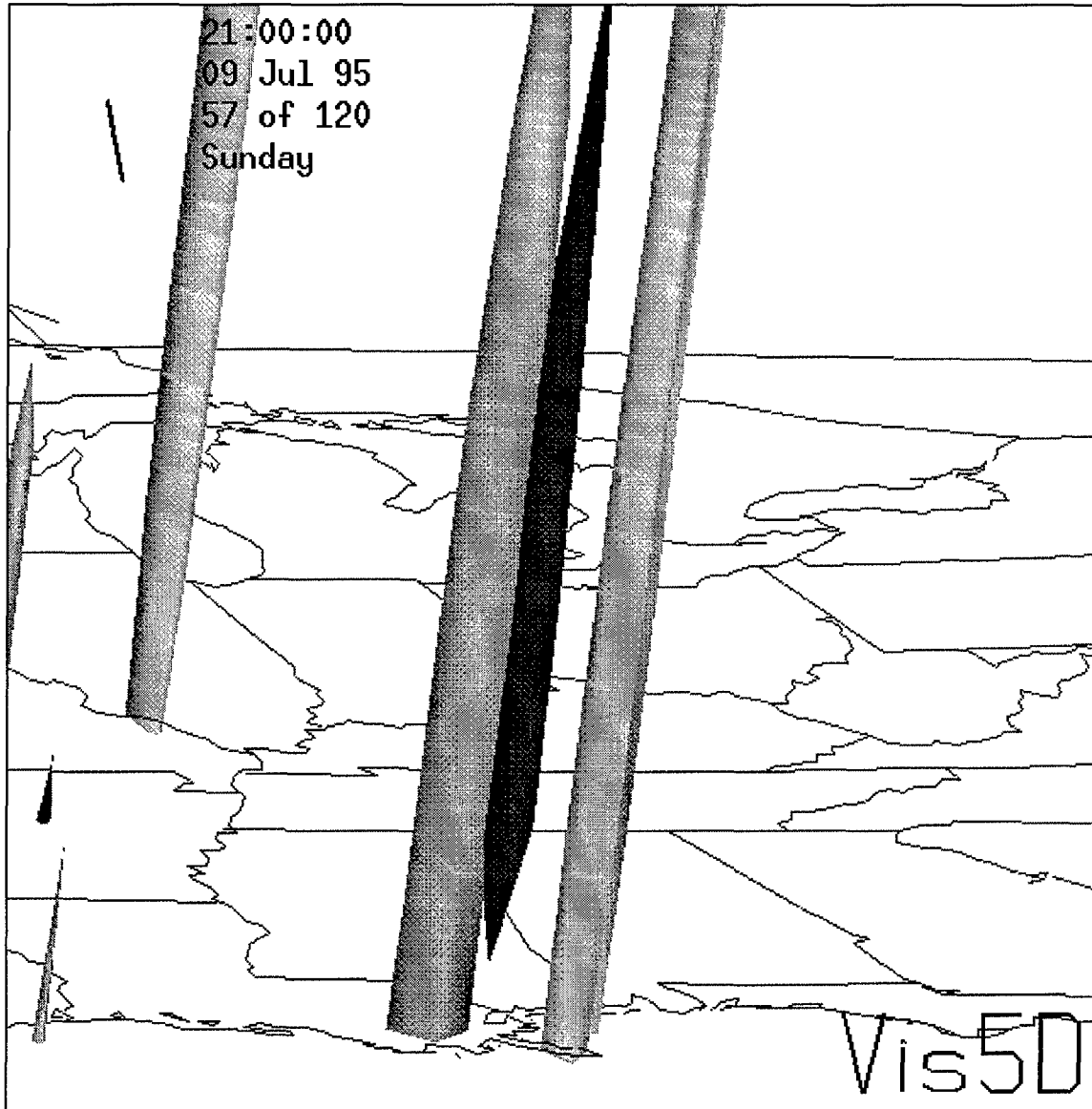


Figure 4.10: Isosurface rendering of -20 ppbv/h diffusion (dark gray surface) and 0.12 ppmv ozone concentration (light gray surface) for the Baton Rouge, LA case

aloft, the horizontal advection became the dominant process and spread the ozone eastward before chemical reactions removed it after dark.

The three-dimensional volume imaging gave little additional information since the concentration and process analysis values were consistent from the bottom layer to the top.

4.3 Atlanta, Georgia, July 10 - 11, 1995 (54 km Resolution)

In this case, the ozone concentration exceeded 0.12 ppmv both from 10/1600 - 11/1100 UTC (10/1200 - 11/0700 Eastern Daylight Time (EDT)) during which it reached a maximum concentration of 0.197 ppmv, and from 11/1500 - 12/1000 UTC (11/1100 - 12/0600 EDT) during which it reached a maximum concentration of 0.170 ppmv. The area of ozone accumulation was actually southwest of Atlanta (30,28) at (31,27) in Figure 4.11. The winds were very weak northeasterlies on both days. The precursors appear to have moved with the light winds early on the 10th; however, due to the very low winds, the author labeled this case a stagnant air event.

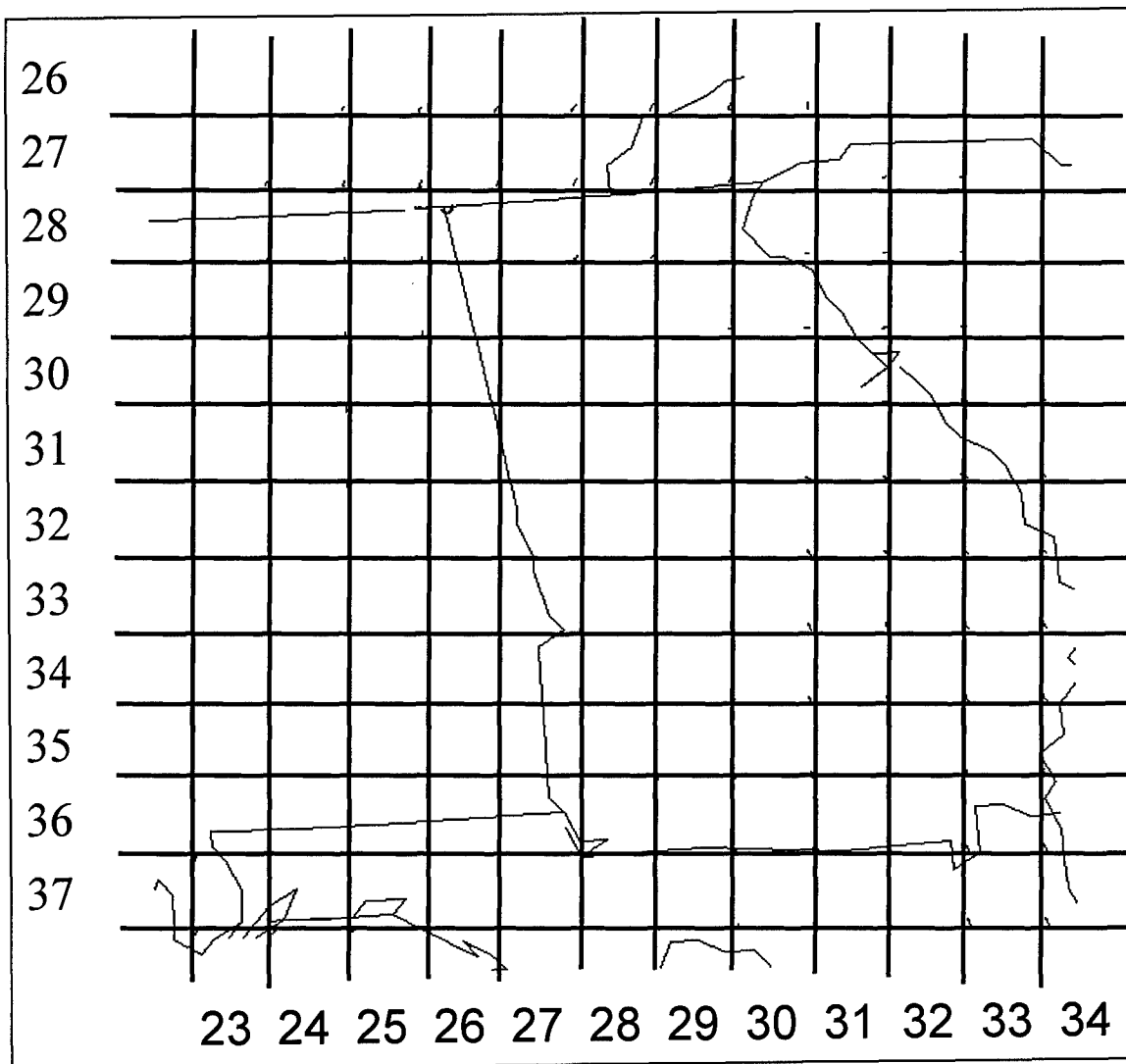
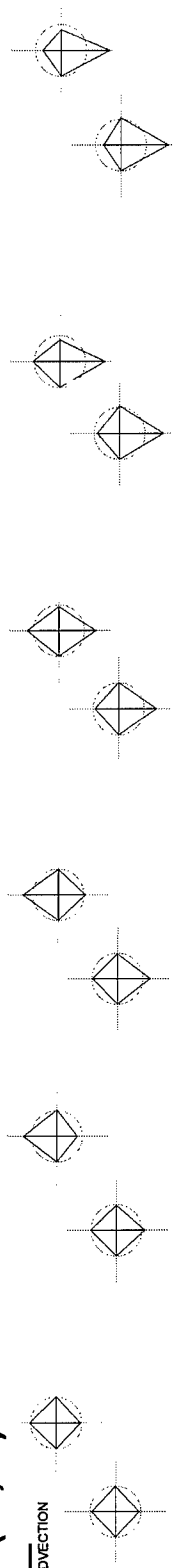
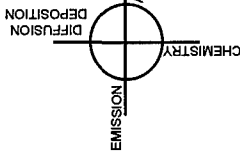


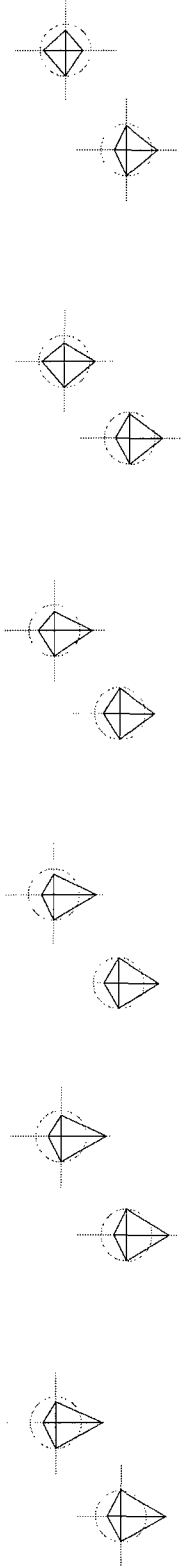
Figure 4.11: Map of cell reference grid over Atlanta, GA (30,28)

When the author compared the small multiples for the 20 m level at cell (30,28) and (31,27) in Figures 4.12 and 4.13, he found that chemistry was very consistent through the daylight hours in this case, and advection made the difference in the concentration of ozone over Atlanta (30,28) on the 10th. Comparing the 10th to the 11th, diffusion and deposition made much more of

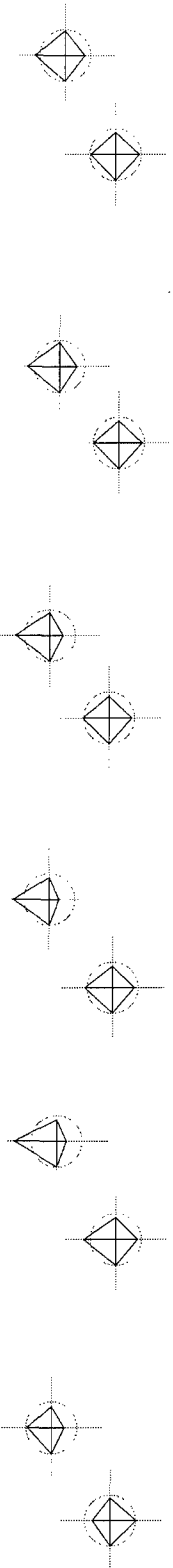
(30,28)
(31,27)



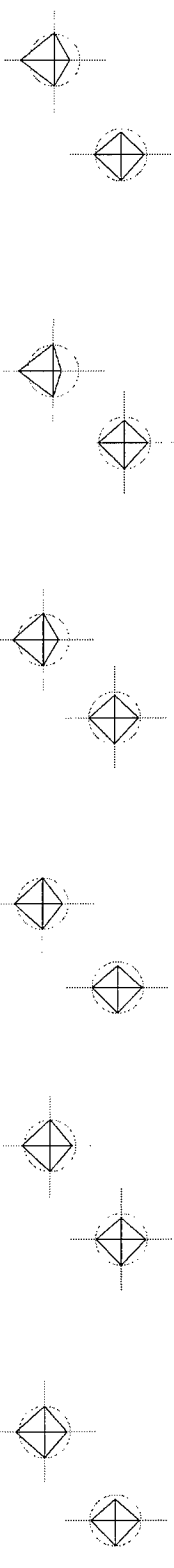
10/1200 UTC 10/1300 UTC 10/1400 UTC 10/1500 UTC 10/1600 UTC 10/1700 UTC



10/1800 UTC 10/1900 UTC 10/2000 UTC 10/2100 UTC 10/2200 UTC 10/2300 UTC



11/0000 UTC 11/0100 UTC 11/0200 UTC 11/0300 UTC 11/0400 UTC 11/0500 UTC

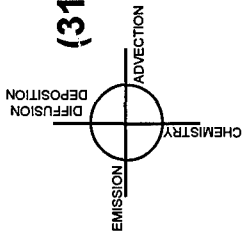


11/0600 UTC 11/0700 UTC 11/0800 UTC 11/0900 UTC 11/1000 UTC 11/1100 UTC

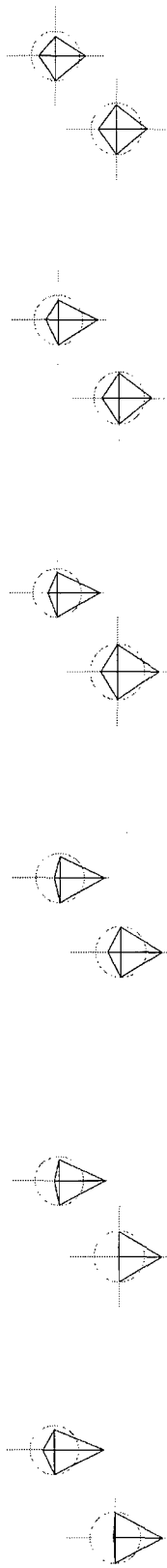
Figure 4.12: Small multiples for 10/12 - 11/11, Atlanta, GA case. Radial Scale is -50 to +50 ppbv/h.

(30,28)

(31,27)

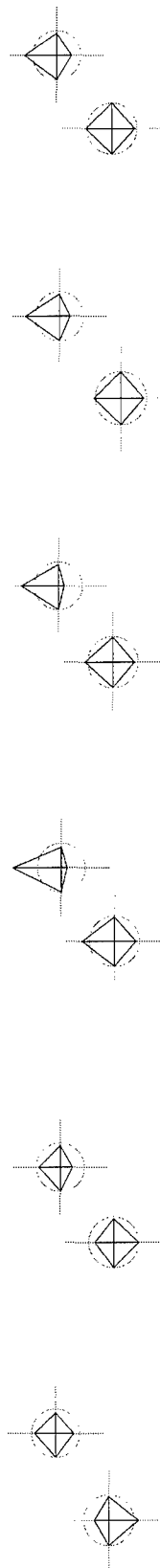


11/1200 UTC 11/1300 UTC 11/1400 UTC 11/1500 UTC 11/1600 UTC

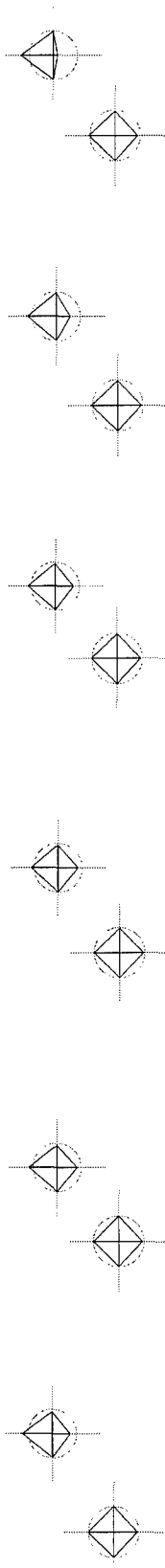


50

11/1700 UTC 11/1800 UTC 11/1900 UTC 11/2000 UTC 11/2100 UTC 11/2200 UTC



11/2300 UTC 12/0000 UTC 12/0100 UTC 12/0200 UTC 12/0300 UTC 12/0400 UTC



12/0500 UTC 12/0600 UTC 12/0700 UTC 12/0800 UTC 12/0900 UTC 12/1000 UTC

Figure 4.13: Small multiples for 11/12 - 12/10, Atlanta, GA case. Radial Scale is -50 to +50 ppbv/h.

a contribution on the 11th. Considering this basic information, the author moved into individual process analysis using planar slices.

On the 9th and the 10th, the air was very stagnant over eastern Alabama and western Georgia. The lack of motion most likely allowed precursors to accumulate in the border region between the two states, southwest of Atlanta. On the 10th, chemical production was very consistent in its rates from 1600 - 2100 at (31,27) and southwest, in this border region and over Atlanta itself. Interestingly enough, on the 11th, when the maximum ozone concentration actually was lower than that of the 10th, the rates of production were higher over the entire region. Chemical process rates were 40-50 ppmv/h on the 10th as in Figure 4.14, and were in the 50s on the 11th as in Figure 4.15.

Advection of ozone, when there was any, tended to suffer the same fate as in the Baton Rouge case, vertical and horizontal advection canceled each other out. However, there was one exception to this situation, and that was on the 10th, directly over Atlanta. Advection was moderate at the lower three levels and slightly stronger in the upper three from 1600 to 0100. These contributions were the primary reason for the low concentrations over the city.

Diffusion and deposition were consistent between (30,28) and (31,27) during ozone production hours on the 10th. At the bottom layer, the

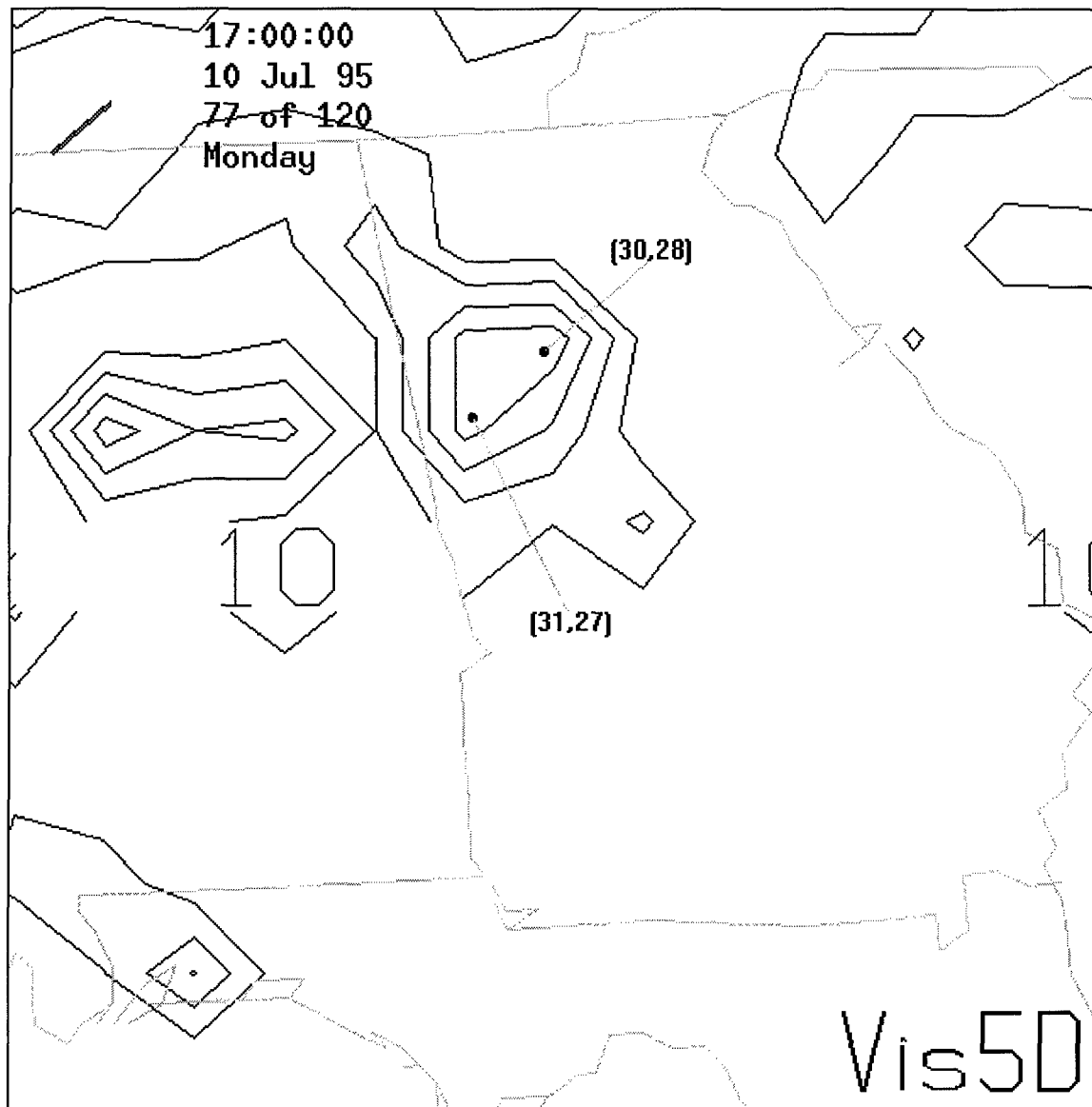


Figure 4.14: Planar slice of chemical process contours for the Atlanta, GA case on July 10th

processes tended to work against each other resulting in some removal, but insufficient to affect the high concentration produced by the chemical processes earlier in the day. Aloft, diffusion was similarly ineffective against the high concentration. On the 11th, deposition was much stronger in the

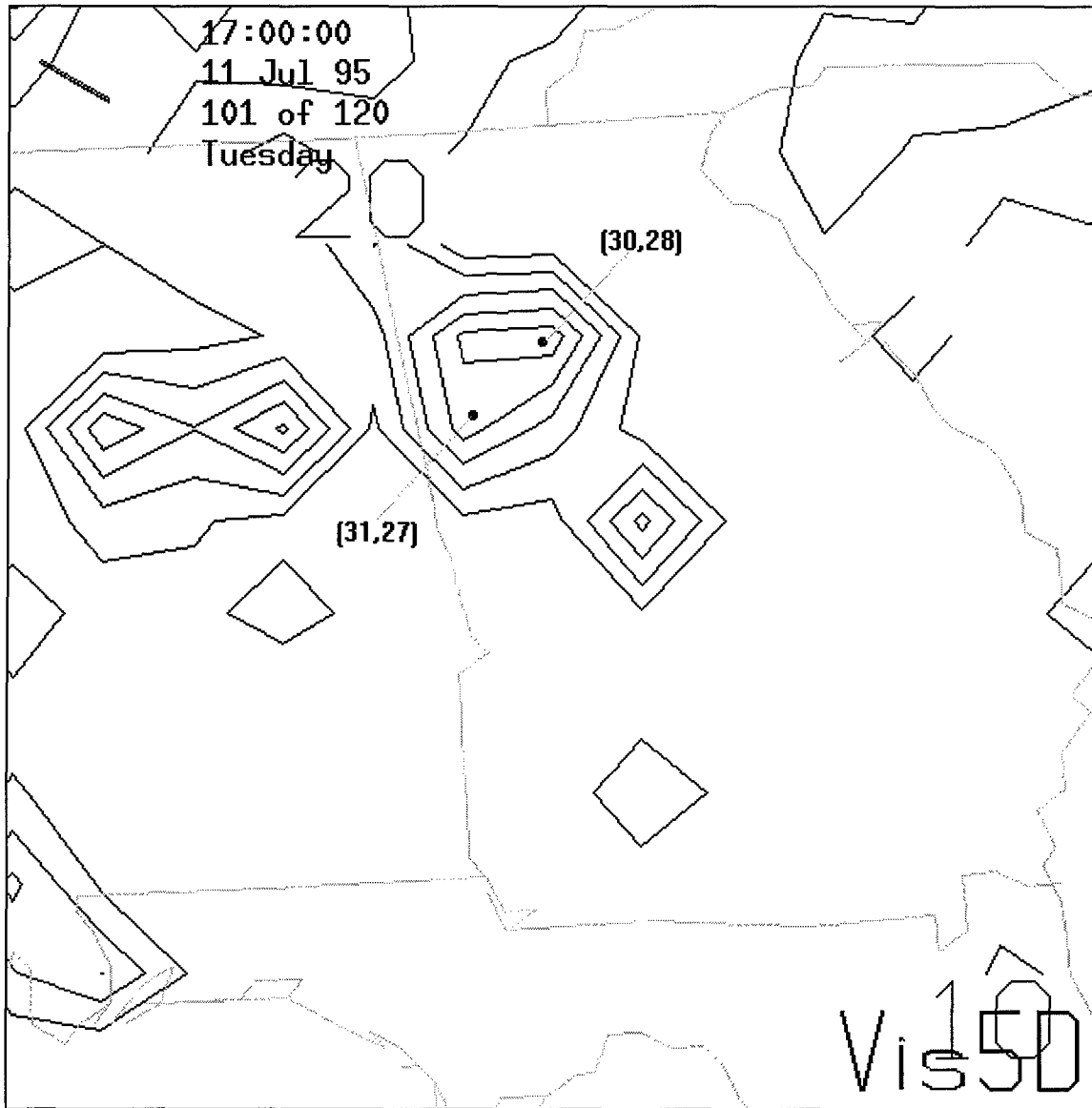


Figure 4.15: Planar slice of chemical process contours for the Atlanta, GA case on July 11th

bottom layer, as was diffusion aloft. These processes were the reason for the lower concentrations on the 11th, even though the chemical processes were stronger than the day before. In fact, at 1700 on the 11th, the removal of

ozone by diffusion increased, causing a temporary drop in ozone concentration at (31,27).

The three-dimensional isosurfacing and volume imaging gave little additional information since the concentration and process analysis values in the layers were mostly consistent from the bottom layers to the top.

4.4 St Louis, Missouri, July 12, 1995 (36 km Resolution)

In this case, the ozone concentration exceeded 0.12 ppmv from 12/1800 - 13/0100 UTC (12/1300 - 2000 CDT) and reached a maximum concentration of 0.168 ppmv. The area of ozone accumulation was actually north of St Louis (30,27) at (29,27). See Figure 4.16. The winds were very strong southerlies, on the order of 4 - 6 m/s. The flow was steady throughout the whole case, and since the accumulation occurred to the north and spread further north, the author labeled this case a downwind event.

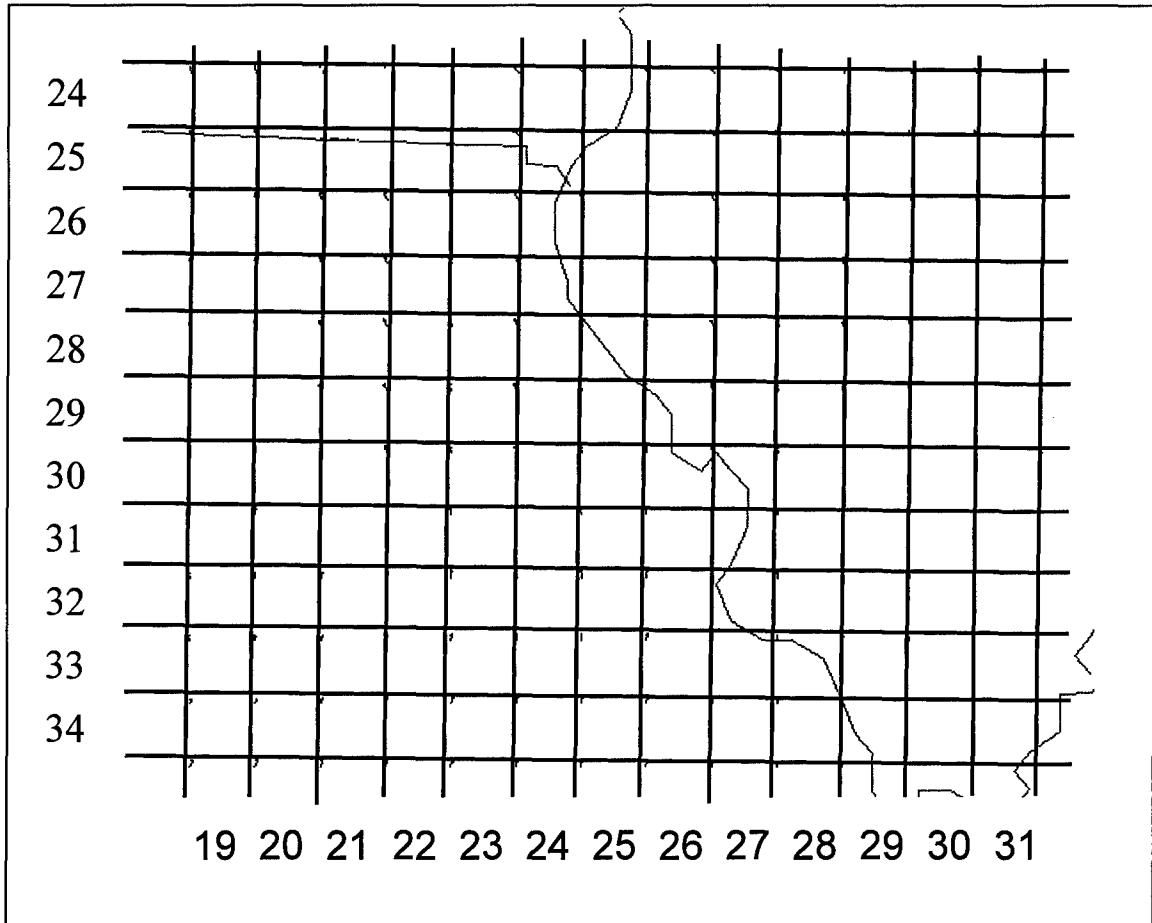


Figure 4.16: Map of cell reference grid over St Louis, MO (30,27)

When the author compared the small multiples for the surface at cell (30,27) and (29,27) in Figure 4.17, he found that chemistry was much stronger in (30,27) over the city, while the concentrations were higher over (29,27). Advection was a powerful removal process over the city while it was nearly zero (0) over (29,27), and removal by diffusion and deposition was slightly stronger in (29,27). Considering this basic information, the author moved into individual process analysis using planar slices.

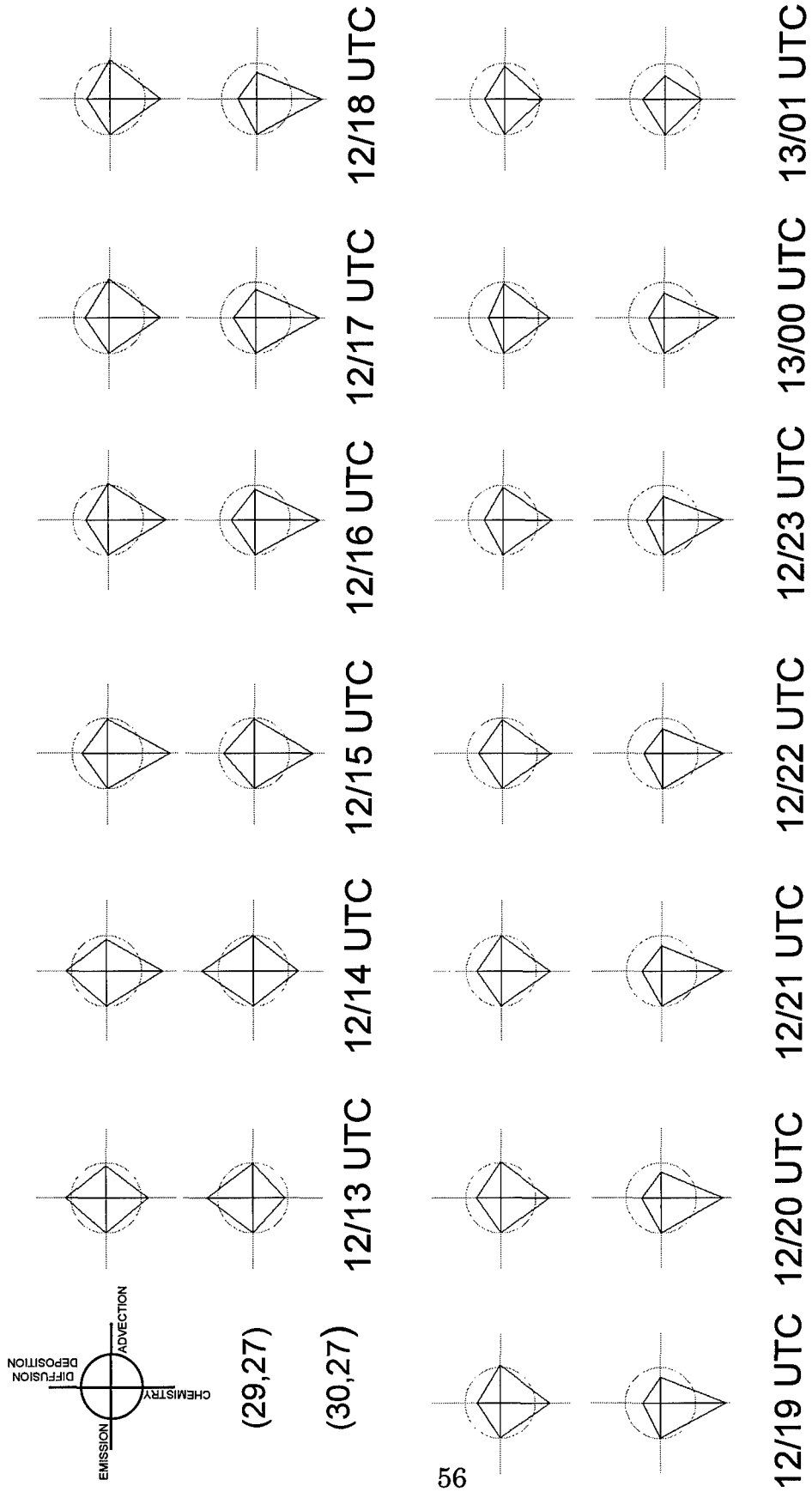


Figure 4.17: Small multiples for St Louis, MO case. Radial Scale is -50 to +50 ppbv/h.

The strong southerly winds most likely advected the early morning precursors north of the city to (29,27) where the earliest chemical production begins by 1300. By 1600, ozone production was more centered directly over St Louis. The 10 ppmv/h interval contours in Figure 4.18 enclose a

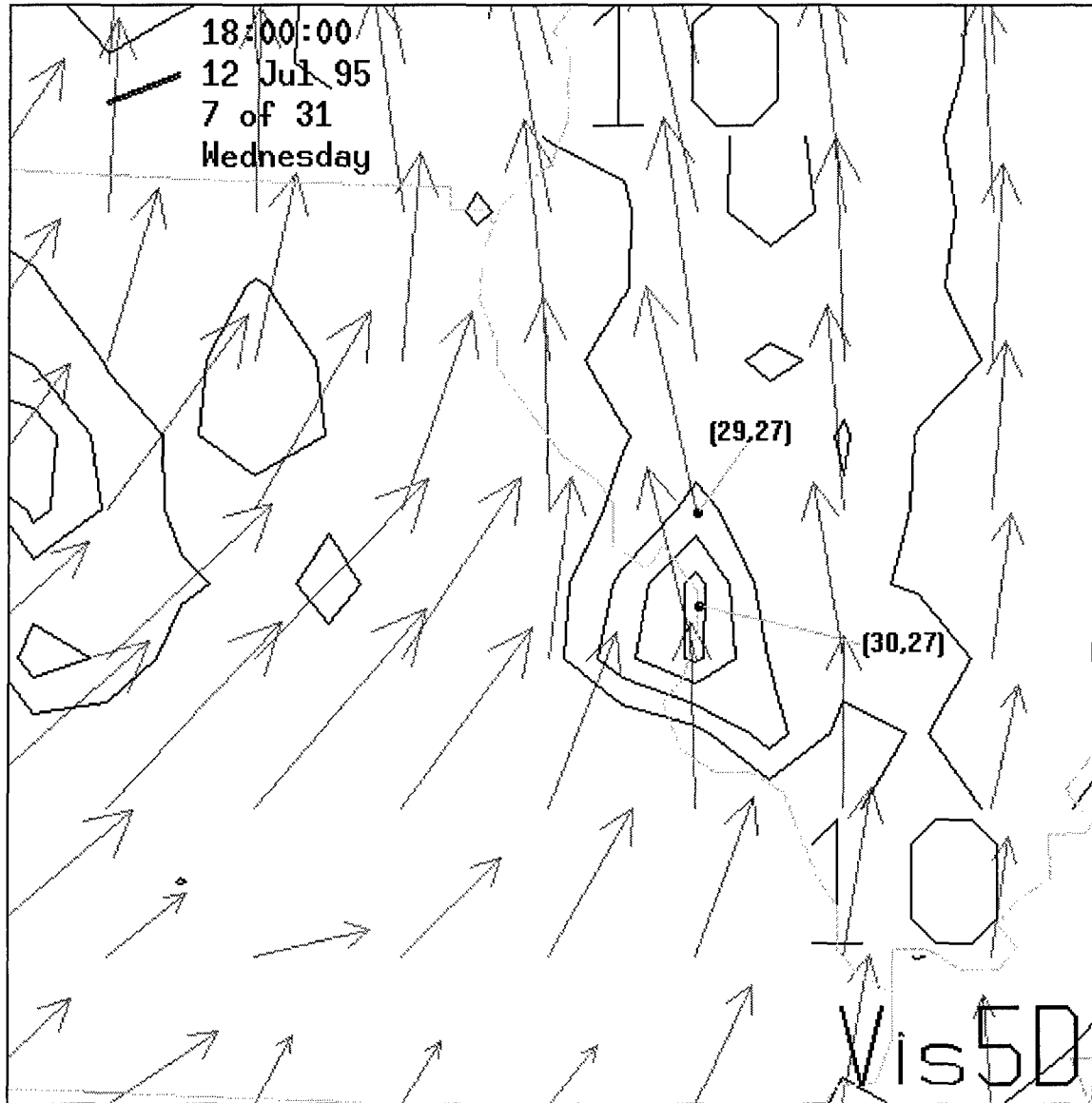


Figure 4.18: Planar slice of chemical process contours with wind vectors for the St Louis, MO case

40-50 ppmv/h cell over St Louis with only 20-30 ppmv/h north of the city. Interestingly, chemical production continued through 0000, but the night time chemical removal processes made little to no contribution to the reduction of ozone.

Because of the strong winds in this case, the horizontal advection was a dominant player. After 1700, the winds were advecting one quarter to one third of all of the ozone produced by the chemical process over St Louis out of the city. In the evening, when the chemical production reduced, the ozone pool began traveling along the streamlines, primarily aloft. In fact, the author found the ozone pool between the areas of high negative and positive advection through the night. The ozone concentration stayed high, though it moved up to the higher layers and along the streamlines, settling over Chicago the next day. Figure 4.19 represents the ozone concentration as an isosurface enclosing concentrations greater than 0.12 ppmv, and represents positive advection, implying accumulation, as dark contours of 5 ppmv/h and negative advection, implying removal, as light ones. Figure 4.19 reveals the location of the ozone pool in relation to the advection cores.

Diffusion and deposition had important roles to play in this case as well. Deposition was very strong over St Louis. Even though diffusion was adding ozone to the bottom layer's cell, this process was only compensating for 50 to 75 percent of the ozone removed by deposition. Deposition was also

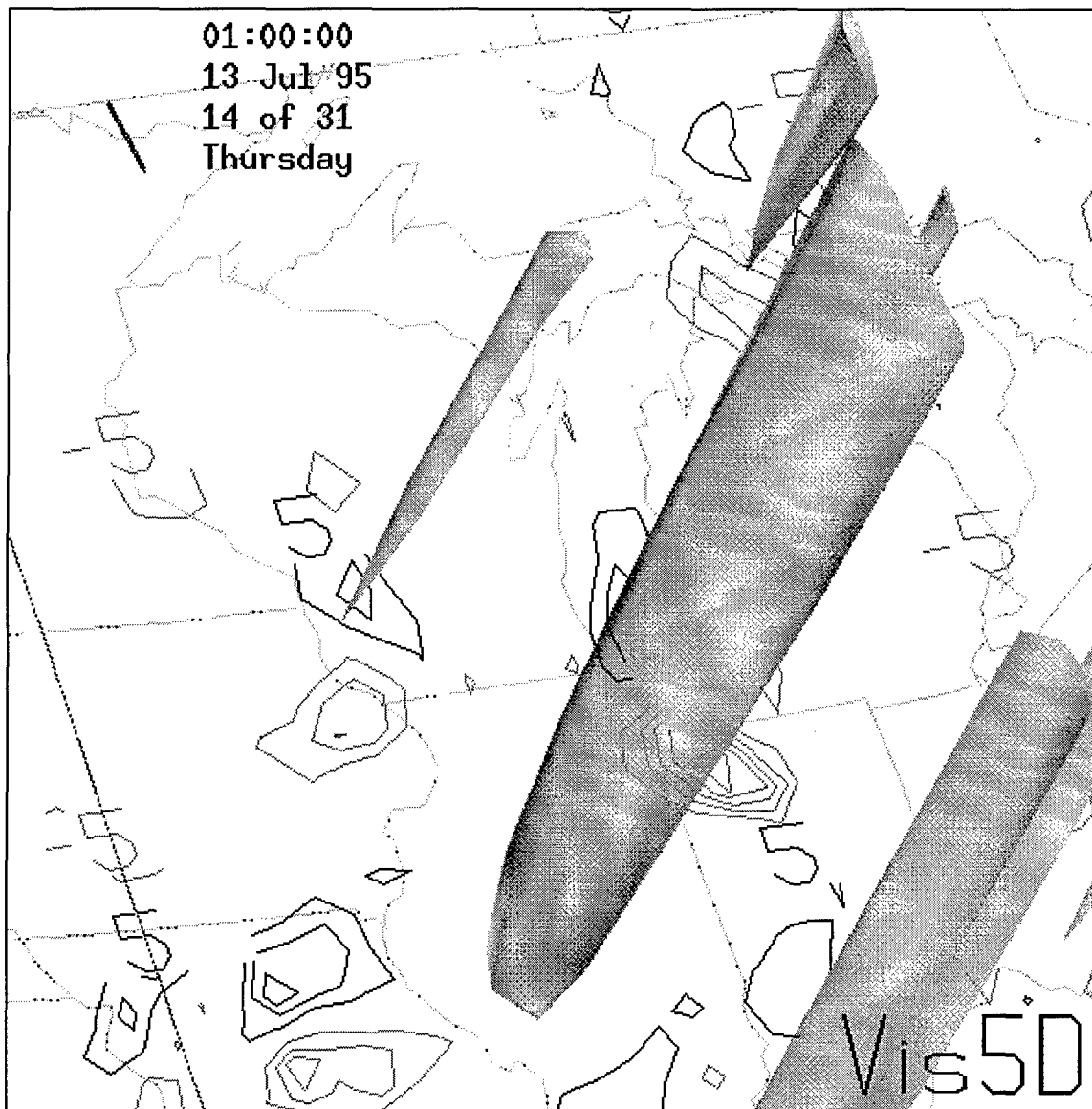


Figure 4.19: Isosurface rendering of 0.12 ppmv ozone concentration and total advection process contours (light gray are negative and dark gray are positive) for the St Louis, MO case

strong at (29,27); however, the diffusion compensated for nearly all of the deposition, allowing the ozone to accumulate north of the city while chemical production was strong over the city. Aloft, diffusion was weak, removing only

small amounts of the ozone. Removal by diffusion was slightly stronger directly over St Louis, accounting for the lower ozone concentration levels there.

The three-dimensional volume imaging gave little additional information since the concentration and process analysis values in the layers were mostly consistent from the bottom to the top of the study area.

4.5 Washington, DC, July 12, 1995 (36 km Resolution)

In this case, the ozone concentration exceeded 0.12 ppmv from 12/1600 - 13/0100 UTC (12/1200 - 2100 EDT) and reached a maximum concentration of 0.178 ppmv. The area of ozone accumulation was actually southeast of Washington (27,58) at (28,59), as seen in Figure 4.20. The winds were very weak throughout the day. They were less than 2 m/s until 1800 and did not exceed 4 m/s until after 0500, and that was only to the east of the city. The weak winds had a northerly component early and became southerly by mid-afternoon. Because of the lack of motion, the author labeled this case a stagnant air event.

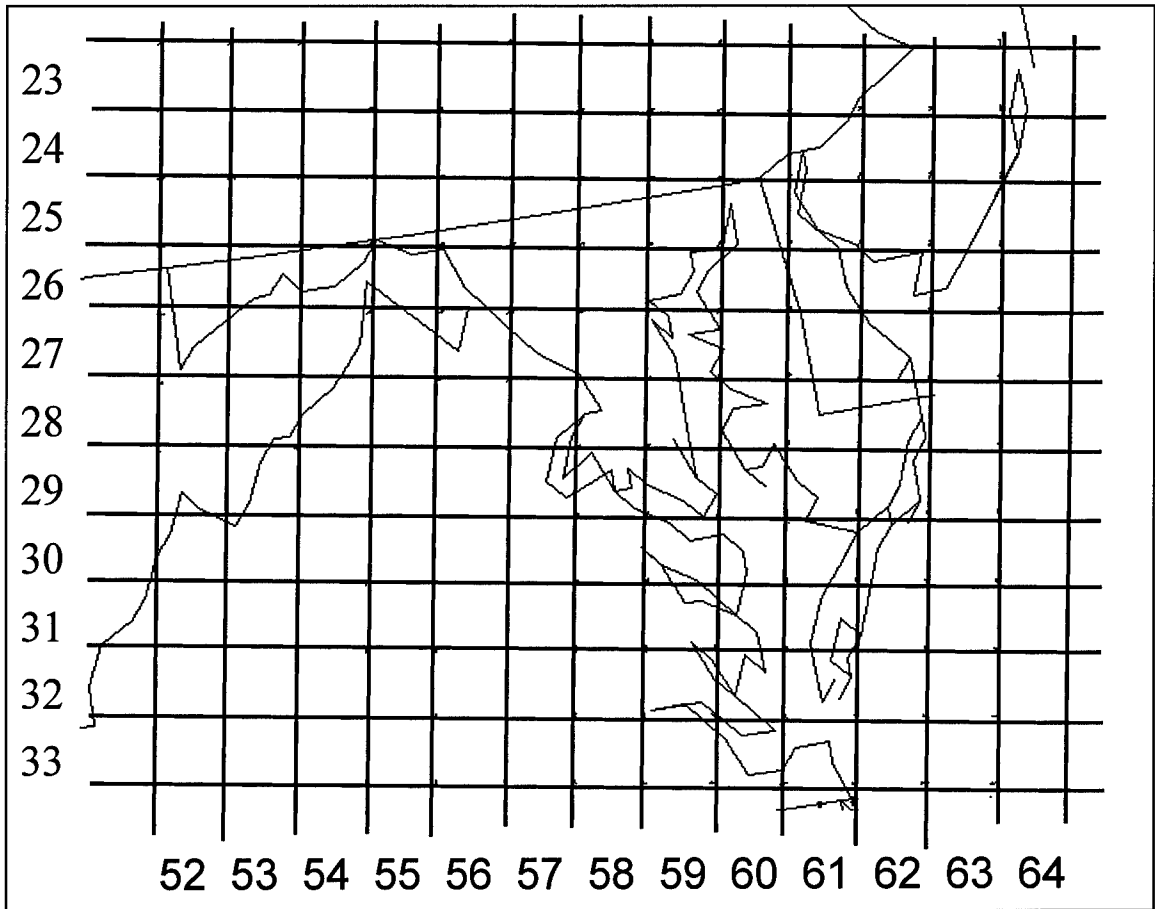


Figure 4.20: Map of cell reference grid over Washington, DC (27,58)

When the author compared the small multiples for the surface at cell (27,58) and (28,59) in Figure 4.21, he found that chemical production was very strong in (28,59) where the concentrations were higher. Over the city, production rates were very low and even negative during mid-day. Advection was of little consequence until late, and removal by diffusion and deposition was stronger in (28,59). Considering this basic information, the author moved into individual process analysis using planar slices.

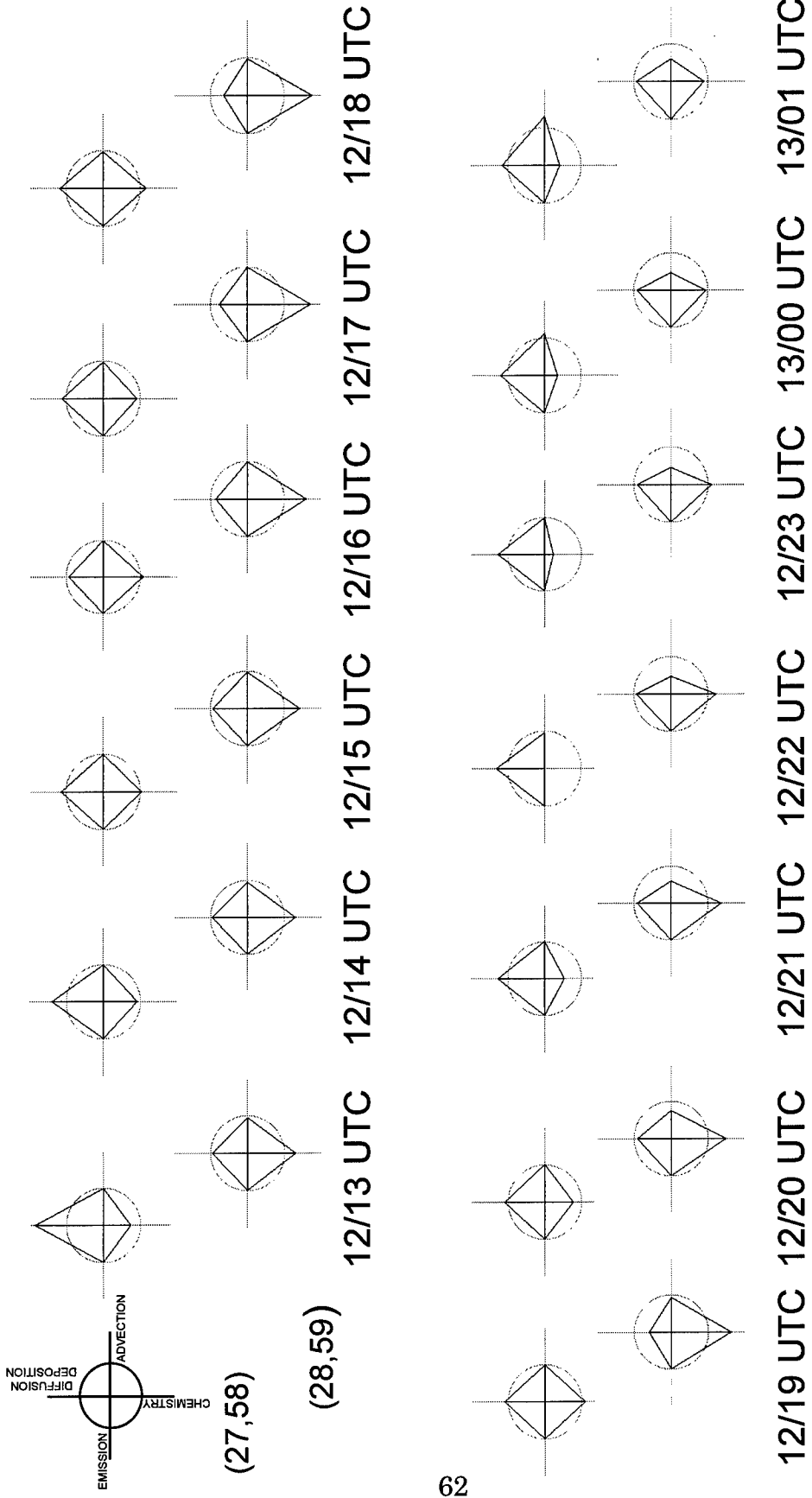


Figure 4.21: Small multiples for Washington, DC case. Radial Scale is -50 to +50 ppbv/h.

The vertical column over Washington (27,58) appeared to be an ozone sink. The chemical production was very low or negative all day long. The primary production was over (28,59) where the high ozone occurred. Notice in Figure 4.22 the dark gray contours marking chemical production at

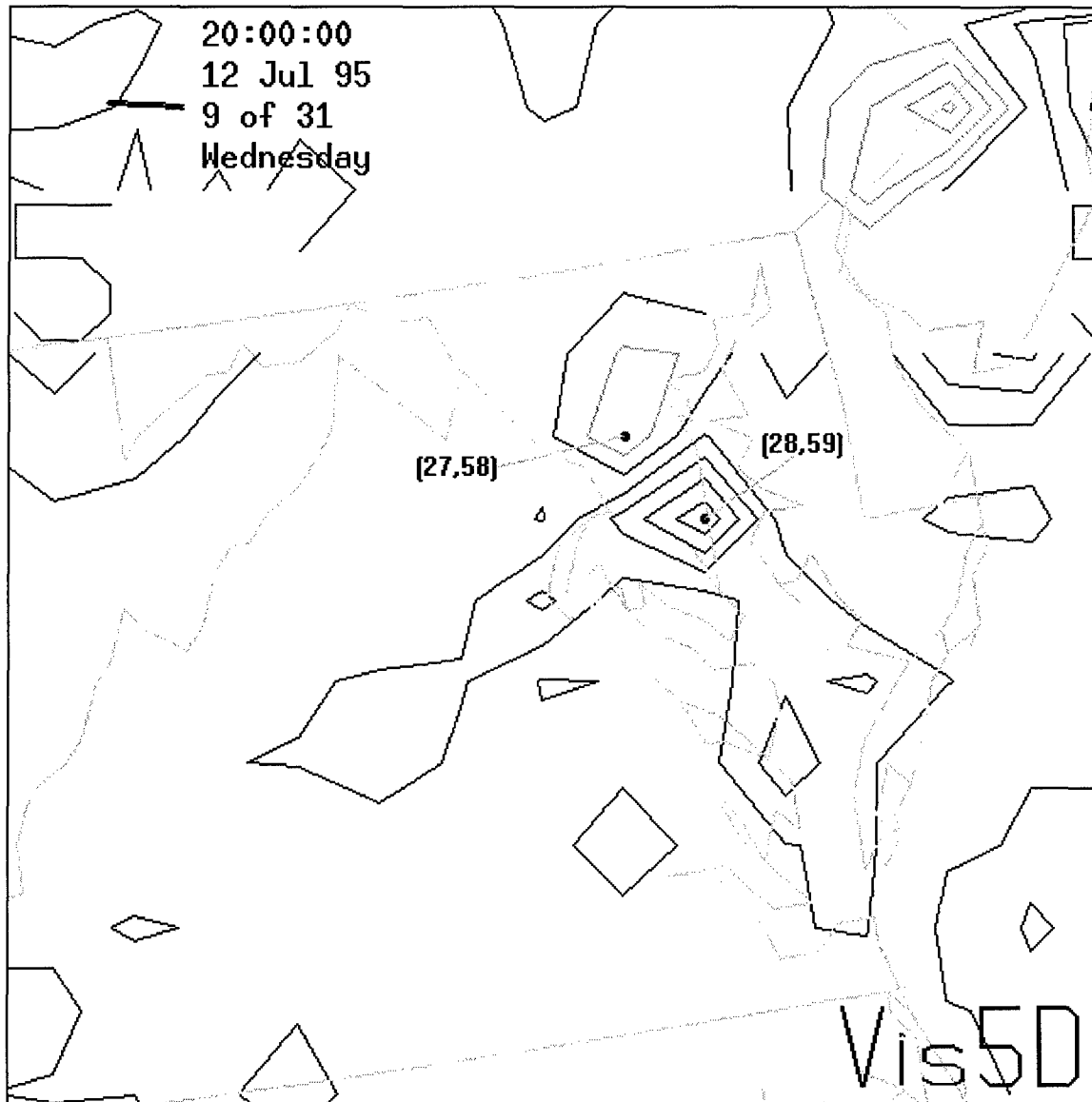


Figure 4.22: Planar slice of chemical process contours (light gray are negative and dark gray are positive) for the Washington, DC case

(29,58), while the light gray surround the city indicating chemical removal at 1600 EDT. Aloft, even this production rate was greatly reduced around the 4th sigma level. In the two upper levels, production was present, but more spread out over the cells around southeastern Washington. This production caused a more widespread pool of ozone aloft. A volume imaging of the chemical processes is in Figure 4.23. It is a perspective shot of the volume looking down from a point in space located a distance along a line which intersects the surface plane at about 30 degrees to the normal. The perspective makes the vertical columns appear to plot north-northwestward from the bottom layer, while in actuality, they are directly overhead of the bottom layer. The column which rose above (28,59) shows the embedded ozone depletion cell around the 140 m point indicated by the bright green core surrounded by the production zones indicated by the red tones over and around the Washington area.

In general, the advection was a weak player in the formative hours of this case. When the winds developed their southerly component after 1900, advection began spreading the ozone north along the eastern side of Washington from (28,59).

Diffusion and deposition made little contribution to this case, and the three-dimensional isosurfacing gave little more information than the volume imaging or planar slices gave.

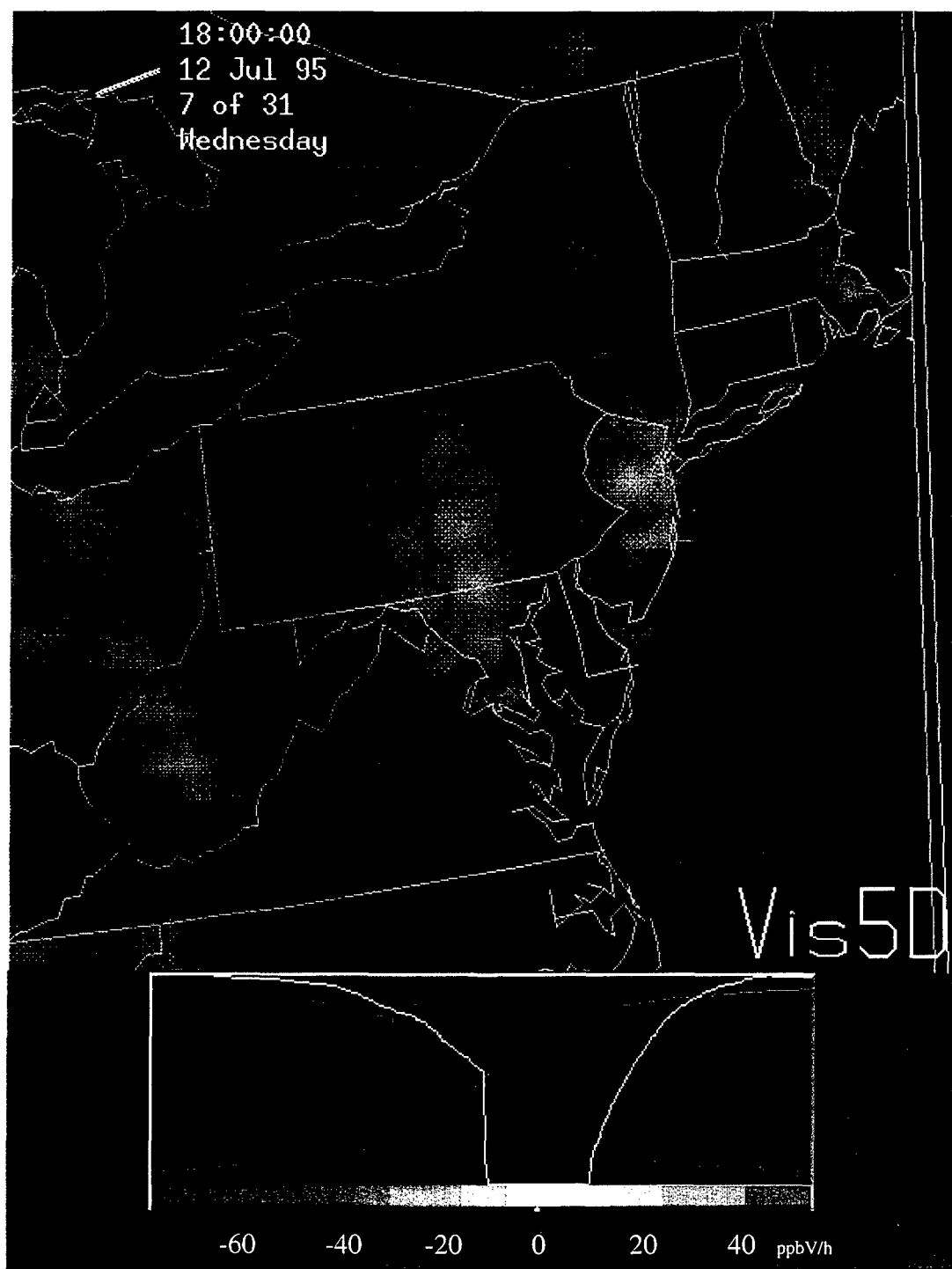


Figure 4.23: Volume imaging of chemical process rates for Washington, DC case

4.6 Comparison of Cases Using Small Multiple Plotting Map

In Figure 4.24, the author compared the shapes of the small multiples over the cells where ozone accumulated to its maximum concentration in the Houston and Baton Rouge cases. The shapes were quite similar, nearly touching the zero circle at all but the lower, chemical process point. These plots did indicate a slight removal by advection, though its contribution is not significant to this conclusion. This elongated diamond shape seemed to be fairly indicative of normal accumulation due to dominant chemical processes.

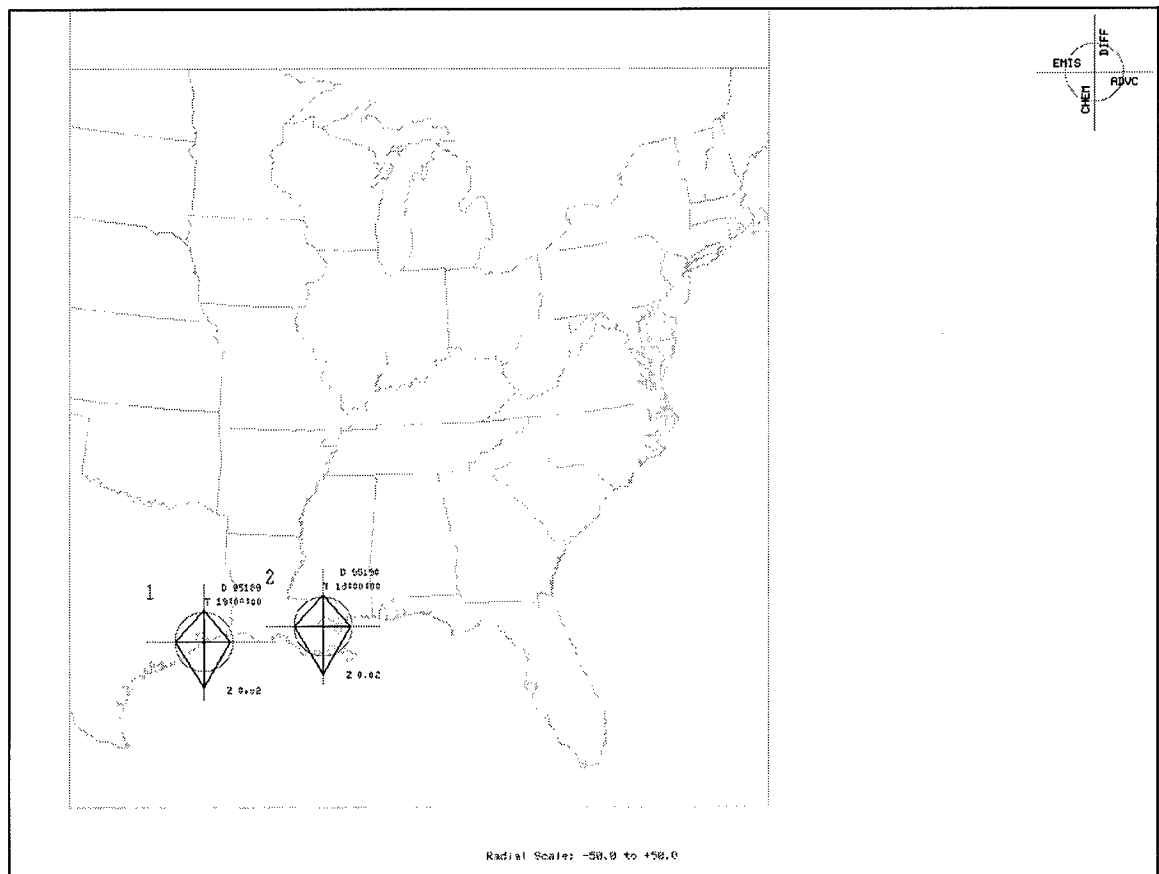


Figure 4.24: Small multiple plotting map comparing similar shapes near Houston, TX, and Baton Rouge, LA

In Figure 4.25, the author compared the shapes of the small multiples over the cells where the ozone did not accumulate despite high chemical production rates for the same case as Figure 4.24. The shapes were, once again, quite similar, showing a squat top and elongated bottom, while the emission and advection rates were near the zero circle. Once again, these plots did indicate a slight removal by advection, though its contribution is not significant to this conclusion. This short kite shape seemed to be fairly indicative of significant removal by diffusion and deposition processes.

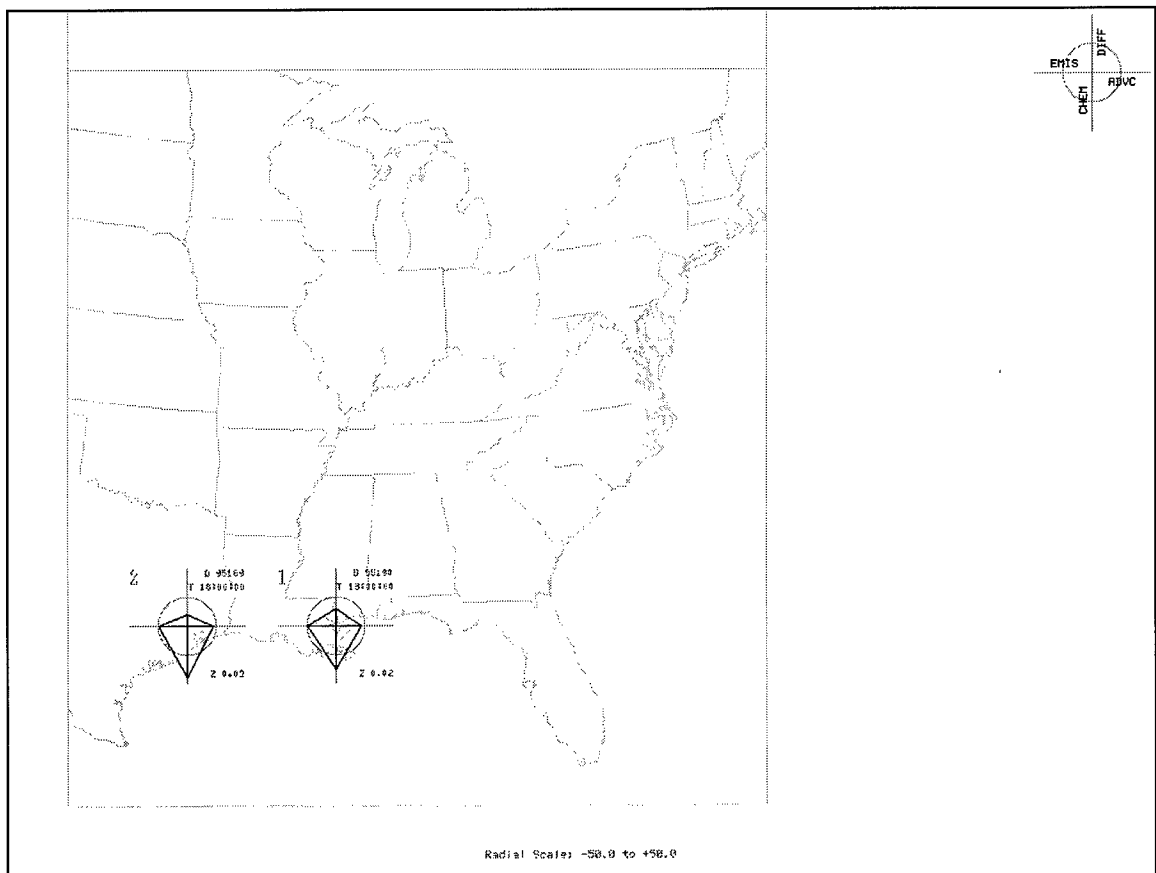


Figure 4.25 Small multiple plotting map comparing similar shapes near Houston, TX, and Baton Rouge, LA

Finally, in Figure 4.26, the author compared the shapes of the small multiples over the city of St Louis where the ozone did not accumulate and the cell southeast of Washington where it did accumulate. In both cases, chemical production was high, and diffusion and deposition were strong ozone removers. In fact, the Washington plot is similar to the short kite shape from Figure 4.25; however, ozone accumulated significantly in Washington while it did not do so in Houston or Baton Rouge. Returning to the time series plots in Figures 4.4, 4.8, and 4.21, Houston case cell (38,9)

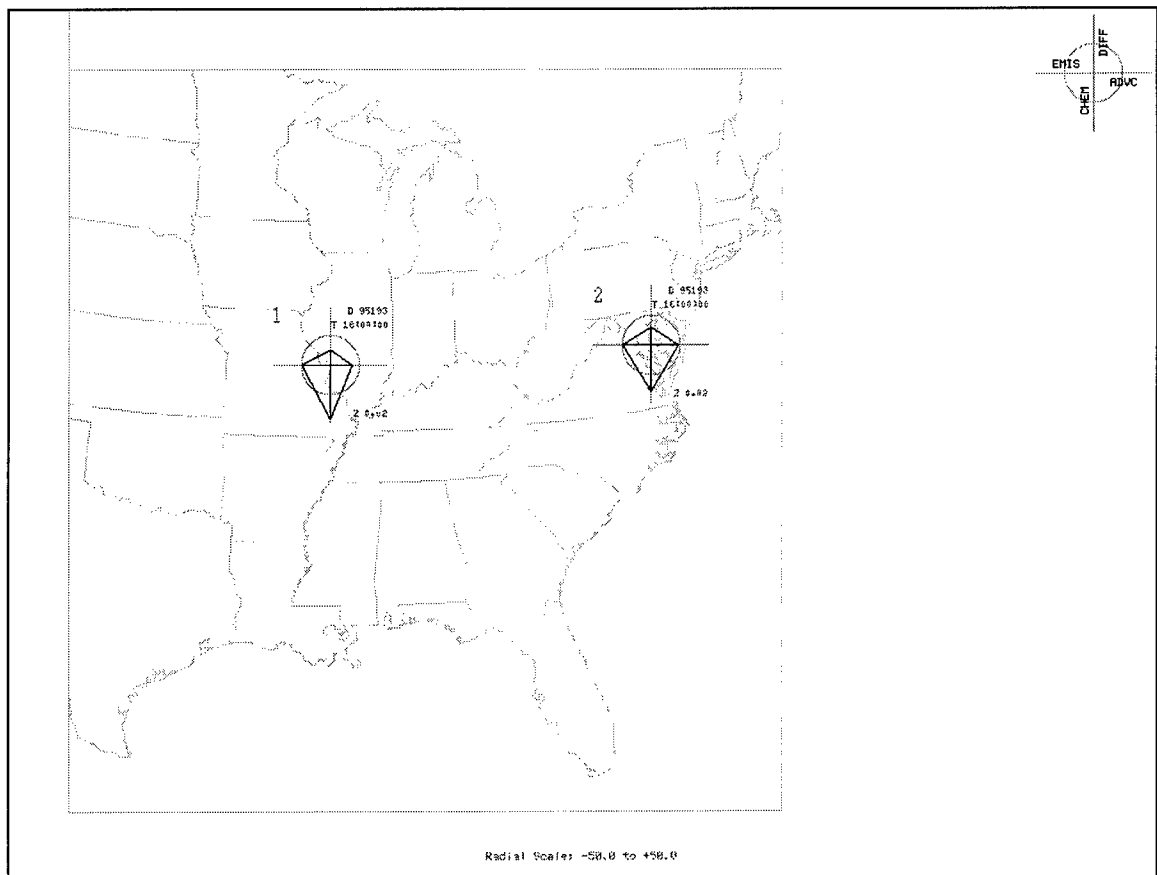


Figure 4.26: Small multiple plotting map comparing dissimilar shapes near St Louis, MO, and Washington, DC

had a much shorter diffusion/deposition leg earlier in the day, and Baton Rouge case cell (38,19) had a short advection leg earlier in the day. These stronger removal processes kept their ozone concentrations lower; while Washington, lacking the removal processes early, reached a higher concentration. The plot over St Louis has a short advection leg due to the high winds during that case. So, this clipped kite shape suggests a high wind event producing strong negative advection and diffusion processes to compensate for the high chemical production rates.

5 CONCLUSIONS

The researchers have evaluated several different procedures for visualizing the contributions to ozone concentration. The variation on the Tiny Cubes method for plotting the data discretely was a valuable tool for identifying points in the atmosphere where maxima in increases and decreases in ozone concentration due to individual processes occur. The author did not find these useful in the case studies because the maxima and minima were consistent though vertical columns. Generally, they simply repeated the same information available through a planar slice, which showed which vertical column had the maximum/minimum for that process.

The planar slices were most useful for positioning the Vis-5D cursor to use in creating the small multiples. Because of the uniformity in the vertical columns, planar slices helped point out the places to examine (the areas of maxima and minima) with the probe as easily as the Tiny Cubes variation. In addition, since positioning the cursor accurately in a virtual three-dimensional display was very difficult with a two-dimensional mouse, the two-dimensional slice made this task much simpler.

The isosurfacing method gave insight into the large volumes which included the maxima. The author was able to use isosurfacing primarily to track the motion of pools of ozone. Isosurfacing was also useful for seeing how certain processes had strong influence in one cell versus another process in a neighboring cell, such as in the Baton Rouge case. It was only useful, however, when the absolute value of the magnitude of the process was high enough so that the computer only displayed one or two isosurfaces.

Volume rendering could give the best look into the three-dimensional structure of the volume-wide contributions; however, practical application of this method revealed little new information due to the uniformity of the concentrations and the magnitude of the process rates through the lower 300 m of the atmosphere. Perhaps this technique would be more useful in a study using more than six sigma layers covering more than just the lowest 300 m of the troposphere.

Small multiples made comparing the relative contributions of the different processes, as well as the relative importance of processes between different cells, an easy task. They quickly identified which processes the author needed to study more intensely. The weakness in the researchers' design was in the combination of the horizontal and vertical advection, as evidenced in the Baton Rouge case. In that case, the small multiples tended to show little contribution due to total advection; however, there was a significant contribution of opposite sign from the components of that small multiple.

The author was not able to use the integrated reaction rate and mass balance routine the researchers developed since it was dependent upon a post-processing program which was not available at publication.

Overall, none of these tools is sufficient by itself. However, the researchers believe that using these tools together will help air quality modelers and users understand what physical and chemical processes are driving the models, and where to look for changes which may improve air quality.

6 LIST OF REFERENCES

- ¹ Finlayson-Pitts, B.J. and Pitts, J.N., Jr. (1986) *Atmospheric Chemistry: Fundamentals and Experimental Techniques*. John Wiley & Sons, Inc., New York.
- ² Jang, J.C., Jeffries, H.E., Byun, D., et al. (1995) Sensitivity of Ozone to Model Grid Resolution — I. Application of High-Resolution regional Acid Deposition Model. *Atmospheric Environment*. 29 (21), 3085-3100.
- ³ Finlayson-Pitts, B.J. and Pitts, J.N., Jr. (1993) Atmospheric Chemistry of Tropospheric Ozone Formation: Scientific and Regulatory Implications. *Journal of the Air and Waste Management Association*. 43, 1091-1100.
- ⁴ Hibbard, W.L., Paul, B.E., Santek, D.A., et al. (1994) Interactive Visualization of Earth and Space Science Computations. *Computer*. 27 (July 1994), 65-72.
- ⁵ Jeffries, H.E. and Tonnesen, S. (1994) A Comparison of Two Photochemical Reaction Mechanisms Using Mass Balance and Process Analysis. *Atmospheric Environment*. 28 (18), 2991-3003.
- ⁶ Fine, S.S. and Mathur, R. (1996) Applying Multivariate Analysis Techniques to Interpreting Results Produced by Chemistry-Transport Models. To appear in *Proceedings, Computing in Environmental Resource Management*. Air & Waste Management Association, Research Triangle Park.
- ⁷ Odman, M.T. and Ingram, C.L. (1996) *Multiscale Air Quality Simulation Platform (MAQSIP): Source Code Documentation and Validation*, ENV-96TR002-v1.0. Environmental Programs, MCNC-North Carolina Supercomputing Center, Research Triangle Park.
- ⁸ Dennis, R.L., Byun, D.W., Novak, J.H., et al. (1996) The Next Generation of Integrated Air Quality Modeling: EPA's Models-3. *Atmospheric Environment*. 30 (12), 1925-1938.
- ⁹ Aneja, V. P. and Li, Z. (1992) Characterization of Ozone at High Elevation in the Eastern United States: Trends, Seasonal Variations, and Exposure. *Journal of Geophysical Research*. 97 (D9), 9873-9888.
- ¹⁰ Warneck, P. (1988) *Chemistry of the Natural Atmosphere*. Academic Press, San Diego.
- ¹¹ Jang, J.C., Jeffries, H.E., and Tonnesen, S. (1995) Sensitivity of Ozone to Model Grid Resolution — II. Detailed Process Analysis for Ozone Chemistry. *Atmospheric Environment*. 29 (21), 3101-3114.
- ¹² Tufte, E.R. (1990) *Envisioning Information*. Graphics Press, Cheshire, Connecticut.
- ¹³ Nielson, G.M., Foley, T.A., Hamann, B., et al. (1991) Visualizing and Modeling Scattered Multivariate Data. *IEEE Computer Graphics & Applications*. 11 (May), 47-55.
- ¹⁴ Advanced Visual Systems, Inc. (1993) *AVS User's Guide*. Advanced Visual Systems, Inc., Waltham, MA.

¹⁵ Thorpe, S., Ambrosiano, J., Balat, R., et al. (1996) The Package for Analysis and Visualization of Environmental Data. *Proceedings: Thirty-Eighth Semi-Annual Cray User Group Meeting*, pp 46-50. Cray User Group, Inc., Charlotte, NC.

¹⁶ Hibbard, W. and Santek, D. (1989) Interactivity is the Key. *Chapel Hill Workshop on Volume Visualization: Conference Proceedings*, pp 39-43. Department of Computer Science, University of North Carolina at Chapel Hill.

¹⁷ Hibbard, B. and Santek, D. (1990) The VIS-5D System for Easy Interactive Visualization. *Proceedings of the First IEEE Conference on Visualization: Visualization '90*, pp 28-35. IEEE Computer Society, San Francisco.

¹⁸ Boutell, T. (1995) *gd 1.2 Users Guide*. Quest Protein Database Center, Cold Spring Harbor Labs.

7 APENDICIES

APPENDIX 7.1 SETUP VIS-5D

In this appendix, the author will describe how to set up Vis-5D with the modification he made to include the Process Analysis small multiple and cycle schematic displays. Whenever the author directs modifications to a file in this document, he will include several lines of code in context from the original file and indicate the changes in **bold** print. Unix commands will be set off in <angular brackets>.

1. Required Files:

a. From University of Wisconsin (anonymous ftp from iris.ssec.wisc.edu/pub/vis5d) retrieve the following files:

vis5d-4.2.tar.Z - source code and documentation (5.4MB)

vis5d-data.tar.Z - sample data sets, map files, topography files
(4.7MB)

b. From Boutell.Com (<http://www.boutell.com:80/gd/>) retrieve the following file:

gd1.2.tar.Z - source code and documentation (105KB)

c. The author's program files in the following file:

rickman.tar.Z - source code for modifications (183KB)

2. Extract the files from **rickman.tar.Z**.

- a. Change to, and place **rickman.tar.Z** in, a working directory (referred to as *working*).
 - b. Uncompress it (<uncompress rickman.tar>).
 - c. Extract the contents (<tar -xvf rickman.tar>).
3. Load gd 1.2 onto the display system.
- a. Change to, and place the **gd1.2.tar.Z** file in, the directory under which you want to store the gd library files (referred to as *gdparent*).
 - b. Uncompress it (<uncompress gd1.2.tar>).
 - c. Extract the contents (<tar -xvf gd1.2.tar>).
 - d. Change to the *gdparent/gd1.2* directory.
 - e. Copy the **gdfontsy.c** and **gdfontsy.h** files from the *working* directory to the current directory.

f. Make the following modifications to the **Makefile**:

(1) Change the **CC=gcc** line to **CC=cc**.

```
#If the ar command fails on your system, consult the ar manpage
#for your system.
```

```
CC=cc
AR=ar
CFLAGS=-O
```

(2) Add `gdfontsy.o` and `gdfontsy.h` to the `libgd` library and `gdfontsy.o` to the AR command:

```
giftogd: giftogd.o libgd.a gd.h
        $(CC) giftogd.o -o giftogd      $(LIBS)

libgd.a: gd.o gdfontt.o gdfontsy.o gdfontmb.o gdfontl.o gdfontg.o \
        gdfontsy.o gd.h gdfontt.h gdfontsy.h gdfontmb.h gdfontl.h \
        gdfontg.h gdfontsy.h
        rm -f libgd.a
        $(AR) rc libgd.a gd.o gdfontt.o gdfontsy.o gdfontmb.o \
        gdfontl.o gdfontg.o gdfontsy.o

webgif: webgif.o libgd.a gd.h
        $(CC) webgif.o -o webgif        $(LIBS)
```

g. Create the new, gd libraries (<make>).

4. Modify the Vis-5D modification include files with the correct location of header files and gd 1.2 include files.

a. Change to the *working* directory.

b. Modify the `rickman.c` file to identify the *working* directory in the only include statement:

```
#include "working/rickman.h"
```

c. Modify the `rickmang.c` file to identify the *working* directory in the only include statement:

```
#include "working/rickmang.h"
```

d. Modify the **rickman.h** file to identify the *gdparent* directory in the include statements:

```
#include "gdparent/gd1.2/gd.h"
#include "gdparent/gd1.2/gdfonts.h"
```

e. Modify the **rickmang.h** file to identify the *gdparent* directory in the only include statements:

```
#include "gdparent/gd1.2/gd.h"
#include "gdparent/gd1.2/gdfontl.h"
#include "gdparent/gd1.2/gdfontt.h"
#include "gdparent/gd1.2/gdfonts.h"
#include "gdparent/gd1.2/gdfontsy.h"
```

5. Create the template GIF files.

a. Modify the **patemplate.c** file to identify the *gdparent* directory in the include statements:

```
#include <stdio.h>
#include "gdparent/gd1.2/gd.h"
#include "gdparent/gd1.2/gdfonts.h"
#include "gdparent/gd1.2/gdfontt.h"
#include "gdparent/gd1.2/gdfontsy.h"
```

b. Compile **patemplate.c** (<cc patemplate.c -Lgdparent/gd1.2 -lgd>).

c. Execute the template creation program (<a.out>).

6. Load the distributed Vis-5D source code.

a. Change to, and place the **vis5d-4.2.tar.Z** file in, the directory under which you want to store the Vis-5D code (referred to as *V5Dparent*).

- b. Uncompress it (<uncompress vis5d-4.2.tar>).
 - c. Extract the contents (tar -xvf vis5d-4.2.tar>).
 - d. Change to, and place the **vis5d-data.tar.Z** file in, the *V5Dparent/vis5d-4.2* directory.
 - e. Uncompress it (<uncompress vis5d-data.tar>).
 - f. Extract the contents (tar -xvf vis5d-data.tar>).
7. Make the modifications to Vis-5D.
- a. Copy the following files to the *V5Dparent/vis5d-4.2* directory from the *working* directory:

PA_out.exe	sm_template.gif
patemplate.gif	state.bdy
 - b. Copy the *pdata.dat* file from the *working* directory to the *V5Dparent* directory.
 - b. Run the Makefile without arguments (<make>).
 - c. From the list determine and note the make command for the system which you are using (ie **alpha-x** for a DEC Alpha 3000 using X Windows) for compiling later (referred to as *MAKE_ARG*).
 - d. Make the following modifications to the **Makefile**:
 - (1) Find the section which begins with the *MAKE_ARG* which corresponds to your system. Note that if your system's *MAKE_ARG* ends

with “-x”, look for a section which substitutes “-mesa” for “-x” (ie **alpha-mesa** for a DEC Alpha 3000 using X Windows).

(2) Add the gd 1.2 library and its location to the libraries (using **alpha-x** as an example):

```
alpha-mesa:
    $(MAKE) target \
    "CC = cc" \
    "CFLAGS = -c -O -Ddec -w -DSINGLE_TASK -DOPENGL -DLITTLE ∪
-DUNDERScore -I../Mesa/include" \
    "AUXOBJS = graphics.ogl.o xdump.o" \
    "LINK = cc" \
    "LIBS = ../lui5/liblui.a -L../Mesa/lib -Lgdparent/gd1.2 ∪
-lMesaGL -lMesaGLU -lX11 -lXext -lm -lgd"
```

NOTE: The “ ∪ ” in the example above is a marker in this document which means that there is no carriage return at that location. All of the text on the line with the symbol and the line after the symbol, must appear on one (1) line in the actual Makefile.

e. Change to the *V5Dparent/vis5d-4.2/src* directory.

f. Make the following modifications to **gui.c**:

(1) Add the statement to include the *working/rickmang.c* file:

```
#define MM_SLICE 4
#define MM_PROBE 5
#include "working/rickmang.c"

static GuiContext gtx_table[VIS5D_MAX_CONTEXTS];

GuiContext get_gui_gtx( int index )
```

(2) Add the four function calls to the "Function Key"

subroutines:

```
static void func6( int index )
{
rickman_func_6();
}
```

```
static void func7( int index )
{
rickman_func_7();
}
```

```
static void func8( int index )
{
rickman_func_8();
}
```

```
static void func9( int index )
{
rickman_func_9();
}
```

g. Make the following modifications to **render.c**:

(1) Add the statement to include the *working/rickman.c* file:

```
/* Vertical spacing between rows of text: (in pixels) */
#define VSPACE 1
#include "working/rickman.c"

/** float2string
*****
Convert a float into an ascii string.
*****/
```

(2) Add the setup function call to the "draw_probe" subroutine:

```
        if (w>x)
            x = w;
    }
    rickman_setup_proc(ctx);
}

set_color( ctx->BoxColor );

/* Draw from bottom of window upward */
```

(3) Add the location, data write, and fail data function calls to the "draw_probe" subroutine:

```
    draw_text( x+10, y, str );
    y -= (ctx->FontHeight+VSPACE);
    rickman_location_proc(r,c,l);
}
if (pSc) rickman_data_write_proc(ctx);
else rickman_fail_data_proc();
}
```

8. Compile Vis-5D.

- a. Change to the *V5Dparent/vis5d-4.2* directory.
- b. Compile the program (<make *MAKE_ARG*>).

APPENDIX 7.1.1 - gdfontsy.c

The following program is written in C. It is the data for a modification to gd 1.2, used in the modified version of Vis-5D referenced in this document.

```
/* CREATE a symbol character set based on the
   gdFontSmall (6 X 12) font.

   INCLUDE the gdfontsy.h file
*/
#include "gdfontsy.h"

char gdFontSymbolData[] = {
    0,0,0,0,0,0,
    0,0,0,0,0,0,
    0,0,0,0,0,0,
    0,0,0,0,0,0,
    0,0,0,0,0,0,
    0,0,0,0,0,0,
    0,0,0,0,0,0,
    0,0,0,0,0,0,
    0,0,0,0,0,0,
    0,0,0,0,0,0,
    0,0,0,0,0,0,
    0,0,0,0,0,0,
    0,0,0,0,0,0,
    0,0,0,0,0,0,

    0,0,0,0,0,0,
    0,0,0,0,0,0,
    0,0,1,0,0,0,
    0,0,1,0,0,0,
    0,0,1,0,0,0,
    0,0,1,0,0,0,
    0,0,1,0,0,0,
    0,0,1,0,0,0,
    0,0,1,0,0,0,
    0,0,0,0,0,0,
    0,0,1,0,0,0,
    0,0,0,0,0,0,
    0,0,0,0,0,0,
    0,0,0,0,0,0,
    0,0,0,0,0,0,
```

0,0,0,0,0,0,
0,1,0,1,0,0,
0,1,0,1,0,0,
0,1,0,1,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,1,0,1,0,0,
1,1,1,1,1,0,
0,1,0,1,0,0,
0,1,0,1,0,0,
1,1,1,1,1,0,
0,1,0,1,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
1,1,1,1,1,0,
0,0,0,0,1,0,
0,0,0,0,1,0,
0,0,0,0,1,0,
1,1,1,1,1,0,
0,0,0,0,1,0,
0,0,0,0,1,0,
0,0,0,0,1,0,
1,1,1,1,1,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
1,1,1,1,1,0,
1,1,0,0,1,0,
0,0,0,1,0,0,
0,0,1,0,0,0,
0,1,0,0,0,0,
1,0,0,1,1,0,
1,0,0,1,1,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,1,0,0,0,0,
1,0,1,0,0,0,
1,0,1,0,0,0,
0,1,0,0,0,0,
1,0,1,0,1,0,
1,0,0,1,0,0,
0,1,1,0,1,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,1,1,0,0,0,
0,1,1,0,0,0,
1,1,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,1,0,
0,0,0,1,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,0,1,0,0,
0,0,0,0,1,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
1,0,0,0,0,0,
0,1,0,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,1,0,0,0,0,
1,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,1,0,0,0,
1,0,1,0,1,0,
0,1,1,1,0,0,
0,0,1,0,0,0,
0,1,1,1,0,0,
1,0,1,0,1,0,
0,0,1,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
1,1,1,1,1,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,1,1,0,0,0,
0,1,1,0,0,0,
1,1,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
1,1,1,1,1,1,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,1,1,0,0,0,
0,1,1,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,1,0,
0,0,0,1,0,0,
0,0,1,0,0,0,
0,1,0,0,0,0,
1,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,1,1,1,0,0,
1,0,0,0,1,0,
1,0,0,1,1,0,
1,0,1,0,1,0,
1,1,0,0,1,0,
1,0,0,0,1,0,
0,1,1,1,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,1,0,0,0,
0,1,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,1,1,1,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,1,1,1,0,0,
1,0,0,0,1,0,
0,0,0,0,1,0,
0,0,0,1,0,0,
0,0,1,0,0,0,
0,1,0,0,0,0,
1,1,1,1,1,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,1,1,1,0,0,
1,0,0,0,1,0,
0,0,0,0,1,0,
0,0,1,1,0,0,
0,0,0,0,1,0,
1,0,0,0,1,0,
0,1,1,1,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,1,0,0,
0,0,1,1,0,0,
0,1,0,1,0,0,
1,0,0,1,0,0,
1,1,1,1,1,0,
0,0,0,1,0,0,
0,0,0,1,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
1,1,1,1,1,0,
1,0,0,0,0,0,
1,1,1,1,0,0,
0,0,0,0,1,0,
0,0,0,0,1,0,
1,0,0,0,1,0,
0,1,1,1,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,1,1,0,0,
0,1,0,0,0,0,
1,0,0,0,0,0,
1,1,1,1,0,0,
1,0,0,0,1,0,
1,0,0,0,1,0,
0,1,1,1,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
1,1,1,1,1,0,
0,0,0,0,1,0,
0,0,0,1,0,0,
0,0,0,1,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,1,1,1,0,0,
1,0,0,0,1,0,
1,0,0,0,1,0,
0,1,1,1,0,0,
1,0,0,0,1,0,
1,0,0,0,1,0,
0,1,1,1,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,1,1,1,0,0,
1,0,0,0,1,0,
1,0,0,0,1,0,
0,1,1,1,1,0,
0,0,0,0,1,0,
0,0,0,1,0,0,
0,1,1,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,1,1,0,0,0,
0,1,1,0,0,0,
0,0,0,0,0,0,
0,1,1,0,0,0,
0,1,1,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,1,1,0,0,0,
0,1,1,0,0,0,
0,0,0,0,0,0,
0,1,1,0,0,0,
0,1,1,0,0,0,
1,1,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,1,0,
0,0,0,1,0,0,
0,0,1,0,0,0,
0,1,1,1,1,1,
0,0,1,0,0,0,
0,0,0,1,0,0,
0,0,0,0,1,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
1,1,1,1,1,0,
0,0,0,0,0,0,
1,1,1,1,1,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
1,0,0,0,0,0,
0,1,0,0,0,0,
0,0,1,0,0,0,
1,1,1,1,0,0,
0,0,1,0,0,0,
0,1,0,0,0,0,
1,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,1,1,1,0,0,
1,0,0,0,1,0,
0,0,0,1,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,0,0,0,0,
0,0,1,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,1,1,0,1,0,
1,0,1,1,0,0,
0,0,0,0,0,0,
1,1,1,1,1,0,
0,0,0,0,0,0,
1,1,1,1,1,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,1,1,1,0,0,
1,0,0,0,1,0,
1,0,0,0,1,0,
1,1,1,1,1,0,
1,0,0,0,1,0,
1,0,0,0,1,0,
1,0,0,0,1,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
1,1,1,1,0,0,
1,0,0,0,1,0,
1,0,0,0,1,0,
1,1,1,1,0,0,
1,0,0,0,1,0,
1,0,0,0,1,0,
1,1,1,1,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
1,1,0,1,1,0,
1,0,0,0,1,0,
0,1,0,1,0,0,
0,0,1,0,0,0,
0,1,0,1,0,0,
1,0,0,0,1,0,
1,1,0,1,1,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,1,0,0,0,
0,1,0,1,0,0,
0,1,0,1,0,0,
0,1,0,1,0,0,
1,0,0,0,1,0,
1,0,0,0,1,0,
1,1,1,1,1,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
1,1,1,1,1,0,
1,0,0,0,0,0,
1,0,0,0,0,0,
1,1,1,1,0,0,
1,0,0,0,0,0,
1,0,0,0,0,0,
1,1,1,1,1,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
1,1,1,1,1,0,
0,0,1,0,0,0,
0,1,1,1,0,0,
1,0,1,0,1,0,
0,1,1,1,0,0,
0,0,1,0,0,0,
1,1,1,1,1,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
1,1,1,1,1,0,
1,0,0,0,1,0,
1,0,0,0,0,0,
1,0,0,0,0,0,
1,0,0,0,0,0,
1,0,0,0,0,0,
1,0,0,0,0,0,
1,1,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
1,0,0,0,1,0,
1,0,0,0,1,0,
1,0,0,0,1,0,
1,1,1,1,1,0,
1,0,0,0,1,0,
1,0,0,0,1,0,
1,0,0,0,1,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,1,1,1,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,1,1,1,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,1,1,0,0,
0,0,1,0,1,0,
0,0,0,1,1,0,
0,0,0,0,1,0,
1,0,0,0,1,0,
1,0,0,0,1,0,
0,1,1,1,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
1,0,0,0,1,0,
1,0,0,1,0,0,
1,0,1,0,0,0,
1,1,0,0,0,0,
1,0,1,0,0,0,
1,0,0,1,0,0,
1,0,0,0,1,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,1,0,0,0,
0,1,0,1,0,0,
0,1,0,1,0,0,
0,1,0,1,0,0,
1,0,0,0,1,0,
1,0,0,0,1,0,
1,0,0,0,1,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
1,0,0,0,1,0,
1,1,0,1,1,0,
1,0,1,0,1,0,
1,0,0,0,1,0,
1,0,0,0,1,0,
1,0,0,0,1,0,
1,0,0,0,1,0,
1,0,0,0,1,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
1,0,0,0,1,0,
1,0,0,0,1,0,
1,1,0,0,1,0,
1,0,1,0,1,0,
1,0,0,1,1,0,
1,0,0,0,1,0,
1,0,0,0,1,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,1,1,1,0,0,
1,0,0,0,1,0,
1,0,0,0,1,0,
1,0,0,0,1,0,
1,0,0,0,1,0,
1,0,0,0,1,0,
0,1,1,1,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
1,1,1,1,1,0,
0,1,0,1,0,0,
0,1,0,1,0,0,
0,1,0,1,0,0,
0,1,0,1,0,0,
0,1,0,1,0,0,
1,1,0,1,1,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,1,1,1,0,0,
1,0,0,0,1,0,
1,0,0,0,1,0,
1,1,1,1,1,0,
1,0,0,0,1,0,
1,0,0,0,1,0,
0,1,1,1,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
1,1,1,1,0,0,
1,0,0,0,1,0,
1,0,0,0,1,0,
1,1,1,1,0,0,
1,0,0,0,0,0,
1,0,0,0,0,0,
1,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
1,1,1,1,1,0,
0,1,0,0,1,0,
0,0,1,0,0,0,
0,0,0,1,0,0,
0,0,1,0,0,0,
0,1,0,0,1,0,
1,1,1,1,1,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
1,1,1,1,1,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
1,0,0,0,1,0,
1,0,0,0,1,0,
0,1,0,1,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,1,1,1,0,0,
1,0,0,0,0,0,
1,0,0,0,0,0,
1,0,0,0,0,0,
0,1,1,0,0,0,
0,0,0,1,0,0,
0,1,1,1,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,1,1,1,0,0,
1,0,0,0,1,0,
1,0,0,0,1,0,
1,0,0,0,1,0,
0,1,0,1,0,0,
0,1,0,1,0,0,
1,1,0,1,1,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
1,1,1,1,1,0,
1,0,0,0,1,0,
0,0,0,0,0,0,
0,1,1,1,0,0,
0,0,0,0,0,0,
1,0,0,0,1,0,
1,1,1,1,1,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
1,0,1,0,1,0,
1,0,1,0,1,0,
1,0,1,0,1,0,
0,1,1,1,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
1,1,1,1,1,0,
0,0,0,0,1,0,
0,0,0,1,0,0,
0,0,1,0,0,0,
0,1,0,0,0,0,
1,0,0,0,0,0,
1,1,1,1,1,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,1,1,1,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,1,1,1,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,1,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
1,0,0,0,1,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
1,1,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
1,1,1,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
1,1,1,1,1,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
1,1,1,1,1,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,1,1,0,0,
0,0,1,1,0,0,
0,0,0,1,1,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,1,1,0,1,0,
1,0,0,1,0,0,
1,0,0,1,0,0,
1,0,0,1,0,0,
0,1,1,0,1,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
1,1,1,0,0,0,
1,0,0,1,0,0,
1,1,1,0,0,0,
1,0,0,1,0,0,
1,0,0,0,1,0,
1,0,0,0,1,0,
1,1,1,1,0,0,
1,0,0,0,0,0,
1,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
1,0,0,0,1,0,
0,1,0,1,0,0,
0,0,1,0,0,0,
0,1,0,1,0,0,
1,0,0,0,1,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,1,1,0,0,0,
0,1,0,0,0,0,
0,1,1,1,0,0,
1,0,0,0,1,0,
1,0,0,0,1,0,
0,1,1,1,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,1,1,1,0,0,
1,0,0,0,1,0,
0,1,1,0,0,0,
1,0,0,0,1,0,
0,1,1,1,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,1,1,1,0,0,
1,0,1,0,1,0,
1,0,1,0,1,0,
0,1,1,1,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
1,1,0,0,1,0,
0,1,0,1,0,0,
0,1,0,1,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
1,0,1,1,0,0,
0,1,0,0,1,0,
0,1,0,0,1,0,
0,1,0,0,1,0,
0,1,0,0,1,0,
0,0,0,0,1,0,
0,0,0,0,1,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,1,0,
0,0,1,1,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,1,0,0,0,0,
1,0,0,1,0,0,
1,0,1,0,1,0,
1,0,1,0,1,0,
1,0,1,0,1,0,
0,1,1,1,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
1,0,0,0,1,0,
1,0,0,1,0,0,
1,1,1,0,0,0,
1,0,0,1,0,0,
1,0,0,0,1,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,1,0,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,1,0,1,0,0,
1,0,0,0,1,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
1,0,0,0,1,0,
1,0,0,0,1,0,
1,0,0,0,1,0,
1,1,0,1,1,0,
1,0,1,0,1,0,
1,0,0,0,0,0,
1,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
1,0,0,0,1,0,
0,1,0,0,1,0,
0,1,0,0,1,0,
0,1,0,1,0,0,
0,1,1,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,1,1,1,0,0,
1,0,0,0,1,0,
1,0,0,0,1,0,
1,0,0,0,1,0,
0,1,1,1,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
1,1,1,1,1,0,
0,1,0,1,0,0,
0,1,0,1,0,0,
0,1,0,1,0,0,
1,0,0,0,1,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,1,1,0,0,0,
1,0,0,1,0,0,
1,0,0,1,0,0,
1,1,1,1,0,0,
1,0,0,1,0,0,
1,0,0,1,0,0,
0,1,1,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,1,1,1,0,0,
1,0,0,0,1,0,
1,0,0,0,1,0,
1,1,0,0,1,0,
1,0,1,1,0,0,
1,0,0,0,0,0,
1,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,1,1,1,1,0,
1,0,0,1,0,0,
1,0,0,0,1,0,
1,0,0,0,1,0,
0,1,1,1,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
1,1,1,1,1,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,0,1,1,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
1,1,0,0,1,0,
0,1,0,0,1,0,
0,1,0,0,1,0,
0,1,0,0,1,0,
0,0,1,1,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
1,1,1,1,1,0,
1,0,0,0,1,0,
1,0,1,0,1,0,
1,0,1,0,1,0,
1,1,0,1,1,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,1,0,1,0,0,
1,0,0,0,1,0,
1,0,1,0,1,0,
1,0,1,0,1,0,
0,1,0,1,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
1,0,0,0,0,0,
0,1,1,1,0,0,
1,0,0,0,0,0,
1,0,0,0,0,0,
0,1,1,1,0,0,
1,0,0,0,0,0,
1,0,0,0,0,0,
0,1,1,1,1,0,
0,0,0,0,1,0,
0,1,1,1,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,1,0,0,0,
1,0,1,0,1,0,
1,0,1,0,1,0,
1,0,1,0,1,0,
0,1,1,1,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
1,0,0,0,0,0,
0,1,1,1,1,0,
1,0,0,0,0,0,
1,0,0,0,0,0,
1,0,0,0,0,0,
1,0,0,0,0,0,
1,0,0,0,0,0,
1,0,0,0,0,0,
0,1,1,1,1,0,
0,0,0,0,1,0,
0,0,1,1,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,1,0,
0,0,0,1,0,0,
0,0,0,1,0,0,
0,0,0,1,0,0,
0,0,1,0,0,0,
0,0,0,1,0,0,
0,0,0,1,0,0,
0,0,0,1,0,0,
0,0,0,1,0,0,
0,0,0,0,1,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,1,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
1,0,0,0,0,0,
0,1,0,0,0,0,
0,1,0,0,0,0,
0,1,0,0,0,0,
0,0,1,0,0,0,
0,1,0,0,0,0,
0,1,0,0,0,0,
0,1,0,0,0,0,
1,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

```

0,0,0,0,0,0,
0,1,1,0,1,0,
1,0,1,1,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,

0,0,0,0,0,0,
0,0,0,0,0,0,
1,0,0,0,0,0,
1,0,0,0,0,0,
1,1,1,1,0,0,
1,0,0,0,1,0,
1,0,0,0,1,0,
1,0,0,0,1,0,
1,1,1,1,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0,
0,0,0,0,0,0
};

gdFont gdFontSymbolRep = {
    96,
    32,
    6,
    12,
    gdFontSymbolData
};

gdFontPtr gdFontSymbol = &gdFontSymbolRep;

```

APPENDIX 7.1.2 - gdfontsy.h

The following program is written in C. It is the declaration file for a modification to gd 1.2, used in the modified version of Vis-5D referenced in this document.

```
#ifndef GDFONTSY_H
#define GDFONTSY_H 1

/* gdfontsy.h: brings in the small symbol set font.
   Also link with gdfontsy.c. */

#include "gd.h"

/* 6x12 font derived from a public domain font in the X
   distribution. Only contains the 96 standard ascii characters,
   sorry. Feel free to improve on this. */

extern gdFontPtr gdFontSymbol;

#endif
```

APPENDIX 7.1.3 - rickman.c

The following program is written in C. It is the code for the functions called from the modified version of Vis-5D's render.c file.

```
/* include the rickman.h file using #include "path/rickman.h" where
   path is the directory in which rickman.h is stored
*/
#include "/alpha2/rickman/rickman.h"

/*Create the Setup Procedure which determines and stores most of the
   unique variables for the new routines. The argument "ctx" is the
   Vis-5D context of the dataset in use.
*/
void rickman_setup_proc(Context ctx)
{
/* INITIALIZE VARIABLES:
   p = generic counting variable
   vSf = pointer array: element # is the Vis-5d variable # (+1)
                        : stored value identifies to which small
                        multiple variable the element # points
   vSc = used as a flag which determines if all the small multiple
        components are present
   black\
   dgray \
   gray > gd 1.2 color variables
   bgray /
   white/
   tPt = *POINTER string used in stdlib string functions
   str = generic labelling string
   cfPtr = *POINTER generic file pointer for stdio file functions
   im = generic image variable for gd 1.2
   points = generic vertex array for gd 1.2
*/

int p, vSf[100]={0}, vSc=0, black, dgray, gray, bgray, white;
char *tPt, str[1000];
FILE *cfPtr;
gdImagePtr im;
gdPoint points[1500];
```

```

/* IDENTIFY the element # for the seven small multiple components
*/
for (p=0;p<ctx->NumVars;p++)
{
    tPt=strtok(strcpy(str,ctx->VarName[p]),"_");

    while (tPt != NULL)
    {
        if (strcmp(tPt,"CHEM")==0 && vS[1]==-1) {vSf[p+1]=1; vS[1]=p;}
        if (strcmp(tPt,"HADV")==0 && vS[2]==-1) {vSf[p+1]=2; vS[2]=p;}
        if (strcmp(tPt,"VADV")==0 && vS[3]==-1) {vSf[p+1]=3; vS[3]=p;}
        if (strcmp(tPt,"VDIF")==0 && vS[4]==-1) {vSf[p+1]=4; vS[4]=p;}
        if (strcmp(tPt,"HDIF")==0 && vS[5]==-1) {vSf[p+1]=5; vS[5]=p;}
        if (strcmp(tPt,"DDEP")==0 && vS[6]==-1) {vSf[p+1]=6; vS[6]=p;}
        if (strcmp(tPt,"EMIS")==0 && vS[7]==-1) {vSf[p+1]=7; vS[7]=p;}
        tPt=strtok(NULL,"_");
    }

    vSc+=vSf[p+1];
}

/* IDENTIFY the path to the data source file and its filename
*/
fStr[0][0]=strcspn(strcpy(str,ctx->DataFile),"/")==0?'\/:\0';
tPt=strtok(str,"/");
strcpy(fStr[1],tPt);
tPt=strtok(NULL,"/");

while (tPt!=NULL)
{
    strcat(fStr[0],fStr[1]);
    strcat(fStr[0],"/");
    strcpy(fStr[1],tPt);
    tPt=strtok(NULL,"/");
}

/* vSc flag will only be true if all seven small multiple components are
available

BEGIN VARIABLE SETUP ROUTINE
*/
if (vSc==28)
{
/*INITIALIZE VARIABLES
    gdSc = sm_map scaling factor
    lPtr = length of current substring
    nPoly = number of polygons to read
    nSeg = number of segments to read
    zfPtr = pointer to current position in current string
    tSrt = string of characters used to identify substring bounds
*/

```



```

float gdSc;
int lPtr,nPoly,nSeg;
long zfPtr;
char tStr[4]={' ','\r','\f','\n'};

/* SET the small multiple flag to true since all small multiple
components are present
*/
pSc=TRUE;

/* DETERMINE the maximum value possible for a line in the small multiple
*/
backSc=ABS(ctx->MaxVal[vS[1]]);
if (ABS(ctx->MinVal[vS[1]])>backSc) backSc=ABS(ctx->MinVal[vS[1]]);
if (ABS((ctx->MaxVal[vS[2]])+(ctx->MaxVal[vS[5]])+
(ctx->MaxVal[vS[3]]))>backSc)
backSc=ABS((ctx->MaxVal[vS[2]])+(ctx->MaxVal[vS[5]])+
(ctx->MaxVal[vS[3]]));
if (ABS((ctx->MinVal[vS[2]])+(ctx->MinVal[vS[5]])+
(ctx->MinVal[vS[3]]))>backSc)
backSc=ABS((ctx->MinVal[vS[2]])+(ctx->MinVal[vS[5]])+
(ctx->MinVal[vS[3]]));
if (ABS((ctx->MaxVal[vS[6]])+(ctx->MaxVal[vS[4]]))>backSc)
backSc=ABS((ctx->MaxVal[vS[6]])+(ctx->MaxVal[vS[4]]));
if (ABS((ctx->MinVal[vS[6]])+(ctx->MinVal[vS[4]]))>backSc)
backSc=ABS((ctx->MinVal[vS[6]])+(ctx->MinVal[vS[4]]));
if (ABS(ctx->MaxVal[vS[7]])>backSc) backSc=ABS(ctx->MaxVal[vS[7]]);
if (ABS(ctx->MinVal[vS[7]])>backSc) backSc=ABS(ctx->MinVal[vS[7]]);
if (backSc<10.) backSc=10.;

/* STORE the constants
*/
smData[5]=50.;
smData[10]=ctx->Nc;
smData[11]=ctx->Nr;
smData[14]=ctx->Nl[vS[1]];
smData[15]=1.;
smData[16]=1.;
smData[17]=1.;
smData[18]=8.;
smData[19]=ctx->NumTimes;
gdSc=800./smData[10]<650./smData[11]?800./smData[10]:650./smData[11];

/* CREATE the sm_temp.gif file
*/
cfPtr=fopen("sm_template.gif","rb");
im=gdImageCreateFromGif(cfPtr);
fclose(cfPtr);

black = gdImageColorExact(im, 0, 0, 0);
white = gdImageColorExact(im, 255, 255, 255);

```

```

    gray = gdImageColorExact(im, 127, 127, 127);
    bgray = gdImageColorAllocate(im, 191, 191, 191);
    dgray = gdImageColorAllocate(im, 64, 64, 64);

/* DRAW THE MAP
*/
    cfPtr=fopen("state.bdy","r");
/* DETERMINE and store the number of polygons in state.bdy
*/
    fread(str,6,1,cfPtr);
    lPtr=strcspn(str,tStr);
    fseek(cfPtr,0,SEEK_SET);
    fread(str,lPtr,1,cfPtr);
    nPoly=atoi(str);

/* ADVANCE to the next data value
*/
    zfPtr=lPtr+2;
    fseek(cfPtr,zfPtr,SEEK_SET);
    fread(str,6,1,cfPtr);
    lPtr=strcspn(str,tStr);
    fseek(cfPtr,zfPtr,SEEK_SET);

/* DETERMINE and store the number of segments in the first polygon
*/
    fread(str,lPtr,1,cfPtr);
    nSeg=atoi(str);

/* ADVANCE to the next data value
*/
    zfPtr+=lPtr+2;
    fseek(cfPtr,zfPtr,SEEK_SET);

/* BEGIN POLYGON LOOP
*/
    for(p=1;p<=nPoly;p++)
    {
/* DEFINE VARIABLES:
        q = generic counting variable
        Rng = number of digits in the data value
        nPts = numer of points stored in Pstore
        Pstore = array of boundary points which are in the screen domain
        a\
        b > generic floats
        c/
        inFlag = flags whether previous point was in the screen domain
        drawFlag = flags whether any points were in the screen domain
*/
        int q,Rng,nPts=0;
        float Pstore[3][1500]={0.},a,b,c;
        short inFlag,drawFlag=FALSE;

```

```

/* DETERMINE length of next data values and advance to it
*/
    fread(str,26,1,cfPtr);
    Rng=str[25]==' '?15:13;
    fseek(cfPtr,zfPtr,SEEK_SET);

/* READ segments into Pstore
*/
    for(q=0;q<nSeg;q++)
        {
            fread(str,Rng,1,cfPtr);
            Pstore[1][q]=-1.*atof(str);
            fread(str,11,1,cfPtr);
            Pstore[0][q]=atof(str);
            Pstore[2][q]=0.;
            zfPtr+=11+Rng;
        }

/* CONVERT Pstore Lat/Lon data to screen xyz data
*/
    geo_to_xyz(ctx,1,1,nSeg,Pstore[0],Pstore[1],Pstore[2],
              Pstore[0],Pstore[1],Pstore[2]);

/* CONVERT first Pstore point from xyz to Vis-5D grid
*/
    xyz_to_grid(ctx,1,1,Pstore[0][nSeg-1],Pstore[1][nSeg-
1],0.,&a,&b,&c);

/* DETERMINE if first point is in the screen domain then scale it to the
map and store in points
*/
    inFlag=(a<=ctx->Nr-1 && a>=0. && b<=ctx->Nc-1 && b>=0.)?
        TRUE:FALSE;
    points[nSeg-1].x=51+gdSc*b;
    points[nSeg-1].y=51+gdSc*a;

/* BEGIN SEGMENT LOOP
*/
    for(q=0;q<nSeg;q++)
        {
/* DEFINE VARIABLE
    m = slope of segment
REDEFINE USE OF VARIABLES
    c = y intercept
    Rng = point location pointer
*/
            float m;

```

```

/* CONVERT current point from xyz to Vis-5D grid and establish slope and
y intercept of segment from current point to previous point
*/
    xyz_to_grid(ctx,1,1,Pstore[0][q],Pstore[1][q],0.,&a,&b,&c);
    Rng=(nPts==0?nSeg-1:nPts-1);
    m=(a-((points[Rng].y-51.)/gdSc))/
      (b-((points[Rng].x-51.)/gdSc));
    c=(points[Rng].y-51.)/gdSc-m*(points[Rng].x-51.)/gdSc;

/* BEGIN POINT STORAGE CONDITIONAL
*/
    if (a<=ctx->Nr-1 && a>=0. && b<=ctx->Nc-1 && b>=0.)
    {
        drawFlag=TRUE;
    }

/* BEGIN POINT INTERPOLATION CONDITIONAL
If the previous point was outside the screen domain and the
current one is inside, determine at what point the segment would
intercept the screen boundary
*/
    if (!inFlag)
    {
        points[nPts+1].x=51+gdSc*b;
        points[nPts+1].y=51+gdSc*a;
        xyz_to_grid(ctx,1,1,Pstore[0][q-1],Pstore[1][q-1],0.,
                    &a,&b,&c);
        m=((points[nPts+1].y-51.)/gdSc-a)/
          ((points[nPts+1].x-51.)/gdSc-b);
        c=(points[nPts+1].y-51.)/gdSc-m*(points[nPts+1].x-51.)/
          gdSc;

/* FOUR conditionals to check if a coner was turned while outside the
screen domain
*/
        if (a>ctx->Nr-1)
        {
            points[nPts].y=51.+gdSc*(ctx->Nr-1);
            points[nPts].x=51.+((float)(ctx->Nr-1)-c)/m*gdSc;
        }

        if (a<0.)
        {
            points[nPts].y=51.;
            points[nPts].x=51.-c/m*gdSc;
        }
    }

```

```

        if (b>ctx->Nc-1)
        {
            points[nPts].x=51.+gdSc*(ctx->Nc-1);
            if (!(a>ctx->Nr-1 || a<0.))
                points[nPts].y=51.+(m*(ctx->Nc-1)+c)*gdSc;
        }

        if (b<0.)
        {
            points[nPts].x=51.;
            if (!(a>ctx->Nr-1 || a<0.)) points[nPts].y=51.+gdSc*c;
        }

        inFlag=TRUE;
        nPts+=2;
    }
/* END POINT INTERPOLATION CONDITIONAL
*/

    else
    {
        points[nPts].x=51+gdSc*b;
        points[nPts].y=51+gdSc*a;
        nPts++;
    }

}
/* END POINT STORAGE CONDITIONAL

BEGIN POINT STORAGE ALTERNATIVE
CHECK to see if a coner was turned upon exiting the scen domain
and set the inFlag to FALSE
*/

    else
    {
        if (a>ctx->Nr-1)
        {
            points[nPts].y=51.+gdSc*(ctx->Nr-1);
            points[nPts].x=51.+((float)(ctx->Nr-1)-c)/m*gdSc;
        }

        if (a<0.)
        {
            points[nPts].y=51.;
            points[nPts].x=51.-c/m*gdSc;
        }
    }

```

```

        if (b>ctx->Nc-1)
        {
            points[nPts].x=51.+gdSc*(ctx->Nc-1);
            if (!(a>ctx->Nr-1 || a<0.))
                points[nPts].y=51.+(m*(ctx->Nc-1)+c)*gdSc;
        }

        if (b<0.)
        {
            points[nPts].x=51.;
            if (!(a>ctx->Nr-1 || a<0.)) points[nPts].y=51.+gdSc*c;
        }

        inFlag=FALSE;
        nPts++;
    }
/* END POINT STORAGE ALTERNATIVE
*/
    }
/* END SEGMENT LOOP

DRAW polygon if it is in the seen domain
*/
    if(drawFlag) gdImagePolygon(im,points,nPts,dgray);

/* RESET drawFlag, advance to next data value, determine the number of
segments in the next polygon
*/
    drawFlag=FALSE;
    zfPtr+=1;
    fseek(cfPtr,zfPtr,SEEK_SET);
    fread(str,6,1,cfPtr);
    lPtr=strcspn(str,tStr);
    fseek(cfPtr,zfPtr,SEEK_SET);
    fread(str,lPtr,1,cfPtr);
    nSeg=atoi(str);
    zfPtr+=lPtr+2;
    fseek(cfPtr,zfPtr,SEEK_SET);
}
/* END POLYGON LOOP
*/

    fclose(cfPtr);

/* DRAW Bounding Box
*/
    points[0].x=51;
    points[0].y=51;
    points[1].x=51+(ctx->Nc-1)*gdSc;
    points[1].y=51;
    points[2].x=51+(ctx->Nc-1)*gdSc;

```

```

    points[2].y=51+(ctx->Nr-1)*gdSc;
    points[3].x=51;
    points[3].y=51+(ctx->Nr-1)*gdSc;

    gdImagePolygon(im,points,4,dgray);

/* SAVE map as sm_back.gif
*/
    sprintf(str,"%ssm_back.gif",fStr[0]);
    cfPtr=fopen(str,"wb");
    gdImageGif(im, cfPtr);
    fclose(cfPtr);

/* LABEL map and save as sm_temp.gif
*/
    sprintf(str,"Radial Scale: -50.0 to +50.0");
    gdImageString(im,gdFontSmall,500-strlen(str)*3,725,str,bgray);

    sprintf(str,"%ssm_temp.gif",fStr[0]);
    cfPtr=fopen(str,"wb");
    gdImageGif(im, cfPtr);
    fclose(cfPtr);
    gdImageDestroy(im);
}
/* END VARIABLE SETUP ROUTINE

BEGIN ALTERNATIVE TO SET UP ROUTINE
IF the vSc flag is not met create a gif which explains why
*/
else
{
    pSc=FALSE;
    im=gdImageCreate(210,310);

    black = gdImageColorAllocate(im, 0, 0, 0);
    gray = gdImageColorAllocate(im, 127, 127, 127);
    white = gdImageColorAllocate(im, 255, 255, 255);

    gdImageString(im,gdFontSmall,10,25,"No Small Multiple",gray);
    gdImageString(im,gdFontSmall,10,35,"The Following Data Sets",gray);
    gdImageString(im,gdFontSmall,10,45,"Are Missing:",gray);
    p=55;

/* DETERMINE which small multiple componet(s) is(are) missing
*/
    if (vS[1]==-1)
        {
            gdImageString(im,gdFontSmall,10,55,"CHEM",white);
            p+=10;
        }
}

```

```

    if (vS[2]==-1)
    {
        gdImageString(im,gdFontSmall,10,p,"HADV",white);
        p+=10;
    }
    if (vS[3]==-1)
    {
        gdImageString(im,gdFontSmall,10,p,"VADV",white);
        p+=10;
    }
    if (vS[4]==-1)
    {
        gdImageString(im,gdFontSmall,10,p,"VDIF",white);
        p+=10;
    }
    if (vS[5]==-1)
    {
        gdImageString(im,gdFontSmall,10,p,"HDIF",white);
        p+=10;
    }
    if (vS[6]==-1)
    {
        gdImageString(im,gdFontSmall,10,p,"DDEP",white);
        p+=10;
    }
    if (vS[7]==-1)
    {
        gdImageString(im,gdFontSmall,10,p,"EMIS",white);
        p+=10;
    }

    sprintf(str,"%sno_sm.gif",fStr[0]);
    cfPtr=fopen(str,"wb");
    gdImageGif(im, cfPtr);
    fclose(cfPtr);
    gdImageDestroy(im);
}
/* END ALTERNATIVE TO SET UP ROUTINE
*/
}
/* END rickman_setup_proc

/* CREATE routine to store current probe location
   arguments are the column, row, and level calculated in the main
   program
*/
void rickman_location_proc(float row, float column, float level)
{
    smData[0]=column;
    smData[1]=row;

```



```

smData[12]=level;
}

/* CREATE the routine to draw the small multiples. The argument "ctx"
   is the Vis-5D context of the dataset in use.
*/
void rickman_data_write_proc (Context ctx)
{
/* INITIALIZE VARIABLES:
   x\
   y > generic storage variables
   z/
   SM_scale = array of component values at probe location
           1 = CHEM value
           2 = HADV value
           3 = VADV value
           4 = VDIF value
           5 = HDIF value
           6 = DDEP value
           7 = EMIS value
   frontSc = radial scale of independent small multiple display
   black\
   dgray \
   gray > gd 1.2 color variables
   bgray /
   white/
   red /
   p = generic counting variable
   str = generic labelling string
   cfPtr = *POINTER generic file pointer for stdio file functions
   im = generic image variable for gd 1.2
   points = generic vertex array for gd 1.2
*/

float x,y,z,SM_scale[8],frontSc;
int p,black,dgray,gray,bgray,white,red;
char str[50];
FILE *cfPtr;
gdImagePtr im;
gdPoint points[4];

/* DETERMINE height of probe for map display
*/
switch(ctx->CoordFlag)
{
case 0:
xyz_to_grid( ctx, 1, -1, ctx->CursorX, ctx->CursorY, ctx->CursorZ,
            &y, &x, &z);
}
}

```

```

        break;
    case 1:
        xyz_to_geo( ctx, 1, -1, ctx->CursorX, ctx->CursorY, ctx->CursorZ,
                   &y, &x, &z );
        z=VERT(z);
    }

/* STORE the small multiple coordinates (height and time)
*/
smData[2]=z;
smData[4]=ctx->TimeStamp[ctx->CurTime];
smData[3]=ctx->DayStamp[ctx->CurTime];
smData[13]=ctx->CurTime;

/* DETERMINE and store the values for each small multiple component
*/
for (p=1;p<8;p++)
    {
        x = interpolate_grid_value( ctx, ctx->CurTime, vs[p], smData[1],
smData[0], smData[12] );
        if (IS_MISSING(x)) SM_scale[p]=smData[5]*-1.;
        else SM_scale[p]=x;
    }
smData[9]=SM_scale[7];
smData[7]=SM_scale[2]+SM_scale[5]+SM_scale[3];
smData[8]=SM_scale[1];
smData[6]=SM_scale[6]+SM_scale[4];

/* DETERMINE the radial scale for the independent small multiple display
(use either sm_map scale or the maximum scale if sm_map scale is too
small)
*/
frontSc=(ABS(smData[6])<=smData[5] && ABS(smData[7])<=smData[5] &&
ABS(smData[8])<=smData[5] && ABS(smData[9])<=smData[5]) ?
smData[5] : backSc;

/*CREATE the independent small multiple
*/
im=gdImageCreate(210,310);
black = gdImageColorAllocate(im, 0, 0, 0);
red = gdImageColorAllocate(im, 255, 0, 0);
gray = gdImageColorAllocate(im, 127, 127, 127);
bgray = gdImageColorAllocate(im, 128, 128, 128);
white = gdImageColorAllocate(im, 255, 255, 255);
points[0].y=105.;
points[1].x=105.;
points[2].y=105.;
points[3].x=105.;

gdImageString(im,gdFontSmall,10,250,"Radial Scale:",bgray);
sprintf(str,"-%.1f to +%.1f",frontSc,frontSc);

```

```

gdImageString(im,gdFontSmall,90,250,str,white);
gdImageLine(im,5.,105.,205.,105.,gray);
gdImageLine(im,105.,5.,105.,205.,gray);
gdImageArc(im,105,105,102,102,0,360,gray);
gdImageLine(im,(5.+50.*((frontSc-SM_scale[7])/frontSc)),105.,105.+
              (50.*((frontSc+SM_scale[2]+SM_scale[5]+SM_scale[3])
                  /frontSc)),105.,red);
gdImageLine(im,105.,(105.+50.*((frontSc+SM_scale[1])/frontSc)),105.,
              105.-(50.*((frontSc+SM_scale[6]+SM_scale[4])/frontSc)),red);

points[0].x=(5.+50.*((frontSc-SM_scale[7])/frontSc));
points[1].y=105.-(50.*((frontSc+SM_scale[6]+SM_scale[4])/frontSc));
points[2].x=105.+(50.*((frontSc+SM_scale[2]+SM_scale[5]+SM_scale[3])
                    /frontSc));
points[3].y=(105.+50.*((frontSc+SM_scale[1])/frontSc));

gdImagePolygon(im,points,4,red);

/*STORE the indepent small multiple in sm.gif
*/
sprintf(str,"%ssm.gif",fStr[0]);
cfPtr=fopen(str,"wb");
gdImageGif(im, cfPtr);
fclose(cfPtr);
gdImageDestroy(im);
}
/* END OF rickman_data_write_proc
*/

/* CREATE routine for loading small multiple failure gif and saving it
as sm.gif
*/
void rickman_fail_data_proc(void)
{
/* INITIALIZE VARIABLES:
    black\
    gray > gd 1.2 color variables
    white/
    str = generic labelling string
    cfPtr = *POINTER generic file pointer for stdio file functions
    im = generic image variable for gd 1.2
*/
int black,gray,white;
char str[50];
FILE *cfPtr;
gdImagePtr im;

sprintf(str,"%sno_sm.gif",fStr[0]);

```

```
cfPtr=fopen(str,"rb");
im=gdImageCreateFromGif(cfPtr);
fclose(cfPtr);

black = gdImageColorExact(im, 0, 0, 0);
white = gdImageColorExact(im, 255, 255, 255);
gray = gdImageColorExact(im, 127, 127, 127);

sprintf(str,"%ssm.gif",fStr[0]);
cfPtr=fopen(str,"wb");
gdImageGif(im, cfPtr);
fclose(cfPtr);
gdImageDestroy(im);
}
/* END OF rickman_fail_data_proc
*/
```

APPENDIX 7.1.4 - rickmang.c

The following program is written in C. It is the code for the functions called from the modified version of Vis-5D's gui.c file.

```
/* include the rickmang.h file using #include "path/rickmang.h"
   where path is the directory in which rickmang.h is stored
*/
#include "/alpha2/rickman/rickmang.h"

/* CREATE the function performed when <F6> is pressed
   Used to change the settings for the IRR Process Analysis Diagram
   post-processor
*/
void rickman_func_6(void)
{
/* DEFINE VARIABLES:
   x = user input switch variable
   y = user input value variable
*/
int x=0,y=0;

/* DISPLAY current values
*/
printf("(1) Sample X Grid Size is %d Grids.\n", (int)smData[15]);
printf("(2) Sample Y Grid Size is %d Grids.\n", (int)smData[16]);
printf("(3) Sample Z Grid Size is %d Grids.\n", (int)smData[17]);
printf("(4) Sample Time Span is %d Steps.\n", (int)smData[18]);

/* SOLICITE Variable to change
*/
printf("Enter the Sample Size to Change or (5) to quit.\n");
scanf("%d", &x);

/* BEGIN CHANGE LOOP CONDITIONAL
*/
while (x!=5)
{

/* BEGIN CHANGE SWITCH
*/
switch (x)
{
```

```

/* CHANGE Sample X Grid Size
*/
    case 1:
        while (y<1 || y>(int)smData[10])
            {
                printf("Sample X Grid Size is currently %d Grids.\n",
                    (int)smData[15]);
                printf("Change to [1 .. %d] ",(int)smData[10]);
                scanf("%d",&y);
                if (y<1 || y>(int)smData[10]) printf("Try Again.\n");
            }

        smData[15]=y;
        break;

/* CHANGE Sample Y Grid Size
*/
    case 2:
        while (y<1 || y>(int)smData[11])
            {
                printf("Sample Y Grid Size is currently %d Grids.\n",
                    (int)smData[16]);
                printf("Change to [1 .. %d] ",(int)smData[11]);
                scanf("%d",&y);
                if (y<1 || y>(int)smData[11]) printf("Try Again.\n");
            }

        smData[16]=y;
        break;

/* CHANGE Sample Z Grid Size
*/
    case 3:
        while (y<1 || y>(int)smData[14])
            {
                printf("Sample Z Grid Size is currently %d Grids.\n",
                    (int)smData[17]);
                printf("Change to [1 .. %d] ",(int)smData[14]);
                scanf("%d",&y);
                if (y<1 || y>(int)smData[14]) printf("Try Again.\n");
            }

        smData[17]=y;
        break;

/*CHANGE Sample Time Span
*/

```

```

    case 4:
        while (y<1 || y>(int)smData[19])
            {
                printf("Sample Time Span is currently %d Steps.\n",
                    (int)smData[18]);
                printf("Change to [1 .. %d] ",(int)smData[19]);
                scanf("%d",&y);
                if (y<1 || y>(int)smData[19]) printf("Try Again.\n");
            }

        smData[18]=y;
        break;

/* INCORRECT VALUE ENTERED
*/
    default:
        printf("Please Enter a single digit, range [1 .. 5]\n");
    }
/* END CHANGE SWITCH
*/
    y=0;

/* DISPLAY current values
*/
    printf("(1) Sample X Grid Size is %d Grids.\n",(int)smData[15]);
    printf("(2) Sample Y Grid Size is %d Grids.\n",(int)smData[16]);
    printf("(3) Sample Z Grid Size is %d Grids.\n",(int)smData[17]);
    printf("(4) Sample Time Span is %d Steps.\n",(int)smData[18]);

/* SOLICITE Variable to change
*/
    printf("Enter the Sample Size to Change or (5) to quit.\n");
    scanf("%d",&x);
    }
/* END CHANGE LOOP CONDITIONAL
*/

}
/* END rickman_func_6
*/

/* CREATE the function performed when <F7> is pressed
Used to produce a small multiple on the map at the probe location and
draw an IRR Process Analysis Diagram
*/
void rickman_func_7(void)
{
/* DEFINE VARIABLES
    cfPtr = *POINTER generic file pointer for stdio file functions
    y = generic counting and value variable

```

```

        black\
        dgray \
        gray > gd 1.2 color variables
        bgray /
        white/
        red /
        rcSc = map scaling factor
        xx = maximum x point on the map
        yy = maximum y point on the map
        paData = Process Analysis post-processor data storage array
        im = generic image variable for gd 1.2
        points = generic vertex array for gd 1.2
        str = generic labelling string
*/
FILE *cfPtr;
int y,black,red,dgray,gray,bgray,white;
float rcSc,xx,yy,paData[33]={0.};
gdImagePtr im;
gdPoint points[4];
char str[100];

/* BEGIN SM PLOTTING CONDITIONAL
   IF all of the small multiple components are available and all of the
   current values are within the current range add the plot to the map
*/
if(pSc && ABS(smData[6])<=smData[5] && ABS(smData[7])<=smData[5]
    && ABS(smData[8])<=smData[5] && ABS(smData[9])<=smData[5] )
    {
        smMax++;

/* READ in the current map from sm_temp.gif
*/
        sprintf(str,"%ssm_temp.gif",fStr[0]);
        cfPtr=fopen(str,"rb");
        im=gdImageCreateFromGif(cfPtr);
        fclose(cfPtr);
        black = gdImageColorExact(im, 0, 0, 0);
        white = gdImageColorExact(im, 255, 255, 255);
        bgray = gdImageColorExact(im, 191, 191, 191);
        gray = gdImageColorExact(im, 127, 127, 127);
        dgray = gdImageColorExact(im, 64, 64, 64);
        red = gdImageColorExact(im, 255, 0, 0);

/* If sm_temp did not already contain red, create it
*/
        if(red==-1) red = gdImageColorAllocate(im, 255, 0, 0);

/* SET constants
*/
        rcSc=800./smData[10]<650./smData[11]?800./smData[10]:650./smData[11];
        xx=51+rcSc*smData[0];

```



```

yy=51+rcSc*smData[1];
points[0].y=yy;
points[1].x=xx;
points[2].y=yy;
points[3].x=xx;

/* DRAW the background sm plot
*/
gdImageLine(im,xx-50.,yy,xx+50.,yy,gray);
gdImageLine(im,xx,yy-50.,xx,yy+50.,gray);
gdImageArc(im,xx,yy,52,52,0,360,gray);

/* DRAW the current small multiple
*/
gdImageLine(im,(xx-50.+25.*((smData[5]-smData[9])/smData[5])),yy,
            xx+(25.*((smData[5]+smData[7])/smData[5])),yy,red);
gdImageLine(im,xx,(yy+25.*((smData[5]+smData[8])/smData[5])),
            xx,yy-(25.*((smData[5]+smData[6])/smData[5])),red);

points[0].x=(xx-50.+25.*((smData[5]-smData[9])/smData[5]));
points[1].y=yy-(25.*((smData[5]+smData[6])/smData[5]));
points[2].x=xx+(25.*((smData[5]+smData[7])/smData[5]));
points[3].y=yy+25.*((smData[5]+smData[8])/smData[5]);

gdImagePolygon(im,points,4,red);

/* Label the plot with time, date, and height
*/
sprintf(str,"%d", smMax);
gdImageString(im,gdFontLarge,xx-50.,yy-50.,str,bgray);
y=4*(int)smData[3]/1461;
sprintf(str,"D %5d",1000*y+(int)smData[3]-(365*y+(y-1)/4));
gdImageString(im,gdFontTiny,xx+15.,yy-50.,str,bgray);
sprintf(str,"T %02d:%02d:%02d", (int)smData[4]/3600,
        ((int)smData[4]/60)%60, (int)smData[4]%60);
gdImageString(im,gdFontTiny,xx+1.,yy-38.,str,bgray);
sprintf(str,"Z %.2f", smData[2]);
gdImageString(im,gdFontTiny,xx+50.-strlen(str)*5,yy+38.,str,bgray);

/* Save the current map in sm_temp.gif
*/
sprintf(str,"%ssm_temp.gif",fStr[0]);
cfPtr=fopen(str,"wb");
gdImageGif(im, cfPtr);
fclose(cfPtr);
gdImageDestroy(im);

/* BEGIN IRR PA DIAGRAM CONDITIONAL
   IF their is sufficient data to fill the current sample size
*/

```

```

    if ((int) (smData[1]+0.01) - (int) (smData[16]) >=-1 &&
        (int) (smData[0]+0.01) + (int) (smData[15]) <=smData[10] &&
        (int) (smData[12]+0.01) + (int) (smData[17]) <=smData[14] &&
        (int) smData[13] + (int) smData[18] <=smData[19])
    {

/* SEND command to the post-processor
*/
    sprintf(str, "PA_out.exe %s%s.pa %d %d %d %d %d %d %d %d",
            fStr[0], fStr[1], (int) smData[11] - (int) (smData[1]+0.01),
            (int) (smData[16]), (int) (smData[0]+0.01)+1,
            (int) (smData[15]), (int) (smData[12]+0.01)+1,
            (int) (smData[17]), (int) smData[13]+1, (int) smData[18]);
    system(str);

/* OPEN the results file
*/
    if((cfPtr=fopen("pa.dat", "r"))==NULL)
        printf("No pa.dat file available.\n");

/* BEGIN PA DIAGRAM DRAW ROUTINE
*/
    else
    {
/* READ in the data from pa.out
*/
        for (y=1; y<=32; y++) fscanf(cfPtr, "%f", paData[y]);
        fclose(cfPtr);

/* LOAD the PA Diagram template
*/
        cfPtr=fopen("patemplate.gif", "rb");
        im=gdImageCreateFromGif(cfPtr);
        fclose(cfPtr);

        black = gdImageColorExact(im, 0, 0, 0);
        white = gdImageColorExact(im, 255, 255, 255);

/* PLOT the PA results in their positions
*/
        /* 1 */
        sprintf(str, "%.2f ppb", paData[1]);
        gdImageString(im, gdFontSmall, 75-strlen(str)*3, 713, str, white);

        /* 2 */
        sprintf(str, "%.2f ppb", paData[2]);
        gdImageString(im, gdFontSmall, 175-strlen(str)*3, 713, str, white);

        /* 3 */
        sprintf(str, "%.2f ppb", paData[3]);
        gdImageString(im, gdFontSmall, 275-strlen(str)*3, 713, str, white);
    }
}

```

```

/* 4 */
sprintf(str, "%.2f ppb", paData[4]);
gdImageString(im, gdFontSmall, 175-strlen(str)*3, 567, str, white);

/* 5 */
sprintf(str, "%.2f ppb", paData[5]);
gdImageString(im, gdFontSmall, 375-strlen(str)*3, 567, str, white);

/* 6 */
sprintf(str, "%.2f ppb", paData[6]);
gdImageString(im, gdFontSmall, 825-strlen(str)*3, 567, str, white);

/* 7 */
sprintf(str, "%.2f ppb", paData[7]);
gdImageString(im, gdFontSmall, 525-strlen(str)*3, 479, str, white);

/* 8 */
sprintf(str, "[NO ] reacted = %.2f ppbV", paData[8]);
gdImageString(im, gdFontSmall, 600-strlen(str)*3, 580, str, black);
gdImageChar(im, gdFontTiny, 618-strlen(str)*3, 584, '2', black);

/* 9 */
sprintf(str, "P   = %.2f", paData[9]);
gdImageString(im, gdFontSmall, 600-strlen(str)*3, 560, str, black);
gdImageChar(im, gdFontTiny, 606-strlen(str)*3, 564, 'N', black);
gdImageChar(im, gdFontTiny, 612-strlen(str)*3, 564, 'O', black);

/* 10 */
sprintf(str, "%.2f NO cycles", paData[10]);
gdImageString(im, gdFontSmall, 600-strlen(str)*3, 660, str, white);

/* 11 */
sprintf(str, "%.2f ppb", paData[11]);
gdImageString(im, gdFontSmall, 1075-strlen(str)*3, 567, str, white);

/* 12 */
sprintf(str, "%.2f ppb", paData[12]);
y=strlen(str)*3;
gdImageString(im, gdFontSmall, 800-2*y, 750, str, white);
sprintf(str, "[O ] init");
gdImageString(im, gdFontSmall, 772-y, 737, str, white);
gdImageChar(im, gdFontTiny, 784-y, 741, '3', white);

/* 13 */
sprintf(str, "%.2f ppb", paData[13]);
gdImageString(im, gdFontSmall, 925-strlen(str)*3, 704, str, white);

/* 14 */
sprintf(str, "%.2f ppb", paData[14]);
gdImageString(im, gdFontSmall, 1075-strlen(str)*3, 801, str, white);

```

```

/* 15 */
sprintf(str, "%.2f ppb", paData[15]);
gdImageString(im, gdFontSmall, 875-strlen(str)*3, 801, str, white);

/* 16 */
sprintf(str, "%.2f ppb", paData[16]);
gdImageString(im, gdFontSmall, 975-strlen(str)*3, 801, str, white);

/* 17 */
sprintf(str, "%.2f ppb", paData[17]);
y=strlen(str)*3;
gdImageString(im, gdFontSmall, 1155, 750, str, white);
gdImageString(im, gdFontSmall, 1125+y, 737, "[O ] final", white);
gdImageChar(im, gdFontTiny, 1137+y, 741, '3', white);

/* 18 */
sprintf(str, "%.2f ppb", paData[18]);
gdImageString(im, gdFontSmall, 1075-strlen(str)*3, 410, str, white);

/* 19 */
sprintf(str, "%.2f ppb", paData[19]);
gdImageString(im, gdFontSmall, 175-strlen(str)*3, 181, str, white);

/* 20 */
sprintf(str, "%.2f ppb", paData[20]);
gdImageString(im, gdFontSmall, 175-strlen(str)*3, 262, str, white);

/* 21 */
sprintf(str, "%.2f ppb", paData[21]);
gdImageString(im, gdFontSmall, 175-strlen(str)*3, 343, str, white);

/* 22 */
sprintf(str, "%.2f ppb", paData[22]);
gdImageString(im, gdFontSmall, 375-strlen(str)*3, 262, str, white);

/* 23 */
sprintf(str, "%.2f ppb", paData[23]);
gdImageString(im, gdFontSmall, 825-strlen(str)*3, 262, str, white);

/* 24 */
sprintf(str, "[VOC] reacted = %.2f ppbV", paData[24]);
gdImageString(im, gdFontSmall, 600-strlen(str)*3, 280, str, black);

/* 25 */
sprintf(str, "%.2f OH cycles", paData[25]);
gdImageString(im, gdFontSmall, 600-strlen(str)*3, 155, str, white);

/* 26 */
sprintf(str, "P = %.2f", paData[26]);
gdImageString(im, gdFontSmall, 600-strlen(str)*3, 260, str, black);

```

```

gdImageChar(im,gdFontTiny,606-strlen(str)*3,264,'O',black);
gdImageChar(im,gdFontTiny,612-strlen(str)*3,264,'H',black);

/* 27 */
sprintf(str, "%.2f", paData[27]);
gdImageString(im,gdFontSmall,631,314,str,white);

/* 28 */
sprintf(str,"RO + NO      NO + RO; r(dt) = %.2f ppb",
        paData[28]);
y=strlen(str)*3;
gdImageString(im,gdFontSmall,525-y,356,str,black);
gdImageString(im,gdFontSymbol,579-y,356,"->",black);
gdImageChar(im,gdFontTiny,537-y,360,'2',black);
gdImageChar(im,gdFontTiny,609-y,360,'2',black);

/* 29 */
sprintf(str, "%.2f ppb", paData[29]);
y=strlen(str)*3;
gdImageString(im,gdFontSmall,177-2*y,415,str,white);
gdImageString(im,gdFontSmall,135-y,402,"old [O ] aloft",white);
gdImageChar(im,gdFontTiny,171-y,406,'3',white);

/* 30 */
sprintf(str,"O + NO      NO + O ; r(dt) = %.2f ppb",
        paData[30]);
y=strlen(str)*3;
gdImageString(im,gdFontSmall,500-y,406,str,black);
gdImageString(im,gdFontSymbol,548-y,406,"->",black);
gdImageChar(im,gdFontTiny,506-y,410,'3',black);
gdImageChar(im,gdFontTiny,578-y,410,'2',black);
gdImageChar(im,gdFontTiny,608-y,410,'2',black);

/* 31 */
sprintf(str, "%.2f ppb", paData[31]);
gdImageString(im,gdFontSmall,675-strlen(str)*3,479,str,white);

/* 32 */
sprintf(str,"[O ]p/[NO ]h = %.2f",paData[32]);
y=strlen(str)*3;
gdImageString(im,gdFontSmall,900-y,510,str,white);
gdImageChar(im,gdFontSymbol,972-y,510,'n',white);
gdImageChar(im,gdFontTiny,912-y,514,'3',white);
gdImageChar(im,gdFontTiny,954-y,514,'2',white);

/* SAVE the diagram as pa###.gif, where ### corresponds to the plot
number on the map
*/
sprintf(str,"%spa%.3d.gif",fStr[0],smMax);
cfPtr=fopen(str,"wb");
gdImageGif(im, cfPtr);

```

```

        fclose(cfPtr);
        gdImageDestroy(im);
    }
/* END PA DIAGRAM DRAW ROUTINE
*/
    }
/* END IRR PA DIAGRAM CONDITIONAL

    BEGIN IRR PA DIAGRAM ALTERNATIVE
*/
    else
    {
/* Ring Terminal Bell twice and send a message indicating which variable
is out of bounds
*/
        XBell(GuiDpy,50);
        printf("Grid Selection for Process Analysis Exceeds Bounds:\n");
        XBell(GuiDpy,50);
        if((int)(smData[1]+0.01)-(int)(smData[16])<-1)
            printf("Y Grid Size (%d) is Larger than Y Location (%d)\n",
                (int)(smData[16]),(int)(smData[1]+0.01)+1);
        if((int)(smData[0]+0.01)+(int)(smData[15])>smData[10])
            {
                printf("X Location (%d) ",(int)(smData[0]+0.01)+1);
                printf("with X Grid Size (%d) is Greater than X Bound (%d)\n",
                    (int)(smData[15]),(int)smData[10]);
            }
        if((int)(smData[12]+0.01)+(int)(smData[17])>smData[14])
            {
                printf("Z Location (%d) ",(int)(smData[12]+0.01)+1);
                printf("with Z Grid Size (%d) is Greater than Z Bound (%d)\n",
                    (int)(smData[17]),(int)smData[14]);
            }
        if((int)smData[13]+(int)smData[18]>smData[19])
            {
                printf("Current Time Step (%d) ",(int)(smData[13])+1);
                printf("with Time Span (%d) is Greater than Time Bound (%d)\n",
                    (int)(smData[18]),(int)smData[19]);
            }
    }
/* END IRR PA DIAGRAM ALTERNATIVE
*/
    }
/* END SM PLOTTING CONDITIONAL

    RING Terminal Bell once if the sm plot cannot be draw because the
    radial scale is too small or there are small multiple components
    missing
*/
else XBell(GuiDpy,50);
}

```

```

/* END rickman_func_7
*/

/* CREATE the function performed when <F8> is pressed
   Used to save the current map as sm_map.gif and reset for a new map
*/
void rickman_func_8(void)
{
/* DEFINE VARIABLES
   cfPtr = *POINTER generic file pointer for stdio file functions
   im = generic image variable for gd 1.2
   black\
   dgray \
   gray > gd 1.2 color variables
   bgray /
   white/
   str = generic labelling string
*/
FILE *cfPtr;
gdImagePtr im;
int black,dgray,gray,bgray,white;
char str[20];

/* IF all of the small multiple components are available
*/
if (pSc)
{
/* COPY sm_temp.gif into sm_map.gif
*/
sprintf(str,"cp %ssm_temp.gif %ssm_map.gif",fStr[0],fStr[0]);
system (str);

/* RESET the number of plots draw on map to zero
*/
smMax=0;

/* CREATE a new sm_tempfile from the sm_back file
*/
sprintf(str,"%ssm_back.gif",fStr[0]);
cfPtr=fopen(str,"rb");
im=gdImageCreateFromGif(cfPtr);
fclose(cfPtr);

black = gdImageColorExact(im, 0, 0, 0);
white = gdImageColorExact(im, 255, 255, 255);
gray = gdImageColorExact(im, 127, 127, 127);
bgray = gdImageColorAllocate(im, 191, 191, 191);
dgray = gdImageColorAllocate(im, 64, 64, 64);

```

```

sprintf(str,"Radial Scale: -%.1f to +%.1f",smData[5],smData[5]);
gdImageString(im,gdFontSmall,500-strlen(str)*3,725,str,bgray);

sprintf(str,"%ssm_temp.gif",fStr[0]);
cfPtr=fopen(str,"wb");
gdImageGif(im, cfPtr);
fclose(cfPtr);
gdImageDestroy(im);
}

/* IF any one or more of the small multiple components are not available
   ring the Terminal Bell once
*/
else XBell(GuiDpy,50);
}
/* END rickman_func_8
*/

/* CREATE the function performed when <F9> is pressed
   Increment the plotting scale by 10 and reset using <F8>
*/
void rickman_func_9(void)
{
/* ADD 10 to the current radial plotting scale.
   IF adding 10 will make the scale larger than the max scale, set the
   scale to the max
   If the scale is already set to the max, set it to 10
*/
if (smData[5]==backSc) smData[5]=10.;
else smData[5]=smData[5]+10.>backSc?backSc:smData[5]+10.;

/* CALL the <F8> routine to reset with the map with the new scale
*/
rickman_func_8();
}
/* END rickman_func_9
*/

```


APPENDIX 7.1.5 - rickman.h

The following program is written in C. It is the code for the declarations for the modified version of Vis-5D's render.c file.

```
/* include the gd 1.2 include files using #include "path/gd1.2/gd.h"
   & #include "path/gd1.2/gdfonts.h" where path is the directory
   in which the gd1.2 directory is stored
*/
#include "/alpha2/rickman/gd/gd1.2/gd.h"
#include "/alpha2/rickman/gd/gd1.2/gdfonts.h"

/* DEFINE external variables
   pSc \
   smData\
   backSc > assigned in rickmang.h
   fStr /
   vS = pointer array: element # is the small multiple component #
                       : stored value identifies which Vis-5D
                       variable the element # points to
*/
extern short pSc;
extern float smData[],backSc;
extern char fStr[][50];
extern int vS[8]={-1,-1,-1,-1,-1,-1,-1,-1};

/* DECLARE the four functions in rickman.c
*/
void rickman_location_proc(float, float, float);
void rickman_data_write_proc(Context);
void rickman_setup_proc(Context);
void rickman_fail_data_proc(void);
```

APPENDIX 7.1.6 - rickmang.h

The following program is written in C. It is the code for the declarations for the modified version of Vis-5D's gui.c file.

```
/* include the gd 1.2 include files using #include "path/gd1.2/gd.h"
   & #include "path/gd1.2/gdfont$.h" where path is the directory
   in which the gd1.2 directory is stored and $ is s, t, l, & sy
*/
#include "/alpha2/rickman/gd/gd1.2/gd.h"
#include "/alpha2/rickman/gd/gd1.2/gdfontl.h"
#include "/alpha2/rickman/gd/gd1.2/gdfontt.h"
#include "/alpha2/rickman/gd/gd1.2/gdfonts.h"
#include "/alpha2/rickman/gd/gd1.2/gdfontsy.h"

/* CREATE the ABSOLUTE VALUE function
*/
#define ABS(X) ( (X) < 0 ? -(X) : (X) )

/* DEFINE EXTERNAL VARIABLES:
   pSc = Small Multiple flag (TRUE if all components for small
        multiples are available)
   fStr = data source file and path
   smMax = number of small multiple plots currently on the map
   smData = plot drawing variables and constants
           0 = x grid
           1 = y grid
           2 = height
           3 = date
           4 = time
           5 = scale
           6 = DIFF=VDIF+DDEP
           7 = ADVC=HADV+VADV+HDIF
           8 = CHEM
           9 = EMIS
          10 = Max Number of Grids in X (Nc)
          11 = Max Number of Grids in Y (Nr)
          12 = z grid
          13 = t step
          14 = Max Number of Grids in Z (Nl [CHEM])
          15 = Sample Grid Size in X
          16 = Sample Grid Size in Y
          17 = Sample Grid Size in Z
          18 = Sample Time Span
          19 = Max Number of Time Steps
```

```
        backSc = maximum scale possible with current data set
*/
extern short pSc=FALSE;
extern char fStr[2][50]=NULL;
extern int smMax=0;
extern float smData[20]={0.},backSc=10.;

/* DECLARE the four functions in rickmang.c
*/
void rickman_func_6(void);
void rickman_func_7(void);
void rickman_func_8(void);
void rickman_func_9(void);
```

APPENDIX 7.1.7 - patemplate.c

The following program is written in C. It is the source code for a program which creates the gif templates used in the modified version of Vis-5D referenced in this document.

```
/* This file creates two template gif files for use with the Vis-5D
   modifications to produce small multiple displays.

INCLUDE the standard input/output library and the gd 1.2 font libraries
*/
#include <stdio.h>
#include "/alpha2/rickman/gd/gd1.2/gd.h"
#include "/alpha2/rickman/gd/gd1.2/gdfonts.h"
#include "/alpha2/rickman/gd/gd1.2/gdfontt.h"
#include "/alpha2/rickman/gd/gd1.2/gdfontsy.h"

/* DEFINE VARIABLES
   red \
   black\
   bgray > = generic gd 1.2 color variables
   white/
   gray/
   cfPtr = *POINTER variable for standard file processing
   im = generic gd 1.2 image variable
   brush = generic gd 1.2 brush image variable
   points = generic gd 1.2 vertex array
   str = generic character string variable
*/
int black,red,gray,bgray,white;
FILE *cfPtr;
gdImagePtr im, brush;
gdPoint points[4];
char str[50];

main(){
/*CREATE the right arrow brush and save as ar_rt.gif
*/
im=gdImageCreate(10,10);
black = gdImageColorAllocate(im, 0, 0, 0);
white = gdImageColorAllocate(im, 255, 255, 255);
points[0].x=1;
points[0].y=1;
points[1].x=9;
points[1].y=5;
```

```

points[2].x=1;
points[2].y=9;
gdImageFilledPolygon(im,points,3,white);
gdImageColorTransparent(im,black);
cfPtr=fopen("ar_rt.gif","wb");
gdImageGif(im, cfPtr);
fclose(cfPtr);
gdImageDestroy(im);

/*CREATE the up arrow brush and save as ar_up.gif
*/
im=gdImageCreate(10,10);
black = gdImageColorAllocate(im, 0, 0, 0);
white = gdImageColorAllocate(im, 255, 255, 255);
points[0].x=1;
points[0].y=9;
points[1].x=5;
points[1].y=1;
points[2].x=9;
points[2].y=9;
gdImageFilledPolygon(im,points,3,white);
gdImageColorTransparent(im,black);
cfPtr=fopen("ar_up.gif","wb");
gdImageGif(im, cfPtr);
fclose(cfPtr);
gdImageDestroy(im);

/*CREATE the down arrow brush and save as ar_dn.gif
*/
im=gdImageCreate(10,10);
black = gdImageColorAllocate(im, 0, 0, 0);
white = gdImageColorAllocate(im, 255, 255, 255);
points[0].x=1;
points[0].y=1;
points[1].x=5;
points[1].y=9;
points[2].x=9;
points[2].y=1;
gdImageFilledPolygon(im,points,3,white);
gdImageColorTransparent(im,black);
cfPtr=fopen("ar_dn.gif","wb");
gdImageGif(im, cfPtr);
fclose(cfPtr);
gdImageDestroy(im);

/* CREATE the Process Analysis Cycle Schematic Template
*/
im=gdImageCreate(1250,850);

black = gdImageColorAllocate(im, 0, 0, 0);
red = gdImageColorAllocate(im, 255, 0, 0);

```

```
gray = gdImageColorAllocate(im, 127, 127, 127);
bgray = gdImageColorAllocate(im, 128, 128, 128);
white = gdImageColorAllocate(im, 255, 255, 255);
```

```
points[0].x=475;
points[0].y=225;
points[1].x=725;
points[1].y=225;
points[2].x=725;
points[2].y=300;
points[3].x=475;
points[3].y=300;
```

```
gdImageFilledPolygon(im,points,4,white);
```

```
points[0].x=475;
points[0].y=525;
points[1].x=725;
points[1].y=525;
points[2].x=725;
points[2].y=600;
points[3].x=475;
points[3].y=600;
```

```
gdImageFilledPolygon(im,points,4,white);
```

```
points[0].x=325;
points[0].y=350;
points[1].x=725;
points[1].y=350;
points[2].x=725;
points[2].y=375;
points[3].x=325;
points[3].y=375;
```

```
gdImageFilledPolygon(im,points,4,white);
```

```
points[0].x=325;
points[0].y=400;
points[1].x=675;
points[1].y=400;
points[2].x=675;
points[2].y=425;
points[3].x=325;
points[3].y=425;
```

```
gdImageFilledPolygon(im,points,4,white);
```

```
gdImageLine(im,175,125,1075,125,white);
gdImageLine(im,375,175,825,175,white);
gdImageLine(im,375,650,825,650,white);
```

```

gdImageLine(im,75,625,275,625,white);
gdImageLine(im,825,750,1125,750,white);
gdImageLine(im,225,262,325,262,white);
gdImageLine(im,425,262,475,262,white);
gdImageLine(im,725,262,775,262,white);
gdImageLine(im,225,412,325,412,white);
gdImageLine(im,225,562,325,562,white);
gdImageLine(im,425,562,475,562,white);
gdImageLine(im,725,562,775,562,white);
gdImageLine(im,875,562,1025,562,white);

gdImageLine(im,175,125,175,162,white);
gdImageLine(im,175,200,175,237,white);
gdImageLine(im,175,287,175,324,white);
gdImageLine(im,75,625,75,675,white);
gdImageLine(im,275,625,275,675,white);
gdImageLine(im,175,600,175,675,white);
gdImageLine(im,375,175,375,237,white);
gdImageLine(im,375,650,375,600,white);
gdImageLine(im,825,237,825,175,white);
gdImageLine(im,825,600,825,650,white);
gdImageLine(im,625,300,625,350,white);
gdImageLine(im,700,375,700,450,white);
gdImageLine(im,650,425,650,450,white);
gdImageLine(im,675,525,675,500,white);
gdImageLine(im,525,525,525,500,white);
gdImageLine(im,1075,125,1075,387,white);
gdImageLine(im,1075,550,1075,425,white);
gdImageLine(im,1075,775,1075,587,white);
gdImageLine(im,875,775,875,750,white);
gdImageLine(im,975,775,975,750,white);
gdImageLine(im,925,725,925,750,white);

/* USE ar_rt.gif image to make arrow heads on the lines in the schematic
*/
cfPtr=fopen("ar_rt.gif","rb");
brush=gdImageCreateFromGif(cfPtr);
fclose(cfPtr);
gdImageSetBrush(im,brush);

gdImageLine(im,321,262,321,262,gdBrushed);
gdImageLine(im,471,262,471,262,gdBrushed);
gdImageLine(im,771,262,771,262,gdBrushed);
gdImageLine(im,321,412,321,412,gdBrushed);
gdImageLine(im,321,562,321,562,gdBrushed);
gdImageLine(im,471,562,471,562,gdBrushed);
gdImageLine(im,771,562,771,562,gdBrushed);
gdImageLine(im,1021,562,1021,562,gdBrushed);
gdImageLine(im,1121,750,1121,750,gdBrushed);

gdImageDestroy(brush);

```

```

/* USE ar_up.gif image to make arrow heads on the lines in the schematic
*/
cfPtr=fopen("ar_up.gif","rb");
brush=gdImageCreateFromGif(cfPtr);
fclose(cfPtr);
gdImageSetBrush(im,brush);

gdImageLine(im,175,291,175,291,gdBrushed);
gdImageLine(im,175,604,175,604,gdBrushed);
gdImageLine(im,375,604,375,604,gdBrushed);
gdImageLine(im,1075,429,1075,429,gdBrushed);
gdImageLine(im,1075,754,1075,754,gdBrushed);
gdImageLine(im,875,754,875,754,gdBrushed);

gdImageDestroy(brush);

/* USE ar_dn.gif image to make arrow heads on the lines in the schematic
*/
cfPtr=fopen("ar_dn.gif","rb");
brush=gdImageCreateFromGif(cfPtr);
fclose(cfPtr);
gdImageSetBrush(im,brush);

gdImageLine(im,175,158,175,158,gdBrushed);
gdImageLine(im,175,233,175,233,gdBrushed);
gdImageLine(im,375,233,375,233,gdBrushed);
gdImageLine(im,625,346,625,346,gdBrushed);
gdImageLine(im,650,446,650,446,gdBrushed);
gdImageLine(im,700,446,700,446,gdBrushed);
gdImageLine(im,675,521,675,521,gdBrushed);
gdImageLine(im,525,521,525,521,gdBrushed);
gdImageLine(im,925,746,925,746,gdBrushed);
gdImageLine(im,1075,746,1075,746,gdBrushed);
gdImageLine(im,975,771,975,771,gdBrushed);

gdImageDestroy(brush);

/* LABEL the schematic
*/
sprintf(str,"[NO] from");
gdImageString(im,gdFontSmall,48,687,str,white);
gdImageString(im,gdFontSmall,148,687,str,white);
sprintf(str,"new NO");
gdImageString(im,gdFontSmall,57,700,str,white);
gdImageString(im,gdFontSmall,154,700,str,white);
gdImageChar(im,gdFontTiny,190,704,'2',white);
sprintf(str,"[NO] bal");
gdImageString(im,gdFontSmall,251,687,str,white);
sprintf(str,"new [NO]");
gdImageString(im,gdFontSmall,151,554,str,white);
sprintf(str,"[NO] reacted");

```



```

gdImageString(im,gdFontSmall,339,554,str,white);
sprintf(str,"[NO] recreated");
gdImageString(im,gdFontSmall,783,554,str,white);
sprintf(str,"[O] produced");
gdImageString(im,gdFontSmall,1036,554,str,white);
gdImageChar(im,gdFontTiny,1048,558,'3',white);
sprintf(str,"[NO] from Other Paths");
gdImageString(im,gdFontSmall,459,466,str,white);
gdImageChar(im,gdFontTiny,477,470,'2',white);
sprintf(str,"Propagation and Termination");
gdImageString(im,gdFontSmall,519,535,str,black);
sprintf(str,"[O] h trans");
gdImageString(im,gdFontSmall,889,691,str,white);
gdImageChar(im,gdFontTiny,901,695,'3',white);
sprintf(str,"[O] reacted");
gdImageString(im,gdFontSmall,1039,788,str,white);
gdImageChar(im,gdFontTiny,1051,792,'3',white);
sprintf(str,"[O] v trans");
gdImageString(im,gdFontSmall,839,788,str,white);
gdImageChar(im,gdFontTiny,851,792,'3',white);
sprintf(str,"[O] depo");
gdImageString(im,gdFontSmall,948,788,str,white);
gdImageChar(im,gdFontTiny,960,792,'3',white);
sprintf(str,"+H O");
gdImageString(im,gdFontSmall,1063,397,str,white);
gdImageChar(im,gdFontTiny,1075,401,'2',white);
sprintf(str,"O + h");
gdImageString(im,gdFontSmall,154,168,str,white);
gdImageChar(im,gdFontSymbol,190,168,'n',white);
gdImageChar(im,gdFontTiny,160,172,'3',white);
sprintf(str,"new [OH]");
gdImageString(im,gdFontSmall,151,249,str,white);
sprintf(str,"aldehydes + h");
gdImageString(im,gdFontSmall,133,330,str,white);
gdImageChar(im,gdFontSymbol,210,330,'n',white);
sprintf(str,"[OH] reacted");
gdImageString(im,gdFontSmall,339,249,str,white);
sprintf(str,"[OH] recreated");
gdImageString(im,gdFontSmall,783,249,str,white);
sprintf(str,"Propagation and Termination");
gdImageString(im,gdFontSmall,519,235,str,black);
sprintf(str,"(NO NO)/VOC =");
gdImageString(im,gdFontSmall,523,314,str,white);
sprintf(str,"->");
gdImageString(im,gdFontSymbol,547,314,str,white);
gdImageChar(im,gdFontTiny,577,318,'2',white);
sprintf(str,"NO");
gdImageString(im,gdFontSmall,644,466,str,white);
gdImageString(im,gdFontSmall,691,466,str,white);
gdImageChar(im,gdFontTiny,703,470,'2',white);
sprintf(str,"->");

```

```

gdImageString(im,gdFontSymbol,669,466,str,white);

/* SAVE the template as patemplate.gif
*/
cfPtr=fopen("patemplate.gif","wb");
gdImageGif(im, cfPtr);
fclose(cfPtr);
gdImageDestroy(im);

/* CREATE the Small Multiple Plotting Map Template
*/
im=gdImageCreate(1000,750);
black = gdImageColorAllocate(im, 0, 0, 0);
white = gdImageColorAllocate(im, 255, 255, 255);
gray = gdImageColorAllocate(im, 127, 127, 127);
gdImageLine(im,895.,55.,995.,55.,gray);
gdImageLine(im,945.,5.,945.,105.,gray);
gdImageArc(im,945,55,52,52,0,360,gray);
sprintf(str,"ADVC");
gdImageString(im,gdFontSmall,960,58,str,white);
sprintf(str,"EMIS");
gdImageString(im,gdFontSmall,910,42,str,white);
sprintf(str,"CHEM");
gdImageStringUp(im,gdFontSmall,932,90,str,white);
sprintf(str,"DIFF");
gdImageStringUp(im,gdFontSmall,948,40,str,white);

/* SAVE the template as smtemplate.gif
*/
cfPtr=fopen("sm_template.gif","wb");
gdImageGif(im, cfPtr);
fclose(cfPtr);
gdImageDestroy(im);

return 0;
}

```

APPENDIX 7.2 - DATA CONVERSION

This appendix contains the steps necessary to convert the MAQSIP data in the case studies into the Vis-5D data format.

1. Required Files:

- a. M3Subset - EPA Vis Lab's Models-3 Data extracting/subsetting routine
- b. ncdump - NetCDF Data information extraction routine
- c. m3tov5d - EPA Vis Lab's Models-3 format to Vis-5D format converter
- d. v5dinfo - Vis-5D's data information utility (in the vis5d-4.2 directory)
- e. v5dedit - Vis-5D's data file parameter editor (in the vis5d-4.2 directory)
- f. v5dimport - Vis-5D's data combiner and sorter (in the vis5d-4.2 directory)
- g. Three MAQSIP source data files (*filename* will refer to these files later)
 - (1) Process Analysis output (*sfilepa*)
 - (2) u and v wind component output (*sfileuv*)
 - (3) Ozone Concentration output (*sfilecc*)

2. Extract necessary data from the three source files into target files (*tfilepa*, *tfileuv*, and *tfilecc*):

a. `<M3Subset sfilepa tfilepa -layer 1 6 -variable O3_CHEM O3_HADV O3_VADV O3_HDIF O3_VDIF O3_DDEP O3_EMIS O3_CLDP>`

b. `<M3Subset sfilecc tfilecc -layer 1 6 -variable O3>`

c. Determine the number of rows and columns in the wind source data:

(1) `<ncdump -h sfileuv | grep NCOLS>`

(2) Note the numerical value of NCOLS (This value will be referred to as **NC**, in later steps).

(3) `<ncdump -h sfileuv | grep NROWS>`

(4) Note the numerical value of NROWS (This value will be referred to as **NR**, in later steps).

d. Subtract one (1) from both **NR** and **NC**. (**NR=NR-1** & **NC=NC-1**)

f. `<M3Subset sfileuv tfileuv -layer 1 6 -variable UWIND VWIND -row 1 NR -column 1 NC>`

where **NR** is replaced with the value of **NR**, and **NC** is replaced with the value of **NC**.

3. Identify the values of the sigma levels:

a. `<ncdump tfilepa | grep VGLVLS>`

b. Step a, above, will list 7 values. If the seven values are, in order, $\sigma [0]$, $\sigma [1]$, ... , $\sigma [6]$, then you can calculate the approximate heights using

the following formula:
$$Z[n] = \left(1 - \frac{(\sigma [n] + \sigma [n+1])}{2} \right) \times 8 \quad \forall \quad n = 1 \dots 6.$$

4. Convert the MAQSIP data from Models-3 format to Vis-5D format:

a. `<m3tov5d tfilepa vfilepa>`

b. `<m3tov5d tfilecc vfilecc>`

c. `<m3tov5d tfileuv vfileuv>`

5. Display the projection data (`<v5dinfo vfilepa>`). Note that you must either look at the PA data file or the concentration data file.

6. Adjust the wind data file's projection parameters to match the others (`<v5dedit vfileuv>`):

(1) Menu Item 1: Change the variable names from "UWIND" to "U" and "VWIND" to "V".

(2) Menu Item 2 (optional): Enter the units of the wind speed, "m/s".

(3) Menu Item 4: Change each of the parameters so that they match those that you displayed in step 5 above.

(4) Menu Item 5: Change the system to "Generic linear, equally spaced", set the bottom bound to zero (0), and set the increment to one (1).

(5) Exit and save the changes.

7. Adjust the vertical coordinates of the concentration and pa data files (<v5dedit *vfilecc*> and <v5dedit *vfilepa*>):

(1) Menu Item 2 (optional): Enter the units of the concentration, “ppmV”, or the process rate, “ppbV/h”, as appropriate.

(2) Menu Item 5: Change the system to “Generic linear, equally spaced”, set the bottom bound to zero (0), and set the increment to one (1).

(3) Exit and save the changes.

8. Combine the data into one file (<v5dimport>):

a. Read in each of the three source files (*vfilepa*, *vfilecc*, and *vfileuw*).

b. Select all of the grids and save as *datafile*.

c. Exit the program.

9. Adjust the vertical coordinates of the data files (<v5dedit *datafile*>):

(a) Menu Item 4: Change the values of “Column of the North/South Pole” and “Row of the North/South Pole” by adding +0.5 to each value.

(b) Menu Item 5: Change the system to “Linear, unequally spaced km”, and set each of the levels to the values you calculated in step 3 above.

(c) Exit and save the changes.

APPENDIX 7.3 - USER'S GUIDE

This appendix describes how to use the modifications to Vis-5D made in the course of this research.

1. Data requirements for small multiple functionality. Must have process analysis data fields in the Vis-5D data file labeled with each of the following names: CHEM, HADV, VADV, EMIS, DDEP, VDIF, and HDIF. The labels for these fields may have any number of characters preceding the four character names. However, if any one of these seven fields is missing, or the four character names are not written in all capital letters, the small multiple function will produce a message in the sm.gif file which identifies which field(s) is(are) missing or mislabeled.
2. Producing independent small multiple, diamond plots. If the data requirements in 1 above are met, when the Vis-5D data probe is activated, the computer will produce the independent small multiple, diamond plot in the sm.gif file. The computer will place the file in the directory in which the Vis-5D data file, entered in the command line, is located. Every time the user moves the probe or advances/retrogresses the timestep the computer updates the sm.gif file.
3. Changing the scale of the independent plot. The radial scale on the independent plot will always default to one of two values. If the scale size

selected by the user, initially set to 50 ppbv/h, is large enough to accommodate the magnitudes of all four contributions, the computer will use that scale. However, if the magnitude of any one of the four contributions exceeds either the positive or negative limit of the scale, the computer will use the maximum scale size available for the data set. The user can change the selectable scale by pressing <F9>. The scale cycles up in increments of 10 ppbv/h until it reaches the maximum scale. The next time the user presses <F9>, the scale recycles to 10 ppbv/h. The computer will not redraw the independent plot, however, until the user moves the probe or increments the time step.

3. Small multiple map output. Once the user has positioned the probe over the location of interest, press <F7>. If the user has not activated the probe yet, the terminal bell will ring once. Also, if the user selected scale is too small to accommodate the magnitude of any of the four contributions, the terminal bell will ring. The computer will plot the small multiple on the map centered on the probe's location. The plot will be numbered in the upper left corner. It will have a date/time label in the upper right corner, and will have the height indicated in the lower right corner. The computer will save the map file as sm_temp.gif in the directory in which the Vis-5D data file, entered in the command line, is located.

4. Saving/Resetting map files. By pressing <F8>, the user will copy the current copy of sm_temp.gif to sm_map.gif. At the same time, the computer will clear the sm_temp.gif file and reset the plot identification number (number which appears in upper left corner of small multiple plots) to 1.
5. Changing the scale on the small multiple map plots. By pressing <F9>, the user will increment the current radial scale setting by 10 ppbv/h. If incrementing will increase the scale beyond the maximum size determined for that data set, the computer will set the scale for the maximum size. If the scale is currently set to the maximum size, the computer will change the scale to 10 ppbv/h. Pressing <F9> will also save the current version of sm_temp.gif as sm_map.gif, clear the sm_temp.gif file, and reset the plotting increment to 1, just like <F8>.
6. Data requirements for Hydroxyl Radical and Nitric Oxide Oxidation Cycle schematic functionality. To create the schematics, the user must meet the following data requirements:
 - a. The post-processor program must be in the *V5Dparent/vis5d-4.2* directory, and the program must be called "PA_out.exe".
 - b. The source data file for the post-processor must be stored in the directory in which the Vis-5D data file, entered in the command line, is located. It must have the same file name as the data file plus a ".pa" extension. Example:

Vis-5D Data File: datafile.v5d

Post-Processor Input Data File: datafile.v5d.pa

c. The first thirty-two (32) lines of the post-processor output file must be in the format of 32 numeric values corresponding to the 32 measurements, one value per line, with no blank lines in between. The file must have the following file name: pa.dat. The file must be in the *V5Dparent/vis5d-4.2* directory.

7. Creating cycle schematics. When the user presses <F7> to plot the small multiple diamond plot for the cursor location on the map, the computer assumes that the user has placed the cursor in the lower, left, bottom corner of the desired sample grid starting from the current timestep, invokes the post-processor with the following command:

PA_out.exe filedirectory\datafile.v5d.pa srow nrow scol ncol slev nlev stime ntime

where *filedirectory* is the directory where the Vis-5D data file is located

datafile.v5d.pa is the post-processor source data file

srow is the starting row of the sample grid (NOTE: this is the row number in MAQSIP navigation not in Vis-5D navigation)

nrow is the number rows to sample

scol is the starting column of the sample grid

<i>ncol</i>	is the number of columns to sample
<i>slev</i>	is the starting vertical level of the sample grid
<i>nlev</i>	is the number of levels to sample
<i>stime</i>	is the starting timestep of the sample grid
<i>ntime</i>	is the number of timesteps to sample

If the current location of the probe plus the sample size exceeds the bounds of the data set, the terminal bell will ring twice. If the bounds are satisfactory, the computer will send the command to the post-processor and await its completion. If the post-processor fails to produce a pa.dat file, the command window will report that error. If the post-processor succeeds and the computer can produce the schematic, the computer will store the schematic as PA###.gif in the directory in which the Vis-5D data file, entered in the command line, is located. The ### will correspond to the plot number identified on the small multiple map (for example, plot 1 will correspond to PA001.gif, plot 2 to PA002.gif, and so on).

8. Changing the sample grid size for the cycle schematic post-processor. By pressing <F6>, the user activates a menu driven program to change the number of rows, columns, levels, and timesteps sent to the post-processor in step 7 above. Initial values are as follows:

<i>nrow</i> = 1	<i>nlev</i> = 1
<i>ncol</i> = 1	<i>ntime</i> = 8