

Naval Research Laboratory

Stennis Space Center, MS 39529-5004



NRL/FR/7441--96-9654

Virtual World Reconstruction Using the Modeling and Simulation Extended Vector Product Prototype (Project Sponsored by Defense Modeling and Simulation Office)

KEVIN SHAW

*Mapping, Charting, and Geodesy Branch
Marine Geosciences Division*

MAHDI ABDELGUERFI
EDGAR COOPER
CHRIST WYNNE

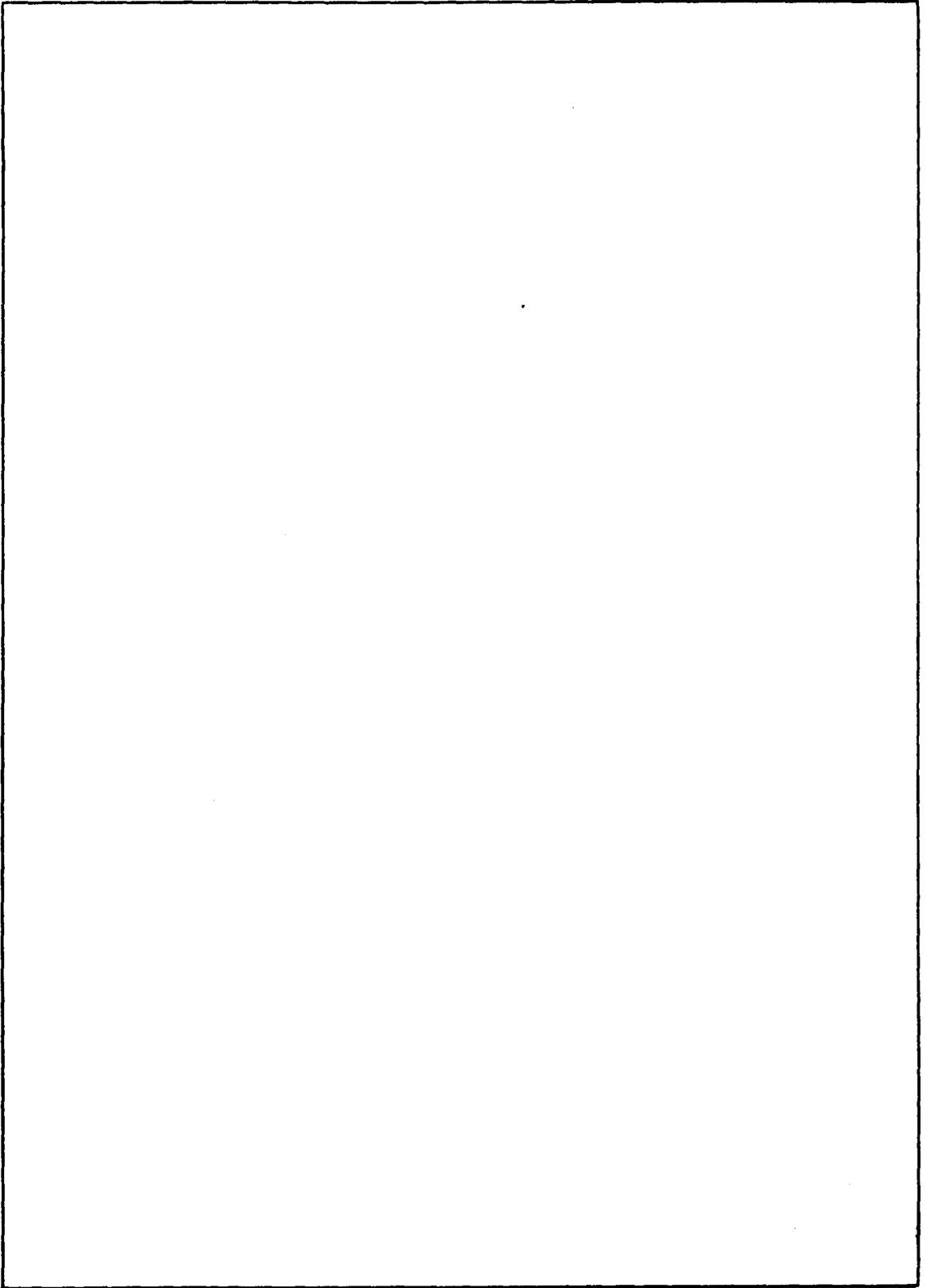
*University of New Orleans
New Orleans, LA*

H. VINCENT MILLER
BARBARA RAY
ROBERT BROOME
TODD LOVITT

*Planning Systems Incorporated
Slidell, LA*

September 30, 1997

Approved for public release; distribution unlimited.



REPORT DOCUMENTATION PAGE

Form Approved
OBM 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 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 September 30, 1997	3. REPORT TYPE AND DATES COVERED Final	
4. TITLE AND SUBTITLE Virtual World Reconstruction Using the Modeling and Simulation Extended Vector Product Prototype (Project Sponsored by Defense Modeling and Simulation Office)		5. FUNDING NUMBERS Job Order No. 574590806 Program Element No. RDT&EDA Project No. Task No. Accession No. DN153251	
6. AUTHOR(S) Kevin Shaw, Mahdi Abdelguerfi*, Edgar Cooper*, Christ Wynne*, H. Vincent Miller†, Barbara Ray†, Robert Broome†, and Todd Lovitt†		8. PERFORMING ORGANIZATION REPORT NUMBER NRL/FR/7441--96-9654	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Research Laboratory Marine Geosciences Division Stennis Space Center, MS 39529-5004		10. SPONSORING/MONITORING AGENCY REPORT NUMBER	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Naval Research Laboratory Marine Geosciences Division Stennis Space Center, MS 39529-5004		11. SUPPLEMENTARY NOTES *University of New Orleans, New Orleans, LA; †Planning Systems Incorporated, 115 Christian Lane, Slidell, LA	
12a. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution unlimited.		12b. DISTRIBUTION CODE	
13. ABSTRACT (Maximum 200 words) <p>The modeling and simulation (M&S) communities for both the Navy and Marine Corps are not currently satisfied by the data provided by the National Imagery and Mapping Agency. In particular, deficiencies exist in both the lack of a continuous surface representation and an inconsistent view of elevation throughout a single database. The M&S Extended Vector Product (MSEVP) prototype being developed is an extended vector product format-based product containing a continuous surface representation and a consistent view of elevation across the thematic coverages contained within a database.</p> <p>A continuous surface representation is provided in the MSEVP prototype as a Triangulated Irregular Network (TIN) and stored in the MSEVP elevation coverage. This report also examines how a TIN representation of the transportation network is constructed from a two-dimensional linear representation. Each linear road is widened to provide a more realistic representation of its real work counterpart and then overlaid over the elevation TIN to ensure that the road adheres to the surface characterized by the elevation coverage. This overlay process ensures that a consistent view of elevation will exist across coverages.</p>			
14. SUBJECT TERMS digital MC&G, performance analysis, modeling and simulation		15. NUMBER OF PAGES 23	
		16. PRICE CODE	
17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT Same as report

INTRODUCTION AND OVERVIEW

This Report covers recent NRL research and development related to the AEC laser-pellet fusion program. The work covered in this report was primarily carried out under AEC Contract No. AT(04-3)-878 between July, 1973 and June, 1974. In some cases, for the sake of clarity, work not performed during this period has been included. In addition, the report discusses the large complementary program supported by the Defense Nuclear Agency at NRL on the study of laser produced plasmas in high atomic number materials. A significant part of the soft x-ray experimental studies (Chapter V) and all of the numerical modelling studies (Chapter VIII and Chapter IX) were supported by the DNA program. This work has been included in the present report for the sake of completeness and since many of the results obtained are of direct benefit to the AEC laser-fusion effort.

There is some overlap in time, with the period covered by the last NRL report, which was presented to the Laser-Fusion Coordinating Committee in November, 1973. However, it was felt useful to establish the precedent of fiscal year reporting starting with this Annual Report. In the future, a report on the AEC supported laser-fusion studies at NRL will appear on a semi-annual basis.

Studies of laser produced plasmas began at NRL in 1968. Development of a large Q-switched glass laser facility was carried out under ARPA sponsorship. Additional support for laser development was received as part of a DNA-supported program to study the interactions between an expanding laser-produced plasma and an ambient magnetized plasma background. The DNA program also provided NRL with the genesis of its expertise in the study of laser plasma generation and interactions. The glass laser system was developed under joint ARPA and DNA sponsorship to the point where, by FY 72, it represented the state-of-the-art in high power short pulse lasers. The system included an NRL designed mode-locked YAG oscillator system and a disc amplifier which became a prototype for those now in use and under development in the laser-fusion program.

When a large AEC supported laser-fusion effort began in FY 72, NRL was a logical place to turn to for support in building up the programs of the AEC laboratories. Shortly after the start of AEC-supported work at NRL, a DNA program at NRL was initiated to study laser produced high atomic number plasmas. The aim of this program was to develop a new source of soft x-rays for weapons effect simulation. The combined AEC and DNA supported programs have been known within NRL as the Laser-Matter Interaction (LMI) Program and Dr. John Stamper has served as Program Manager since its inception. The LMI Program has included contributions from a number of divisions at NRL and has involved both experimental and theoretical studies. The funding of the program during FY 74 consisted of \$500,000 from the AEC for laser-fusion studies as well as additional funding from DNA for x-ray source investigations.

One of the unique features of the NRL laser-fusion program has been the availability of a reliable Nd:glass laser system which produces exceptionally clean pulses (i.e., spatially, temporally and spectrally pure) at sufficient power levels ($\lesssim 0.5$ TW) for many studies of interest to laser fusion. Over 1500 laser shots were put on target in FY 74 for our AEC

CONTENTS

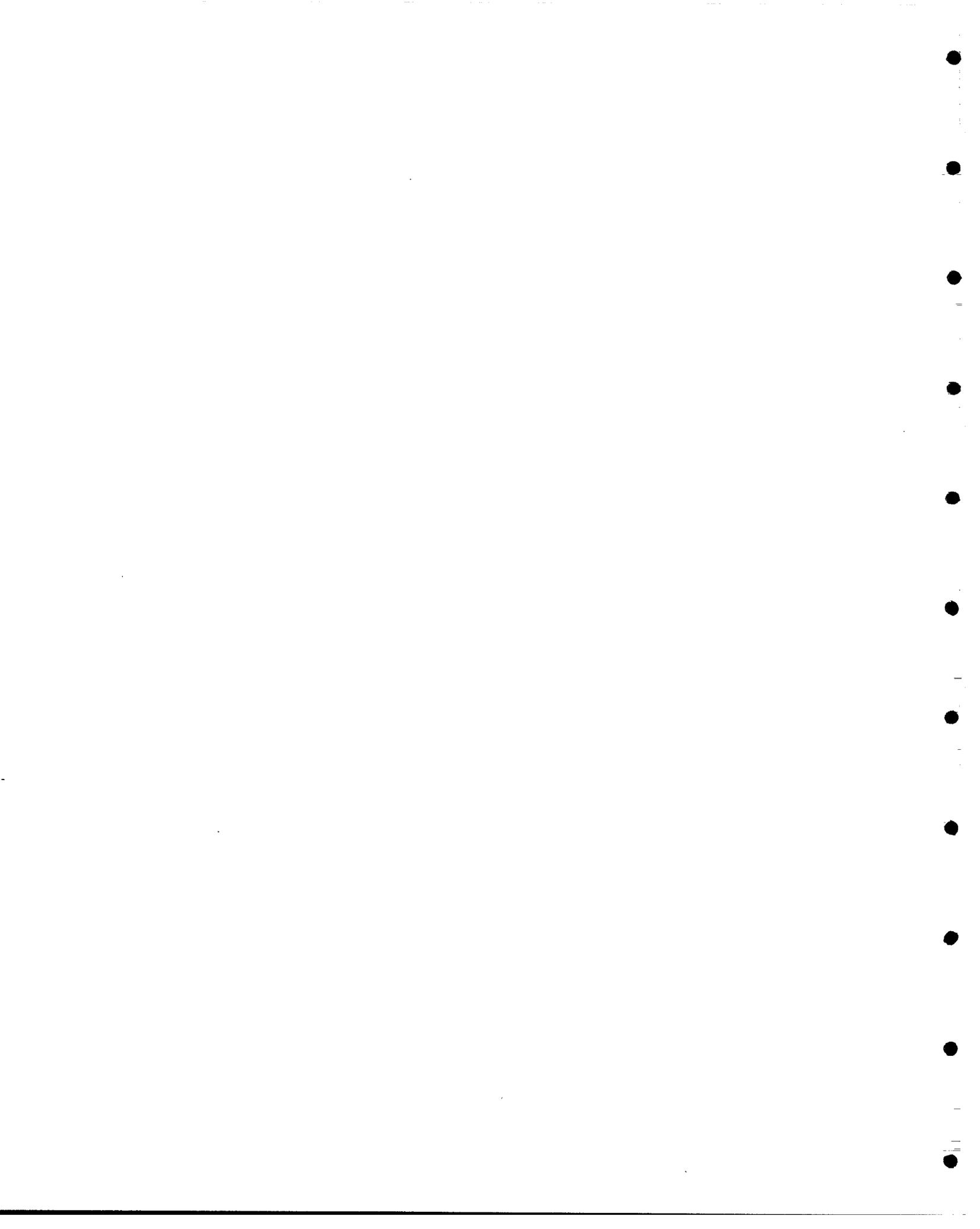
EXECUTIVE SUMMARY	E-1
1.0 INTRODUCTION	1
2.0 EXTENDED VECTOR PRODUCT FORMAT	2
2.1 EVPF Primitive Table Definitions	2
2.2 EVPF Coverage Level Tables	4
2.3 TINs in Tiled Coverages	5
2.4 Modeling and Simulation Extended Vector Product	6
3.0 POPULATING MSEVP TABLES	7
3.1 Generation of Elevation TIN	7
3.2 Generation of Road Polygons	8
3.3 Polygon to TIN Conversion	13
4.0 CONCLUSION AND RECOMMENDATIONS	18
5.0 ACKNOWLEDGMENTS	19
6.0 REFERENCES	19

5.0	ADDITIONAL SNR GAIN FOR THE WISPR AND AWSUM ₄ FILTERS USING STANDARD DEVIATION OF THE TIME SERIES: THE WISPR II _k AND AWSUM ₄ II _k FILTERS	61
5.1	Introduction	61
5.2	Signal Gain Improvement Rationale for the AVGPR Processor	61
5.3	Signal Gain Improvement for the WISPR Filter: WISPR II _k	63
5.4	Signal Gain Improvement for the AWSUM ₄ Filter: AWSUM ₄ II _k	63
5.5	Gains Achieved for WISPR II _k Processing	63
5.6	Removal of Clutter Tonals with WISPR II ₄ Processing	67
5.7	Comparison of WISPR II ₄ Performance with WISPR and AWSUM ₄ for Specific Signals	67
5.8	Summary	70
6.0	TYPICAL WISPR FILTER APPLICATIONS	71
6.1	Introduction	71
6.2	Robustness to Harsh Transients and Intermittent High-Level Noise	71
6.3	Improvement in Spatial Resolution – A Substitute for Increased Array Aperture	74
6.4	Spectral/Spatial Resolution Enhancement and Compensation for Array Misalignments and Instabilities	78
6.5	Summary	85
7.0	ADVANCED WISPR FILTER APPLICATIONS	86
7.1	Introduction	86
7.2	Multifrequency Matched-Field Processing	86
7.3	WISPR Applied to Frequency Domain Matched Filtering	89
7.4	WISPR and the Active Barrier	90
7.5	WISPR Time History (Gram) Performance/Robustness to Masking	94
7.6	Noise Spectrum Equalization (NSE) via Fluctuation-Based Processing to Enhance Submerged Source Detection and Classification	101
7.7	Summary	109
8.0	WISPR FILTER COHERENT PROCESSING APPLICATIONS, WISPR III _k AND A _k CSDM	109
8.1	Introduction	109
8.2	The WISPR III _k Processor	109
8.3	The AWSUM Cross-Spectral Density Matrix, A _k CSDM Processor, and the WISPR Cross-Spectral Density Matrix, WCSDM = A ₁ CSDM Processor	112
9.0	COMBINING THE WISPR FILTER WITH OTHER HIGH-RESOLUTION PROCESSORS	117
9.1	Introduction	117
9.2	High Gain and Resolution with the Combined WB ² and WISPR Processors	117
9.3	High Gain and Resolution with the Combined DIET and WISPR Processors	119

EXECUTIVE SUMMARY

The modeling and simulation (M&S) communities for both the Navy and Marine Corps are not currently satisfied by the data provided by the National Imagery and Mapping Agency. In particular, deficiencies exist in both the lack of a continuous surface representation and an inconsistent view of elevation throughout a single database. The M&S Extended Vector Product (MSEVP) prototype being developed is an extended vector product format-based product containing a continuous surface representation and a consistent view of elevation across the thematic coverages contained within a database.

A continuous surface representation is provided in the MSEVP prototype as a Triangulated Irregular Network (TIN) and stored in the MSEVP elevation coverage. This report also examines how a TIN representation of the transportation network is constructed from a two-dimensional linear representation. Each linear road is widened to provide a more realistic representation of its real work counterpart and then overlaid over the elevation TIN to ensure that the road adheres to the surface characterized by the elevation coverage. This overlay process ensures that a consistent view of elevation will exist across coverages.



**VIRTUAL WORLD RECONSTRUCTION USING THE MODELING AND
SIMULATION EXTENDED VECTOR PRODUCT PROTOTYPE
(PROJECT SPONSORED BY DEFENSE MODELING AND SIMULATION OFFICE)**

1.0 INTRODUCTION

The National Imagery and Mapping Agency (NIMA) holds the responsibility of providing large geographic data sets to a wide range of users. Among the users receiving data from the NIMA are the modeling and simulation (M&S) communities for both the Navy and Marine Corps. This report focuses on providing solutions via an Extended Vector Product Format (EVVPF) to the deficiencies of the NIMA's Vector Product Format (VPF) as they relate to the Navy and Marine Corps Digital Mapping, Charting, and Geodesy Requirements for M&S (Shaw et al. 1996).

VPF is a military standard data format (Mil-Std-2407) that serves as a foundation from which database products are designed using its georelational data model (Department of Defense 1993). The data model defines geographic entities as a collection of node, edge, and/or face primitives. Features and their accompanying attribute data are then linked to the appropriate primitives. Individual products are constructed using the VPF data model to contain different sets of feature classes and attribution. Among the current NIMA products are Digital Nautical Chart (DNC), Urban Vector Smart Map, and Digital Topographic Data (DTOP). DNC contains selected marine-significant physical features collected from harbor, approach, coastal, and general charts (Defense Mapping Agency 1993).

Products based on the VPF standard, including those above, have insufficient feature and attribution content to completely fulfill the requirements of the M&S community (Shaw et al. 1995). Attribute data such as radar reflectivity and infrared signatures are among those not currently provided by the NIMA (Shaw et al. 1995). The level of resolution and accuracy of the terrain and integrated surface feature classes must also be increased to meet M&S requirements. The problems with VPF extend beyond content to the capabilities of the format itself. Contour lines that represent the most common method of distributing elevation data in VPF products fail to provide a continuous surface representation regardless of the resolution. The Digital Terrain Elevation Data (DTED) series of elevation grids could be extended to include higher resolution data, but the high storage requirements and lack of a continuous surface remain to be solved. Triangulated Irregular Networks (TIN) provide the ability to store a high-resolution continuous surface in an efficient manner. EVVPF is the result of the integration of TIN data tables into VPF, based on the investigation of various TIN data structures and VPF as presented in (Abdelguerfi et al. 1995). The M&S Extended Vector Product (MSEVP) prototype being developed is an EVVPF-based product containing the M&S required feature and attribute content, as well as TIN data representation for the elevation coverage and transportation network.

In Sec. 2.0, EVVPF and MSEVP are briefly reviewed. Data structures will be presented for the efficient storage of TIN primitive and feature tables as presented in more detail by Abdelguerfi et al. (1995). Section 3.0 will present methods for populating these tables to create a virtual world.

The first stage in the development consists of constructing the elevation TIN for the desired region. Once a continuous and unambiguous representation of elevation is available, surface features can be integrated into the virtual world. Edges of zero width representing the transportation network were extracted from DTOP and converted to polygons with real-world width. These polygons, as well as other arbitrary area features such as woodland areas, were then integrated into the virtual world by overlaying them onto the elevation TIN. Unlike existing VPF products, MSEVP allows the rendering of surface features in a three-dimensional (3-D) environment, as well as ensuring consistent elevation values between those found in the elevation and other coverages.

2.0 EXTENDED VECTOR PRODUCT FORMAT

The first part of this section will briefly review the actual data structures (i.e., table definitions) used to store the improved triangle-based approach and the relationships between them. For a complete description of the improved triangle-based data structure, the interested reader is referred to Abdelguerfi et al. (1995). This will be followed by an explanation of how these primitives can be linked to geographic features such as lakes, roads, and mountains. The section will conclude by addressing the MSEVP prototype on an abstract level, while the data generation will be presented in the following section.

2.1 EVPF Primitive Table Definitions

The first stage in the integration of TINs into VPF begins with the incorporation of the improved triangle-based data structures into the primitive data model. Figure 1 shows the primitive directory

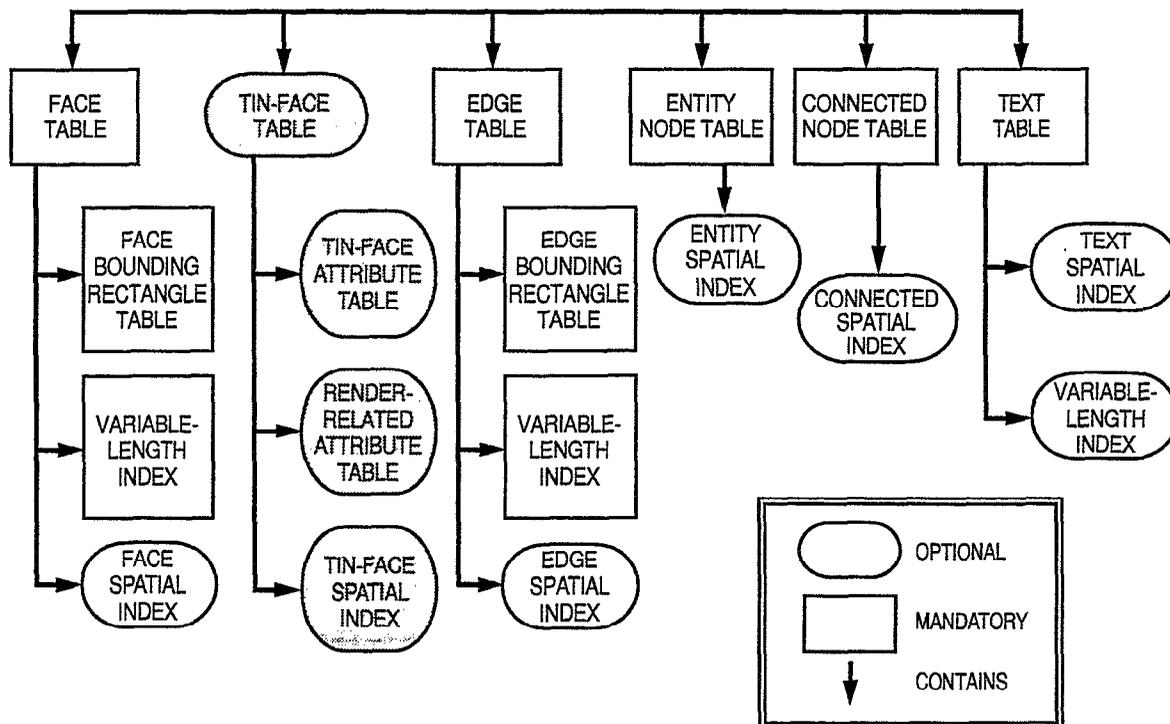


Fig. 1 — Extended primitive directory (topology level 3)

of EVPF. The optional TIN-face table has been added for topology level 3 and comes with optional attribute, render-related attribute, and spatial index tables. In the EVPF TIN-face table shown in Table 1, vertices and adjacent triangles are stored in a counterclockwise fashion. Table 2 shows the definition of a modified connected-node table. The new table includes an additional column, First-TIN, that implements the partial relationship between a connected node and its containing triangles.

While the TIN-face and the connected-node tables can be used to generate a wire mesh frame, a complete 3-D image requires more information about each primitive. To provide this information, a TIN-face attribute table must be introduced to store attribute data for each TIN primitive. Although stored at the primitive level to remain transparent to the user, this table can be compared with an area feature table storing attribute data for the corresponding face primitives. The TIN-face attribute

Table 1 — TIN-Face Definition

COLUMN NAME	DESCRIPTION	COLUMN TYPE	KEY TYPE	OP/MAN
ID	TIN primary key	I	P	M
VERTEX_ONE	First vertex (foreign key to the connected-node table)	I	N	M
VERTEX_TWO	Second vertex (foreign key to the connected-node table)	I	N	M
VERTEX_THREE	Third vertex (foreign key to the connected-node table)	I	N	M
ADJACENT_ONE	ID of triangle adjacent to current triangle along edge between vertex one and two	I/K	N	M
ADJACENT_TWO	ID of triangle adjacent to current triangle along edge between vertex two and three	I/K	N	M
ADJACENT_THREE	ID of triangle adjacent to current triangle along edge between vertex three and one	I/K	N	M

Table 2 — Connected-Node Definition

COLUMN NAME	DESCRIPTION	COLUMN TYPE	KEY TYPE	OP/MAN
ID	Node primary key	I	P	M
*.PFT_ID	Point feature table ID	I	N	OF
CONTAINING_FACE	Always null (included for compatibility)	X	N	O
FIRST_TIN	TIN key (foreign key to the TIN table)	I	N	MI
FIRST_EDGE	Edge key (foreign key to the edge table)	I	N	M1-3
COORDINATE	Node coordinate	C/Z/B/Y	N	M

table stores attribute data for every triangle regardless of its relation or lack of relation to an area or TIN feature. Each triangle in the TIN table must be rendered to display a continuous surface even though each triangle may not be used to define a specific area or TIN feature, such as triangles between two lakes. The definition of the TIN-face attribute table is shown in Table 3.

To render 3-D surfaces and objects adequately, certain attributes must be defined for each triangle or for each vertex of each triangle. Among the attributes that could be defined are color, ambient reflection, diffuse reflection, specular reflection, emissions, and shininess. The color could be represented by a 3-tuple (R, G, B) of floating-point values between 0 and 1. The RGB values represent the primary colors: red, green, and blue. Black is represented by (0, 0, 0) and white by (1, 1, 1). The grayscales are represented in between by equal RGB values. The other values listed above are used to determine shading and various effects light has on the object. To eliminate redundancy, these values are stored in a related attribute table and referenced by a number of individual triangles in the TIN-face table. The actual values needed by an individual rendering package can vary, and as a result, no standard fields are defined. The definition of the render-related attribute table is shown in Table 4.

2.2 EVPF Coverage Level Tables

Besides the addition of primitive level tables, there are three new coverage level tables that closely resemble the area feature tables in both structure and usage. These tables can be used to store features such as roads, lakes, and buildings, as well as arbitrary attribution for each feature class. These tables are the TIN-feature table, the TIN-feature index table, and the TIN-feature join table. The new coverage level feature structure is shown in Fig. 2. The TIN-feature table is used to store attribute information for a given real-world entity represented by the TIN primitives. Any existing area feature or 3-D object can be triangulated and stored using the TIN coverage- and primitive-level tables. The TIN-feature table definition is shown in Table 5. The TIN-feature join

Table 3 — TIN-Face Attribute Table Definition

COLUMN NAME	DESCRIPTION	COLUMN TYPE	KEY TYPE	OP/MAN
ID	TIN feature primary key	I	P	M
RENDER_VALUE	Render values for this triangle (foreign key to the render.rat table)	I	N	O
<Attribute <i>n</i> >	<i>n</i> th attribute	Any	Any	O

Table 4 — Render-Related Attribute Table

COLUMN NAME	DESCRIPTION	COLUMN TYPE	KEY TYPE	OP/MAN
ID	Render primary key	I	P	M
<Attribute <i>n</i> >	<i>n</i> th attribute	Any	Any	O

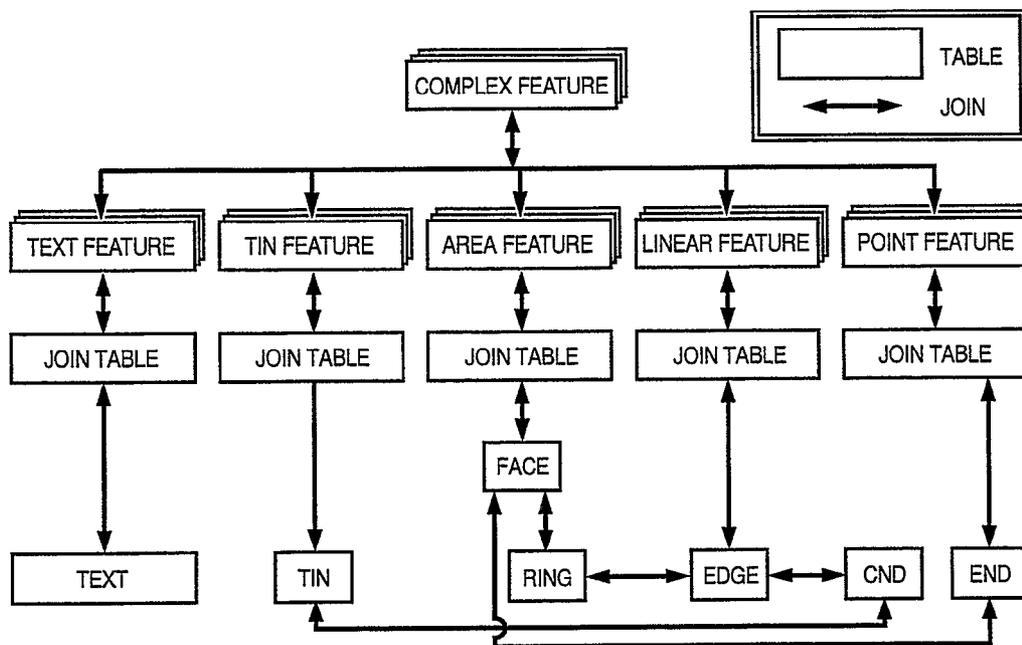


Fig. 2 — New feature class schema

Table 5 — TIN Feature Table Definition

COLUMN NAME	DESCRIPTION	COLUMN TYPE	KEY TYPE	OP/MAN
ID	Feature primary key	I	P	N
TILE_ID	Tile reference ID	S	N	MT
TIN_ID	Primitive ID	S/I/K	N	M
<Attribute <i>n</i> >	<i>n</i> th Attribute	Any	Any	O

table, shown in Table 6, is meant as a link from a specific feature class to the primitive records used to construct the individual feature. This provides an excellent method of selective rendering of features. The TIN-feature index table, shown in Table 7, serves as a comprehensive join table between TIN primitives and related features. From any TIN primitive, i.e., triangle, the feature index table can be used to obtain the feature class ID and feature ID that uses it. The feature class schema table is accessed using the class ID to obtain the name of the proper TIN-feature table. These TIN-feature classes have entries in the feature-class attribute table just as the area, the linear, or point features classes would.

2.3 TINs in Tiled Coverages

VPF allows cross-tile referencing via the introduction of a triplet ID data type. Each triplet ID begins with an 8-bit type byte depicting the format for the rest of the field. The three fields composing the triplet ID are referred to as: ID representing the internal tile primitive ID, TILE_ID representing the external tile reference ID, and EXT_ID representing the external tile reference

Table 6 — TIN Feature Join Table Definition

COLUMN NAME	DESCRIPTION	COLUMN TYPE	KEY TYPE	OP/MAN
ID	Row ID	I	P	M
*_ID	Feature key	I	N	M
TILE_ID	Tile reference ID	S	N	MT/O
TIN_ID	TIN primitive primary key	I/S/K	N	M

Table 7 — TIN Feature Index Table Definition

COLUMN NAME	DESCRIPTION	COLUMN TYPE	KEY TYPE	OP/MAN
ID	Row ID	I	P	M
PRIM_ID	Primitive ID	I	N	M
TILE_ID	Tile reference ID	S	N	MT
FC_ID	Feature class ID	I	N	M
FEATURE_ID	Feature ID	I	N	M

primitive ID. The first 2 bits of the type byte indicate the length of the internal ID; the second 2 bits indicate the length of the TILE_ID field; the third 2 bits indicate the length of the EXT_ID; the final 2 bits are reserved. The field length is either 0, 1, 2, or 4 bytes, depending on the value stored in the 2 bits (i.e., 00 indicates 0 bytes while 11 indicates 4 bytes).

When a triplet ID is encountered, the type byte is analyzed and the resulting values used to find the correct primitive(s). In reference to an adjacent triangle, the existence (nonzero length) of the TILE_ID and EXT_ID fields indicates that the edge of the triangle is on a tile boundary and the adjacent triangle primitive is included in the tile specified by the TILE_ID. The EXT_ID is then used within the correct tile to find the appropriate primitive. This primitive should have a corresponding triplet ID pointer to the original triangle.

In a tiled coverage, triangles will not be allowed to cross tile boundaries. In addition, care must be taken so that triangles at the edge of a tile have at most three neighboring triangles (including the neighboring triangle in the adjacent tile). The tile boundaries could be used as breaklines to ensure this property. This would ensure that no triangles are bisected by the tile boundary while also preserving the adjacency requirement.

2.4 Modeling and Simulation Extended Vector Product

With the introduction of EVPF, the attention now turns to the production of an Extended Vector Product for the M&S community. The production of a prototype database begins with the selection of a representative area of interest. To address the feature and attribute deficiencies adequately and demonstrate the TIN capabilities, the area must contain a wide range of surface

characteristics. A region over Killeen, TX, was chosen for this reason. The selected area spans a 5' x 5' section (-97° 30' to -97° 35' longitude and 31° 5' to 31° 10' latitude) of Killeen, TX, and has sufficient variances in elevation, as well as multiple bodies of water to demonstrate the capabilities of the TIN data structures adequately. In addition, numerous NIMA products exist for the region, providing an excellent source of data from which to populate the prototype.

The MSEVP prototype as presented in Shaw et al. (1996) consisted of eleven coverages: Basic Earth Surface, Data Quality, Demarcation, Elevation, Hydrography, Industry, Physical Geography, Population, Transportation, Utilities, and Vegetation. The basic Earth surface coverage consists of all data relating to all Earth surfaces including coastal and bottom types of hydrological features. The data quality coverage follows the VPF standard of specifying the reliability and accuracy of data contained in the database. The demarcation coverage delineates boundary information for the region of interest. The elevation and transportation coverages will be elaborated on in the next section. It contains a TIN representation of the terrain and transportation network, respectively. The hydrography coverage contains all water-related feature classes such as reservoirs and lakes. Industrial sites, factories, power plants, and related features are divided among the industry and utility coverage. The physical geography coverage is used to store information concerning natural terrain features such as ridge lines. The population coverage contains information on population density and related information.

The culmination of new feature and attribute data, as well as the newly integrated TIN data structures, offers a significant step toward satisfying the needs of the M&S community.

3.0 POPULATING MSEVP TABLES

VPF and EVPF represent a means by which large geographic data sets can be distributed. While the VPF specification does not place any requirements as to how to populate the data structures, it is important to demonstrate how the proposed EVPF TIN tables can be populated using current NIMA data. Although not discussed in great detail, superior alternatives are mentioned when not restricted by currently available data.

The procedure for EVPF data generation begins with the construction of the elevation TIN from gridded elevation values, as well as any breaklines being used to enhance the representation of the terrain. Road polygons are then generated using linear road features and elevation values interpolated from the elevation TIN. These road polygons and other area features are then overlaid onto the elevation TIN to provide an integrated representation of the surface. The process is summarized in Fig. 3 and explained in detail in the remainder of this section.

NIMA data in the form of level 2 DTED and DTOP data exists for the MSEVP prototype region of Killeen, TX, and will be used in the construction of the MSEVP TIN data for the elevation coverage. The transportation edges stored in DTOP have been extracted, widened, and overlaid onto the elevation TIN, thereby providing a "real-world" view of the surface and roads. Additionally, existing two-dimensional (2-D) area features extracted from DTOP, such as woodlands, are also overlaid onto the elevation TIN.

3.1 Generation of Elevation TIN

The first stage in the development of an EVPF product begins with the production of the elevation coverage. The TIN mesh representing the terrain will be used in assigning elevation

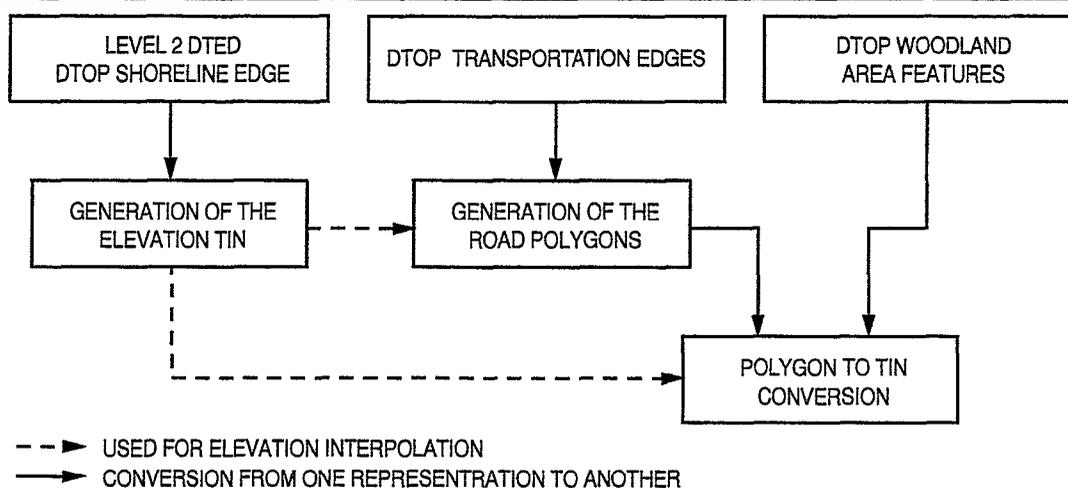


Fig. 3 — Overview of EVPF data generation procedure

values to the nodes in the remaining coverages. Once the coverages (i.e., primitive tables) are complete, TIN data can be converted to a second format and displayed using a 3-D rendering package.

TIN data can be generated from a variety of input sources, such as elevation grids, contour lines, arbitrary collection of edges, or previously generated TINs. This section will trace the populating of the elevation coverage of the MSEVP prototype using 90,300 nodes extracted from level 2 DTED and 639 nodes from five shoreline edges extracted from the hydrography coverage in DTOP.

The elevation points were used to create an ARC/INFO lattice, which serves as the foundation for building an initial TIN for the area. A z-tolerance of 10.0 m was chosen for the construction of this TIN in an effort to balance the need for a high-resolution surface representation while maintaining a manageable data set. The shorelines defined for the area in DTOP were added to the original triangulation as breaklines. The ARC/INFO *ungeneratetin* command was performed on the constrained TIN to create an ASCII file containing the TIN wire frame of the region.

Rendering values based on the elevation of the terrain were assigned for each triangle and each vertex to highlight the versatility of the TIN attribute and render-related attribute tables. The bodies of water, whose elevation can be determined by the color/elevation of the land surrounding it, were assigned a constant color of blue regardless of the elevation. In addition to the assignment of attribute values, a first tin value for each connected node and adjacency relationships for each triangle side is computed before writing the EVPF primitive tables described in the previous section. Figure 4 provides a flow chart describing the complete procedure for the generation of the elevation TIN. The results of this process are shown in Fig. 5.

3.2 Generation of Road Polygons

With the elevation coverage completed, attention can be turned to the addition of other surface features to the surface model. The transportation network is represented by VPF edges in DTOP and all other current NIMA products. The transportation network has been converted from an edge

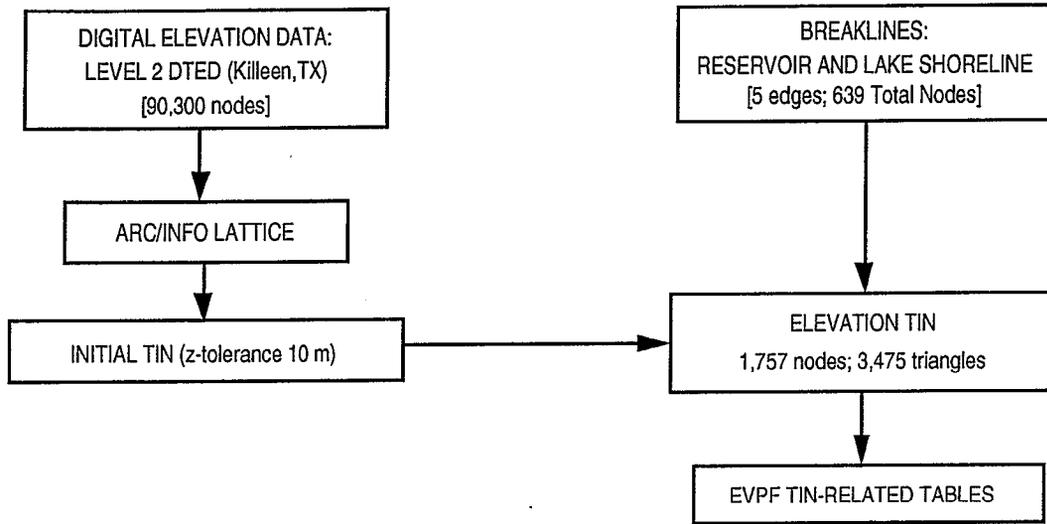
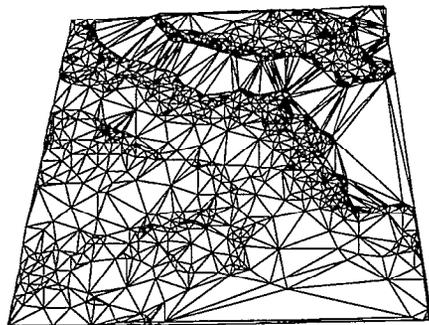


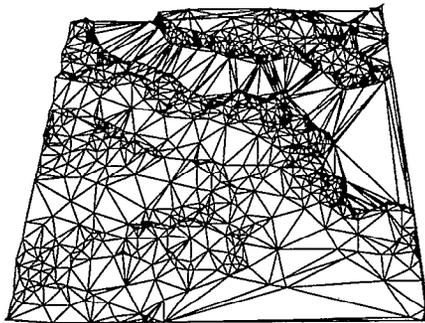
Fig. 4 — Generation of EVPF elevation TIN (Killeen, TX)



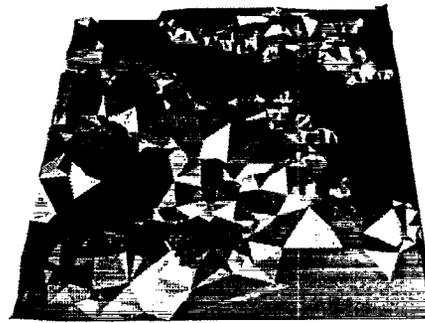
(a) WIRE FRAME UNSCALED



(b) SHADED ELEVATION UNSCALED



(c) WIRE FRAME SCALED BY 4



(d) SHADED ELEVATION SCALED BY 4

Fig. 5 — Examples of Killeen, TX, elevation TIN

with zero width to a polygon with a real-world appearance. This represents the first stage in the construction of the transportation TIN data.

Each VPF edge consists of a series of coordinates in 2- or 3-D space. Based on the coordinate values, as well as on the elevation calculated from the elevation TIN, polygon points are defined to the left and right of the original points. Each edge node is either a start node, intermediate node, or end node. Start and end nodes are handled in essentially the same manner, while intermediate nodes must be treated differently. Figure 6 displays the ultimate goal of expanding an edge to a polygon.

As stated above, this task is accomplished by adding new nodes to the left and right of each original node. The process begins by defining the left and right nodes for the start node. These new nodes are maintained in separate structures/lists for the left and right. The new nodes are added along the vector perpendicular to the first segment and contained in the plane defined by the triangle containing the start node. The slope of each edge segment is calculated and used to determine the equation of the vector perpendicular to the segment. The new nodes will be added along this vector at a fixed, predetermined distance, as shown in Fig. 7.

The intermediate nodes cannot be handled in this way because of the obvious difference in the slope of the segments before and after the node. Single points to the left and right of each intermediate node are calculated by first determining temporary left and right points based on the segments before and after the current intermediate node. The intersection of the lines parallel to the original segments that intersect the temporary points are added as the new left and right points for that intermediate node. The complete procedure for processing intermediate nodes is depicted in Fig. 8.

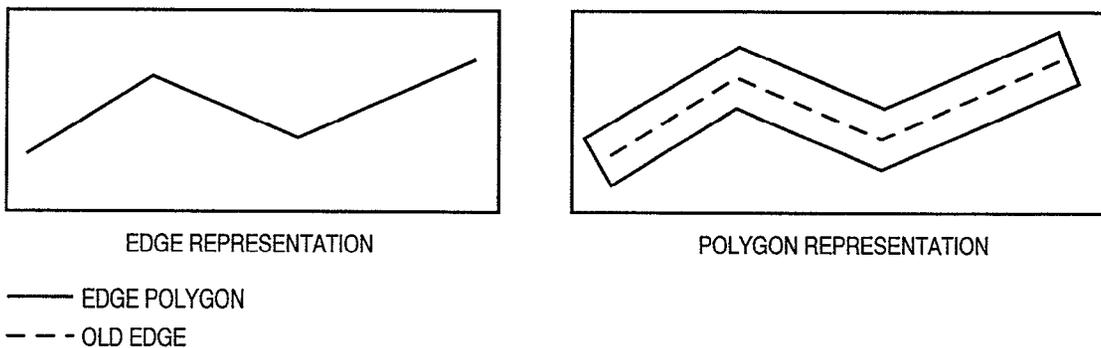


Fig. 6 — Edge-to-polygon conversion

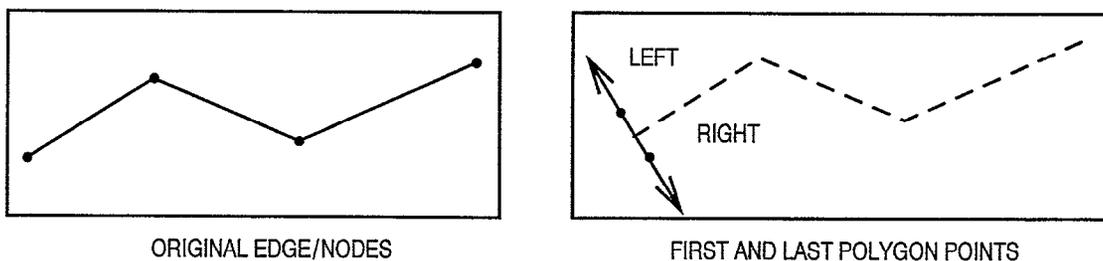


Fig. 7 — Left and right start node determination

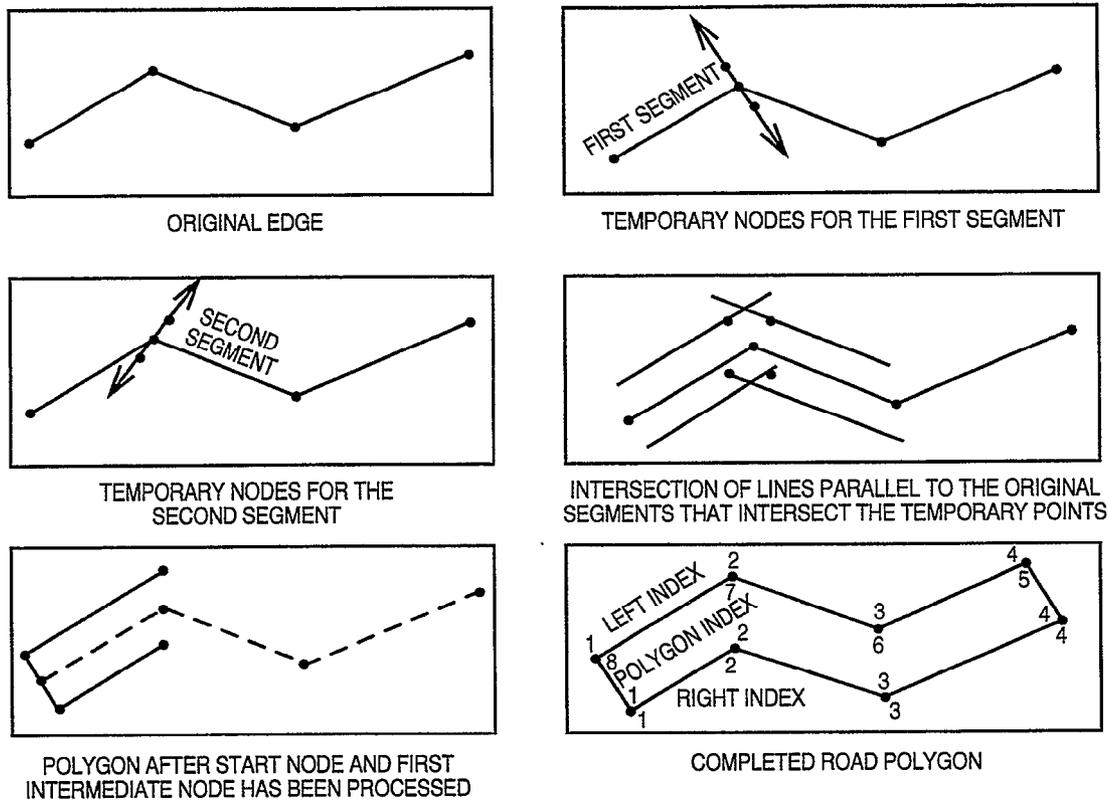


Fig. 8 — Intermediate node processing

After each intermediate node is processed, the left and right coordinates for the end node are calculated and added to the appropriate list. The polygon is then determined by listing the new points in counterclockwise order. This is achieved by listing the right points from the first to the last node followed by the left points from the last to the first node. Since the polygon is not required to be convex, points must be listed in counterclockwise order (which is ensured by listing the right points in order followed by the inverse left points). The pseudo-code of our road-widening algorithm is given in Fig. 9. Figure 15(a) depicts actual road polygons constructed from transportation edges for the prototype region. The boundary of the polygon gives absolute x,y coordinate values for the road. This polygon will then be overlaid and triangulated as described in the next section.

Research into the automated construction of large-scale virtual worlds performed at Carnegie Mellon University (CMU) was presented in Polis et al. (1995). Their work included both terrain representation using TINs as well as a polygonal representation of roads. Their initial terrain TIN is constructed and then altered by the road polygons. The resulting integrated terrain skin consists of the road polygons surrounded by terrain triangles. Although this approach appears to work extremely well in the representation of a site model using a single collection of polygons, VPF and EVPF segregates primitives into disjoint groups (i.e., coverages). Under this approach, the representation of the complete transportation network would be duplicated in both the transportation and elevation coverages.

Ideally, the transportation polygons would be directly extracted from the maps or satellite imagery currently being used to generate the road edges. In this case, the absolute longitude and

```

Variables:
orig_pnt[num_road_points]    -- Original points in the road edge
left[num_road_points]       -- New points added to the left of the original nodes
right[num_road_points]      -- New points added to the right of the original nodes
tmp[2], tmpr[2]             -- Temporary determined from each intermediate node

Functions:
find[left,roght]node(pnts,index,segment)    -- Determines the left/right node given a set of points,
                                              the index of the desired source node, and whether
                                              to use the first or second segment (i.e.,slope
                                              of segment (index-1..index) or (index..index +1)).

intersection(node1, node2, i)    --Determines the intersection of lines through node 1 and node 2
                                  with slopes equal to segment (i-1..i) and (i..i+1),
                                  respectively.

Algorithm:
for i = 0..num_road_points-1 do
  get orig_pnt[i]

left[0] = findleftnode(orig_pnt[0], first)
right[0] = findrightnode(orig_pnt[0], first)

for i = 1..num_road_points-2 do --Process Intermediate Nodes
{
  tmp[10] = findleftnode(orig_pnt[i], second)
  tmpr[0] = findrightnode(orig_pnt[i], second)

  tmp[1] = findleftnode(orig_pnt[i], first)
  tmpr[1] = findrightnode(orig_pnt[i], first)

  left[i] = intersection(tmp[0], tmp[1], i)
  right[i] = intersection(tmpr[0], tmpr[1], i)
}

left[num_road_points-1] = findleftnode(orig_pnt[num_road_points-2], second)
right[num_road_points-1] = findrightnode(orig_pnt[num_road_points-2], second)

for i = 0..num_road_points-1 do
  write(right[i])
for i = num_road_points-1..0 do
  write(left[i])

```

Fig. 9 — Pseudo-code of road expansion algorithm

latitude for the left and right bank of the road would be accurate regardless of the slope of the terrain. As performed at CMU, areas of high relief may require that a subset of the road polygons be used as breaklines to the original triangulation, as the shorelines are used to enhance the accuracy of the triangulation. Figure 10 demonstrates how the use of road polygons as breaklines would enhance the surface representation.

If the road was not used as a constraint but overlaid as described in the next section, the road would not be an accurate representation of its real-world counterpart. Moreover, the road would be at such a slope that the road would not be navigable. For this reason, there are cases in the generation of the database that the roads should be used in the creation of the elevation TIN. Cases

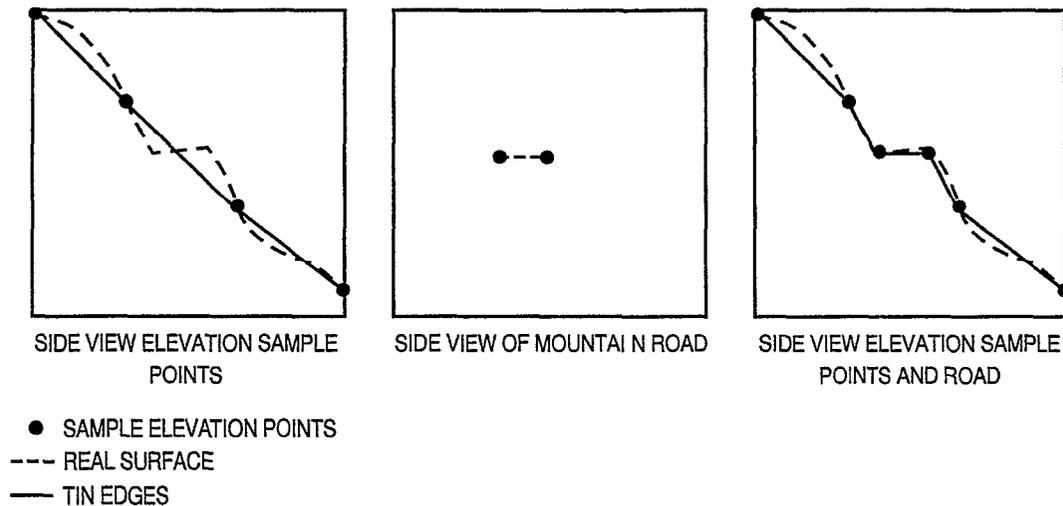


Fig. 10 — TIN enhancement using road polygon

where the terrain is “cut” into or altered by the creation of the road are the most obvious instance, but roads can also be used when the elevation data for the roads are believed to be more accurate than that of the elevation grid or contours available for the region. In these cases, road edges can still be widened by the process described above, but the elevation of each original edge node would be assigned to its new left and right nodes rather than interpolating the value from the elevation TIN. This would ensure that the road is flat from left to right while allowing the road’s elevation to increase or decrease from segment to segment.

In summation, human decisions on preprocessing must be made based on knowledge of the terrain and real roads as to whether all or part of the transportation polygons are used to constrain the elevation coverage.

3.3 Polygon to TIN Conversion

The next stage of developing an integrated virtual world is ensuring the correspondence between area features (or road polygons) to the actual terrain representation in the elevation coverage. The process begins with a polygon with absolute x,y boundary coordinates. After determining which triangles contain these points, the polygon is overlaid by dividing it into one or more child polygons based on the elevation TIN edges. Once the child polygons are defined they are triangulated and written to standard EVPF TIN tables. The pseudo-code representation of the overlay process is specified in Fig. 11.

3.3.1 Overlaying the Polygon

The polygon must be broken into child polygons based on the elevation TIN edges to ensure a continuous surface above the terrain. This is necessary because of the change in slope encountered from triangle to triangle as displayed in Fig. 12.

In addition to adding points where polygon segments intersect triangle edges, triangle vertices or even whole triangles must be added when they are contained within the polygon. With these special cases in mind, the process to overlay the polygon can begin.

```

Variables:
orig_pnts[num_points]           --Original polygon points
temp_pnts[max_points]          --Original polygon points plus intersections of
                                polygon and triangle edges
poly_pnts[max_points]          --Child polygon to be triangulated

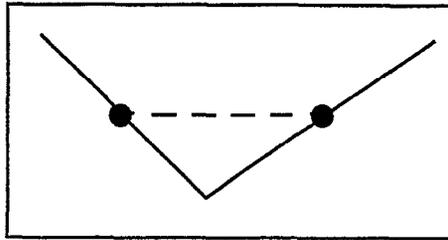
Algorithm:
overlay(triangle, orig_pnts)
{
  for i: = 0..num_points-1 do
    if (orig_pnts[i] in triangle)
      temp_pnts[count++] := orig_pnts[i]
    if (orig_pnts[(i+1)% num_points] not in triangle)
      temp_pnts[count++] := tagged as exit point: intersection of triangle &
                            segment (orig_pnts[i]..orig_pnts[(i+1)%num_points])
    else if (orig_pnts[(i+1)%num_points] in triangle)
      temp_pnts[count++] := tagged as entry point: intersection of triangle &
                            segment (orig_pnts[i]..orig_pnts[(i+1)%num_points])

  while not all temp_pnts processed
  {
    for i: = 0..count-1 do
      if temp_pnts[i] not processed
        poly_pnts[poly_count] := temp_pnts[i]
        temp_pnts[i] := processed
      if temp_pnts[i] is exit point
        if closest entry point (in counterclockwise direction from exit) < i
          i := count
        else
          i := closest entry point index

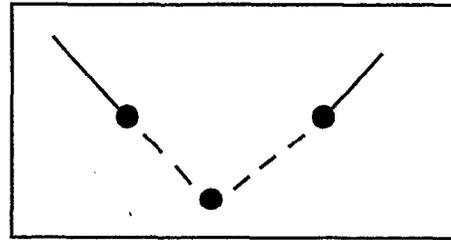
      num_triangles += triangulate(poly_pnts)
    }
  }
  for i: = 1..3 do
    if side[i] intersected then overlay(adjacent side[i], orig_pnts)
  }
  for i: = 1..3 do
    if vertex[i] used
      for each triangle, t, using vertex[i]
        if overlay(t, orig_pnts) == 0
          add t to final triangle list
  }
  return num_triangles
}

```

Fig. 11 — Pseudo-code for polygon overlay algorithm



CONNECTING POINTS IN DIFFERENT TRIANGLES THROUGH THE ADDITION OF A NEW INTERMEDIATE NODE



CONNECTING POINTS IN DIFFERENT TRIANGLES THROUGH THE ADDITION OF A NEW INTERMEDIATE NODE

Fig. 12 — Side view of polygon segment spanning two triangles

The first step is to overlay the polygon onto the triangle containing the first polygon point. The algorithm determines the following cases (Table 8) while processing each of the original polygon nodes and takes the appropriate action:

Table 8 — Current and Next Node Position Cases

CURRENT NODE	NEXT NODE (Current + 1) mod TOTAL	ACTION
Inside Triangle	Inside Triangle	Add current to new list
Outside Triangle	Inside Triangle	Add intersection of current segment and triangle to list and tag as an entry point
Outside Triangle	Outside Triangle	If intersects triangle, add point of intersection closest to the current node as a point of entry and add the second intersection as a point of exit
Inside Triangle	Outside Triangle	Add intersection of current segment and triangle to list and tag as an exit point

After adding all new points to a list, the points must be converted to child polygon(s). These child polygon(s) must then be triangulated for storage in the EVPF TIN tables. The proposed algorithm constructs these child polygons starting with the first new node (which must be an entry or internal point) and adds it to a final polygon list. Subsequent points are added to the final list until an exit point is encountered. Whenever a point is added to the final list, it is marked as processed so that unprocessed points can be processed in subsequent passes through the new polygon list. When an exit point is found, the closest entry point is found by following the side intersected by the exit point in a counterclockwise direction. If a triangle vertex is found, the vertex is added to the final list and the search continues along the next edge. This process continues until the first entry point is encountered. The entry and exit points always occur in pairs because the polygons being overlaid are regular, enclosed polygons. When the index of the entry point is less

than that of the exit point, the child polygon has been completed. If the index of the entry point is greater than that of the exit point, the list pointer is set to the entry point and points continue to be processed. Figure 13 shows how a polygon with three pairs of exit/entry points is broken into two distinct child polygons that must be triangulated individually, and Fig. 14 shows how the vertices are added to the polygon when there is no entry point along the edge being exited.

The algorithm follows the processing of the triangle containing the first polygon node recursively to ensure that all triangles that are intersected by the polygon are processed as well as the triangles not intersected but contained within the polygon. The first stage of recursion begins by overlaying the polygon onto the triangles adjacent to intersected edges. Whenever a triangle edge is intersected, each triangle sharing that edge has a portion of its area inside the polygon and, therefore, the overlay algorithm must be performed for that triangle. Triangles that are completely internal to the polygon are determined by the triangle vertices that are used to construct triangles intersected by the polygon. All triangles having an interior vertex must be called for processing. If a triangle having an interior vertex is not intersected by the polygon, then it is completely internal to the polygon and the triangle itself should be added to the polygon TIN.

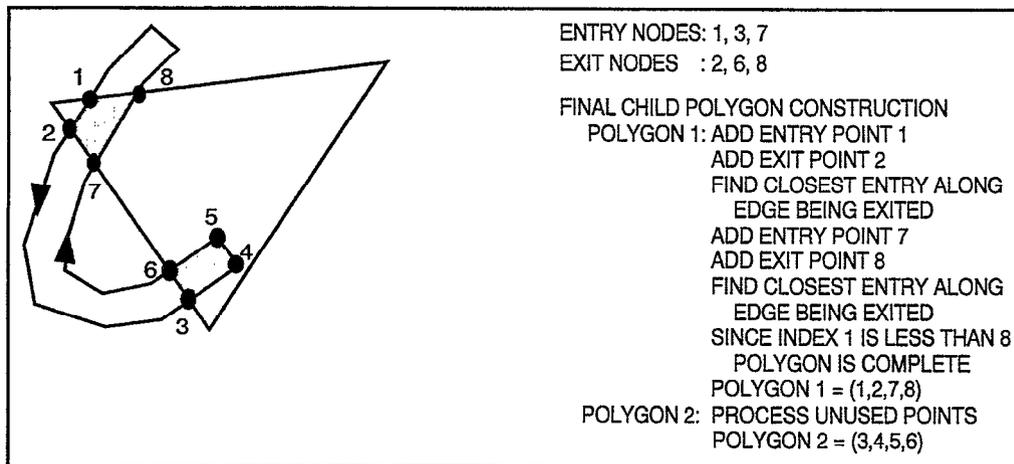


Fig. 13 — Defining distinct intersecting polygons

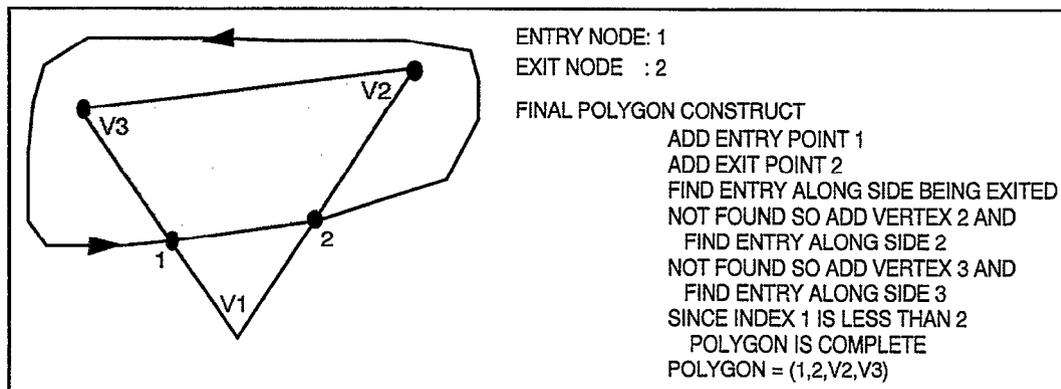


Fig. 14 — Adding triangle vertices

3.3.2 Triangulating Polygons

The child polygons created in the previous section must be triangulated for storage in the EVPF TIN tables. The method of triangulation is derived from the implementation in O'Rourke (1993). Three consecutive polygon vertices are treated as a triangle. If the triangle's sides are not intersected by any polygon edges and are in counterclockwise order (i.e., form a convex angle), the middle vertex is extracted from the list and the three points are stored as a triangle. The algorithm recurses on the remaining vertices (until a single triangle remains). The progression from road polygons to overlaid TINs is exhibited in Fig. 15.

3.3.3 Creation of the EVPF TIN Tables

Once all polygons (i.e., the entire transportation network) have been completely overlaid and triangulated on the TIN mesh, the `first_tin` for each connected node and the adjacency relations for

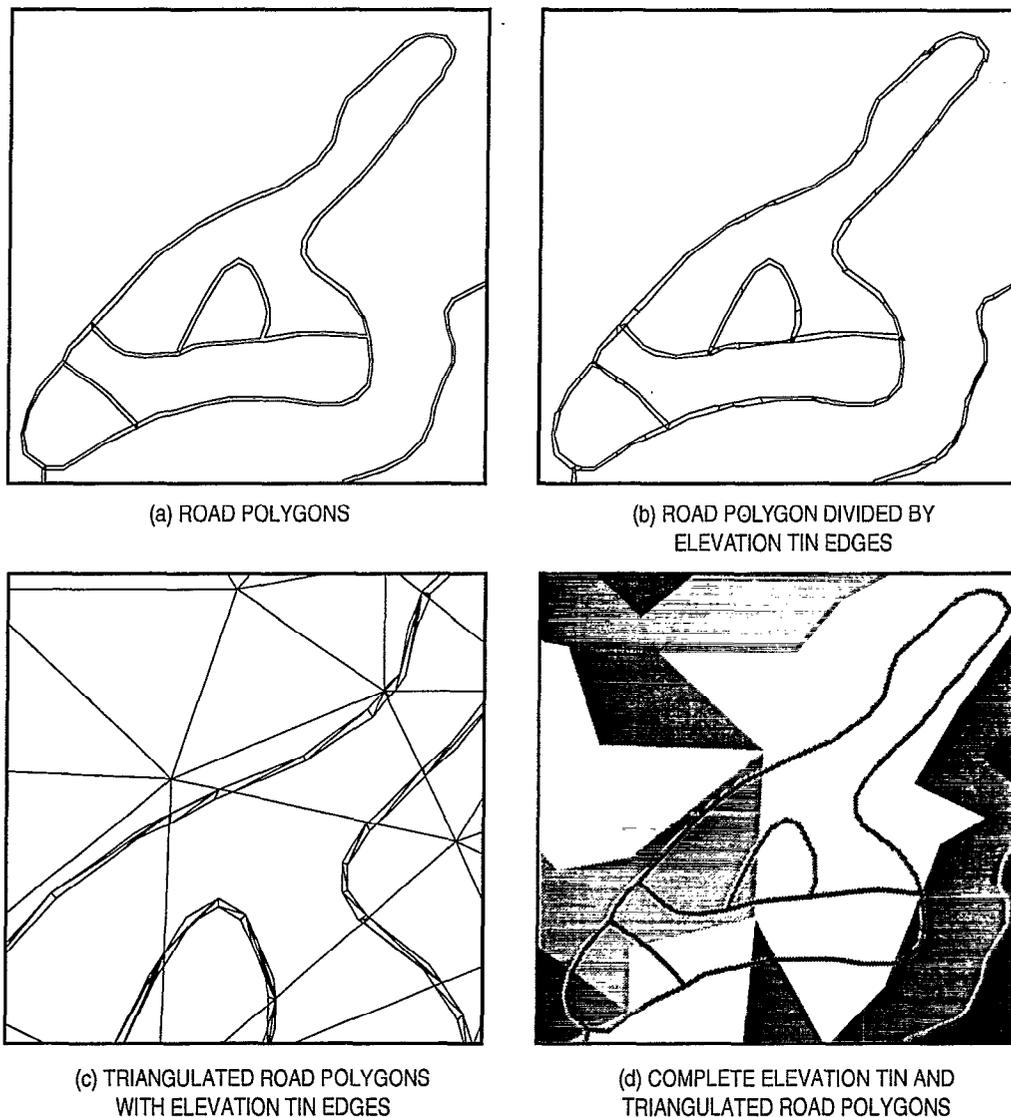


Fig. 15 — Integration of transportation network