REAL-TIME 2D/3D DISPLAY OF DTED MAPS AND EVALUATION OF INTERPOLATION ALGORITHMS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

ALİ DEMİR

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
GEODETIC AND GEOGRAPHIC INFORMATION TECHNOLOGIES

MARCH 2010

Approval of the thesis:

REAL-TIME 2D/3D DISPLAY OF DTED MAPS AND EVALUATION OF INTERPOLATION ALGORITHMS

submitted by **ALİ DEMİR** in partial fulfillment of the requirements for the degree of **Master of Science in Geodetic And Geographic Information Technologies, Middle East Technical University** by,

Prof. Dr. Canan Özgen _____
Dean, Graduate School of **Natural and Applied Sciences**

Assoc. Prof. Dr. Mahmut Onur Karslıoğlu _____
Head of Department, **Geodetic And Geographic Inf. Tech.**

Assoc. Prof. Dr. Ahmet Coşar
Supervisor, **Computer Engineering Dept., METU** _____

Assoc. Prof. Dr. Zuhal Akyürek
Co-Supervisor, **Civil Engineering Dept., METU** _____

Examining Committee Members:

Prof. Dr. Adnan Yazıcı _____
Computer Engineering Dept., METU

Assoc. Prof. Dr. Ahmet Coşar _____
Computer Engineering Dept., METU

Assoc. Prof. Dr. Zuhal Akyürek _____
Civil Engineering Dept., METU

Prof. Dr. Hakkı Toroslu _____
Computer Engineering Dept., METU

Assoc. Prof. Dr. Mahmut Onur Karslıoğlu _____
Civil Engineering Dept., METU

Date: _____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last name: Ali Demir

Signature :

ABSTRACT

REAL-TIME 2D/3D DISPLAY OF DTED MAPS AND EVALUATION OF INTERPOLATION ALGORITHMS

Demir, Ali
M.Sc., Geodetic and Geographic Information Technologies
Supervisor : Assoc. Prof. Dr. Ahmet Coşar
Co-Supervisor: Assoc. Prof. Dr. Zuhal Akyürek

March 2010, 65 pages

In Geographic Information System (GIS) applications, raster data constitutes one of the major data types. The displaying of the raster data has an important part in GIS applications. Digital Terrain Elevation Data (DTED) is one of the raster data types, which is used as the main data source in this thesis. The DTED data is displayed on the screen as digital images as a pixel value, which is represented in gray scale, corresponding to an elevation (texel). To draw the images, the texel values are mostly interpolated in order to perform zoom-in and/or zoom-out operations on the concerned area. We implement and compare four types of interpolation methods, nearest neighbor, bilinear interpolation, and two new proposed interpolation methods (1) 4-texel weighted average and (2) 8-texel weighted average.

The real-time graphical display, with zoom-in/zoom-out capabilities, has also been implemented by buffering DTED data in memory and using a C++ class that manages graphical operations (zoom-in, zoom-out, and 2D, 3D display) by

using Windows GDI+ and OpenGL graphic libraries resulting in 30-40 frames-per-second for one grid of DTED Level 0 data.

Keywords: Digital Terrain Elevation Data (DTED), interpolation, digital image, GDI+, OpenGL

ÖZ

GERÇEK ZAMANDA SAYV HARİTALARIN 2B/3B GÖRÜNTÜLENMESİ VE ENTERPOLASYON ALGORİTMALARININ DEĞERLENDİRİLMESİ

Demir, Ali
Yüksek Lisans, Jeodezi ve Coğrafi Bilgi Teknolojileri Ana Bilim Dalı
Tez Yöneticisi : Doç. Dr. Ahmet Coşar
Ortak Tez Yöneticisi: Doç. Dr. Zuhal Akyürek

Mart 2010, 65 sayfa

Raster veriler Coğrafi Bilgi Sistemleri (CBS) uygulamalarında başlıca veri tiplerinden birini oluşturmaktadır. Bu tip verilerin görüntülenmesi CBS uygulamalarında önemli bir yer tutmaktadır. Bu tezde ana veri kaynağı olarak kullanılan Sayısal Arazi Yükseklik Verileri (SAYV) raster veri tiplerinden birisidir. SAYV verileri ekranda sayısal görüntü olarak gösterilmektedir. Bu görüntüleri oluşturan herbir piksel gri tonlamalı değerler içermekte, ve bu gri tonları SAYV içerisindeki bir yükseklik değerine (teksel) karşılık gelmektedir. Bir görüntünün çizilmesi sırasında, görüntü yakınlaştırılıp uzaklaştırılırken, bir alana ait teksel değerleri çoğunlukla enterpole edilmektedir. Bu tezde dört farklı enterpolasyon metodunu uyguladık ve karşılaştırdık. Bunlar en yakın komşuluk, bilineer, ve iki yeni önerilen metot (1) 4-teksel ağırlıklı ortalama ve (2) 8-teksel ağırlıklı ortalama.

Ayrıca gerçek zamanlı grafiksel görüntüleme, uzaklaştırma ve yakınlaştırma kabiliyetleriyle birlikte, SAYV verilerini ara belleğe alarak grafiksel uygulamaları yöneten (uzaklaştırma, yakınlaştırma ve 2B, 3B görüntüleme) bir

C++ sınıfı geliştirildi. Bu C++ sınıfı Windows GDI+ ve OpenGL grafik kütüphanelerini kullanmakta ve bir gridlik SAYV Seviye 0 verisi saniyede 30-40 çerçeve olacak şekilde çizilebilmektedir.

Anahtar Kelimeler: Sayısal Arazi Yükseklik Verisi (SAYV), enterpolasyon, sayısal görüntü, GDI+, OpenGL

To My Family

ACKNOWLEDGEMENTS

I would like to thank Assoc. Prof. Dr. Ahmet Coşar for his valuable supervision and support throughout the development and improvement of this thesis. This thesis would not have been completed without his guidance. Also, I would like to thank Assoc. Prof. Dr. Zuhal Akyürek for her help us to finish the study. I also would like to thank to my friend Ramazan Sıcakyüz for his help in literature survey, and especially to my dear wife Asuman for always being patient, next and supportive to me during the whole process of this thesis.

TABLE OF CONTENTS

ABSTRACT iv
ÖZ vi
ACKNOWLEDGEMENTS ix
TABLE OF CONTENTS x
LIST OF TABLES xii
LIST OF FIGURES xiii
CHAPTER
1 INTRODUCTION 1
1.1 Digital Terrain Elevation Data (DTED) 1
1.2 Geodetic Datum Used in Turkey 6
1.3 Digital Images 7
1.4 Objectives and Motivation 9
1.5 Organization of the Thesis 9
2 BACKGROUND 10
2.1 Interpolation 10
2.2 Nearest Neighbor Interpolation 10
2.3 Bilinear Interpolation 13
3 THE METHODS AND IMPLEMENTATION 17
3.1 Software 17
3.1.1 The Features of the Software 20
3.1.2 Software Design 23
3.1.3 The Sequence of Operations 29
3.2 Four Texel Weighted Average 31
3.3 Eight Texel Weighted Average 34
3.4 View Quality Comparisons 38
3.4.1 Zoom-In 38

3.4.2 Zoom-Out 41
3.5 3D Image Generation 44
3.6 How To Draw 3D Image 47
4 EXPERIMENTAL RESULTS 49
4.1 Description of Experiment Environment 49
4.2 Execution Time Performances of Interpolation Algorithms 50
4.2.1 Single Grid File Display Experiment 50
4.2.2 Four Grid Files Display Experiment 52
4.2.3 All of Turkey Grid Files Display Experiment 53
4.3 2D Timing Results with GDI+ 54
4.4 3D Timing Results with OpenGL 56
5 CONCLUSION 59
5.1 Recommendations and Future Work 60
REFERENCES 62

LIST OF TABLES

TABLES

Table 1.1 – Parameters of the WGS-84 ellipsoid 2
Table 1.2 - Matrix intervals for DTED Level 0 [NIMA2000] 3
Table 1.3 - Matrix intervals for DTED Level 1 [NIMA2000] 3
Table 1.4 - Matrix intervals for DTED Level 2 [NIMA2000] 4
Table 3.1 – Time measurement sample 29
Table 4.1 - 2D GDI+ drawing times for different size regions 54
Table 4.2 – 3D OpenGL drawing time for different size regions 56
Table 4.3 – 3D rotation times of 1 DTED grid for different degree differences. 57

LIST OF FIGURES

FIGURES

Figure 1.1 - The DTED data record ascending order 4
Figure 1.2 - The grid structure of Turkey [GE2010]..... 7
Figure 1.3 - Digital image representation..... 8
Figure 2.1 - The representation of nearest neighbor algorithm..... 11
Figure 2.2 - Nearest neighbor interpolation on a uniform 2D grid (black points)[Wiki2010] 11
Figure 2.3 – A view of 1 degree DTED cell (using nearest neighbor)..... 12
Figure 2.4 -View of the same cell taken from VTP project [VTP2009] 13
Figure 2.5 - Unit square grid cell layout for bilinear interpolation 14
Figure 2.6 – A view of 1 degree DTED cell (using bilinear interpolation) 15
Figure 3.1 - General view from the application software..... 18
Figure 3.2 – A view from VTP that covers all areas of Turkey 18
Figure 3.3 – a) Representation of grayscale values in an image, ... 20
Figure 3.4 - Distribution of the DTED file tiles on the drawing area 21
Figure 3.5 – File selection dialog box 22
Figure 3.6 - The class diagram of the software 25
Figure 3.7 - The data structure to keep loaded file data 27
Figure 3.8 - The sequence diagram of the software 30
Figure 3.9 - The representation of four texel weighted average interpolation algorithm 33
Figure 3.10 – A view of 1 degree DTED cell (using 4-texel weighted average) 34
Figure 3.11 - The representation of 8-texel weighted average interpolation algorithm 35
Figure 3.12 – A view of 1 degree DTED cell (using 8-texel weighted average) 37
Figure 3.13 - The comparison of the interpolation methods at zoom-in..... 39

Figure 3.14 – RMSEs of the algorithms in some locations (zoom-in)..... 41
Figure 3.15 – The comparison of the interpolation methods in zoom-out..... 43
Figure 3.16 - RMSEs of the algorithms in some locations (zoom-out) 44
Figure 3.17 – Sample 3D images generated from DTED grids 45
Figure 3.18 – Sample views from a 1 DTED grid cell image rotated around y-axis..... 46
Figure 4.1 – The execution time (1-grid cell) of the algorithms, a) Zoom-out, b) Zoom-in 51
Figure 4.2 – The execution time (4-grid cell) of the algorithms, a) Zoom-out, b) Zoom-in 53
Figure 4.3 – The execution time (all of Turkey) of the algorithms, a) Zoom-out, b) Zoom-in..... 54
Figure 4.4 - GDI+ zoom-in time 55
Figure 4.5 – GDI+ zoom-out time..... 56
Figure 4.6 – Zoom in drawing time of 3D images 57
Figure 4.7 – Zoom out drawing time of 3D images 58
Figure 5.1 – A flyover view of 3D martian terrain 61

CHAPTER 1

INTRODUCTION

1.1 Digital Terrain Elevation Data (DTED)

Military applications require accurate positioning and distance/elevation measurement anywhere on earth. With this purpose, the National Imagery and Mapping Agency (NIMA) of USA has developed standard digital datasets, called Digital Terrain Elevation Data (DTED), which is a uniform matrix of terrain elevation values providing basic quantitative data for systems and applications that require terrain elevation, slope, and/or surface roughness information. This standard was originally developed in the 1970s to support aircraft radar simulation and prediction. DTED supports many applications, including line-of-sight analyses, terrain profiling, 3D terrain visualization, mission planning/rehearsal, modeling and simulation. The elevation data are with respect to the reflective surface, which may be vegetation, man-made features or bare earth.

The horizontal datum is the World Geodetic System 1984 (WGS-84), whose specifications are given in Table 1.1 [Hofmann2008]. The coordinates in WGS-84 are given in terms of geodetic latitude (ellipsoidal geographical latitude), geodetic longitude (ellipsoidal geographical longitude) and geodetic height (ellipsoidal geographical height). The vertical datum in DTED is mean sea level (MSL) as determined by the Earth Gravitational Model (EGM 96) geoid, which is

based on orthometric height obtained from the geodetic height (see WGS84) by taking into account geoid undulation [Torge2001].

Table 1.1 – Parameters of the WGS-84 ellipsoid

Parameter and Value	Description
a = 6378137.0m	Semi-major axis of the ellipsoid
f = 1/298.257223563	Flattening of the ellipsoid
$\omega_e = 7292115 \cdot 10^{-11}$ rad/s	Angular velocity of the earth
$\mu = 3986004.418 \cdot 10^{8}$ m^3/s^2	Earth's gravitational constant

A DTED Level 0 data file is produced from DTED Level 1 or VMAP 0. A data file of DTED Level 0, DTED Level 1 or DTED Level 2 is a 1° by 1° cell defined by whole degree latitude and longitude lines. A DTED file shall not cross whole degree latitude or longitude lines. Adjacent one-degree data files shall not have gaps between them and the only overlap that exists is along adjacent boundaries. All adjacent boundaries shall be coincident [NIMA2000].

According to this specification, elevation values within a lake with a diameter equal to or greater than 1200 meters for DTED Level 1 or with a diameter equal to or greater than 600 meters for DTED Level 2 must be identical. Sea or ocean elevation values shall be zero. Drains with a width equal to or greater than 183 meters shall be visible in the DTED data [NIMA2000].

The land elevation values shall be higher than the adjacent water elevations. Extremely shallow land just interior to coastlines shall have +1 meter elevation to force proper land boundary portrayal. Islands with the major axis (longer axis of an ellipse shaped object) equal to or greater

than 600 meters for DTED Level 1 or 300 meters for DTED Level 2 shall be included in the DTED data. Smaller islands shall be included in the DTED data if the relief is equal to or greater than 15 meters above the water level. All land or water bodies below mean sea level shall have negative elevations [NIMA2000].

A DTED data file is a grid cell expressed by geodetic latitudes and geodetic longitudes of a reference system WGS84. The unit of terrain elevation is stated in meters. The positions of elevation posts are identified by the intersections of matrix rows and columns. The intervals of the matrix, which are defined in terms of arc seconds, alter according to change of latitude (see Table 1.2 for DTED Level 0, Table 1.3 for DTED Level 1 and Table 1.4 for DTED Level 2) [NIMA2000].

Table 1.2 - Matrix intervals for DTED Level 0 [NIMA2000]

Zone	latitude	interval latitude (arc sec)	interval longitude (arc sec)
I	0°–50° (North–South)	30	30
II	50°–70° (North–South)	30	60
III	70°–75° (North–South)	30	90
IV	75°–80° (North–South)	30	120
V	80°–90° (North–South)	30	180

Table 1.3 - Matrix intervals for DTED Level 1 [NIMA2000]

Zone	latitude	interval latitude (arc sec)	interval longitude (arc sec)
I	0°–50° (North–South)	3	3
II	50°–70° (North–South)	3	6
III	70°–75° (North–South)	3	9
IV	75°–80° (North–South)	3	12
V	80°–90° (North–South)	3	18

Table 1.4 - Matrix intervals for DTED Level 2 [NIMA2000]

Zone	latitude	interval latitude (arc sec)	interval longitude (arc sec)
I	0°–50° (North–South)	1	1
II	50°–70° (North–South)	1	2
III	70°–75° (North–South)	1	3
IV	75°–80° (North–South)	1	4
V	80°–90° (North–South)	1	6

A DTED data file contains elevation data that falls in a 1-degree grid cell. The southwest corner of the 1-degree cell constitutes the reference origin of a DTED data file. All data files are arranged primarily by latitudes in ascending order (90° South to 89° North), secondarily by longitudes in ascending order (180° West to 179° East) (see Figure 1.1) [NIMA2000].

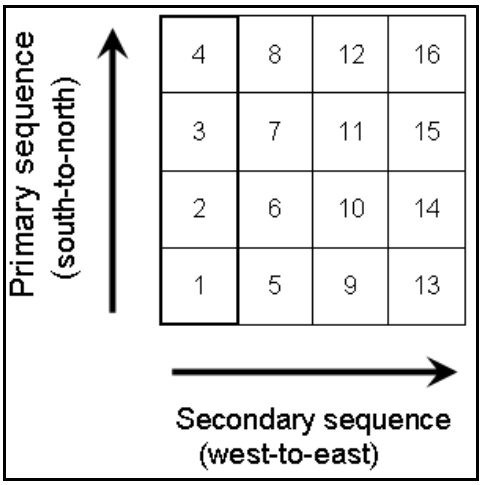


Figure 1.1 - The DTED data record ascending order

To provide overlap between adjacent data files, the degree cell coverage in this standard includes the integer degree values on all sides of the area.

Each data record has one point of overlap with the cell above and one with the cell below (if the record extends to the degree cell limits). Entire data records lying on integer degree longitude values shall exist also in the adjacent degree cell. Data files will not cross integer degree latitude or longitude lines. Adjacent data files shall not have gaps between them and the only overlap that exists is along adjacent boundaries. All data files derived from coincident boundaries of adjacent cells shall be comprised of duplicate data records [NIMA2000].

The grid spacing of latitudes and longitudes is defined as intervals of whole second. The latitude spacing is always consistent in the same level of data, namely, it does not change within various latitudes. The longitude spacing is dependent on the level of the data, as in latitude, and also dependent on the geographic zone. In a data record, the elevation values have a constant longitude. In other words, the first data value in a data record is the southernmost elevation and the last data value is the northernmost elevation. Each data record has different longitude value in a data file. The data records are arranged as ascending (west to east) longitude order in a data file. The elevation values are defined as 2-byte integer, which is high order first and right justified. Negative elevation values are signed magnitude and not complemented [NIMA2000]. The C++ code segment for reading a DTED file is given next.

Algorithm: C++ code segment for reading a DTED file

```
#define DATARECORDSTART 3428

      ifstream DTED_File;
      DTED_File.open(astrFileName.c_str(), ios::binary);

      int line_length = 8 + (2 * m_LatCount) + 4;
      char *linebuf = new char[line_length];

      // Allocate memory for elevations
      m_pElevations = (short *)malloc(m_LonCount*m_LatCount*
sizeof(short));

      DTED_File.seekg(DATARECORDSTART, ios::beg);
      // each elevation, z, is stored in 2 bytes
      unsigned char swap[2];
```

```
int i, j, offset;
for (i = 0; i < m_LonCount; i++) {
      DTED_File.read(linebuf, line_length);
      offset = 8; // record header length
      for (j = 0; j < m_LatCount; j++) {
            swap[1] = *(linebuf + offset);
            swap[0] = *(linebuf + offset+1);
            // swap bytes
            short z = *((short *)swap);
            setValueAt(i, j, z);
            offset += 2;
      }
}
DTED_File.close();
delete [] linebuf;
```

The DTED data traditionally originate from digitization of existing maps or stereographic analysis of aerial photographs. These methods are very time consuming and also susceptible to human and physical handling errors. It is possible to obtain terrain elevation data from other sources, one of which is data produced from C-band Interferometric Synthetic Aperture Radar (IFSAR) used on Shuttle Radar Topography Mission (SRTM) project, which has been initiated in USA and is being operated by National Geospatial-Intelligence Agency (NGA, formerly NIMA) and National Aeronautics and Space Administration (NASA) of this country. The SRTM data collection covers nearly 80% of Earth with a spatial resolution of about 30 meters [Rabus2003]. There may be some voids in the collected data as stored in the finished SRTM DTED. Since all applications require fully populated DTED, the missing parts are obtained by using other alternative DEMs [Intermap2004]. This operation uses basically two approaches, which are interpolation from existing lower resolution data and data obtained by digitizing topographic maps with 1:25000 scale [Bildirici2007].

1.2 Geodetic Datum Used in Turkey

For calculation of geodetic coordinates in Turkey European Datum (ED) 50 has vastly been used as a datum. ED50 was based on International Ellipsoid and

defined for the international connection of the geodetic networks. To transform the coordinates observed by Global Positioning System (GPS) with respect to WGS84 into ED50 datum transformation is needed. With the huge development of the GPS (which is based on the datum WGS84), there existed a need for the development of new datum. In 1999, the national geodetic network TUTGA99 was established [Ayhan2001]. After that, GIS applications, the new maps, are generated according to the new national geodetic network.

The grid structure of Turkey is represented in Figure 1.2. As it can be understood from the figure, geographic location of Turkey is approximately defined as follows; 36º-42º North latitudes and 26º-45º East longitudes.

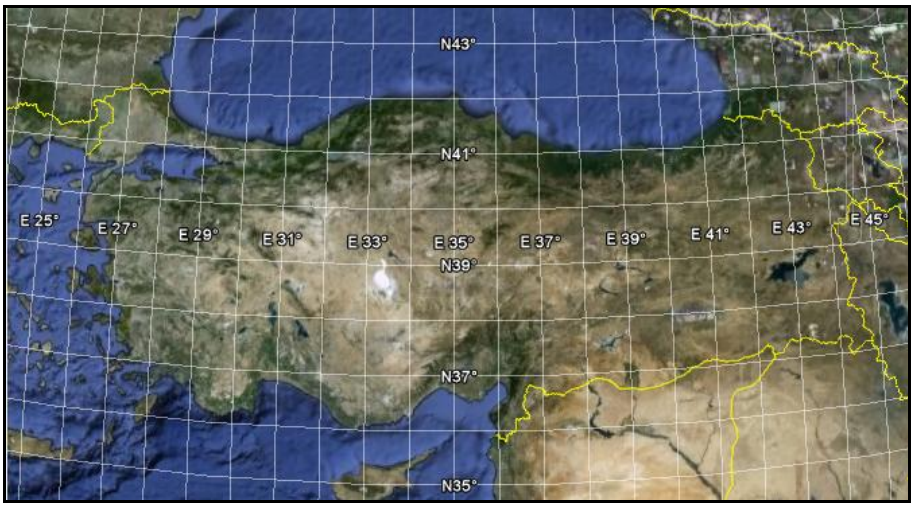


Figure 1.2 - The grid structure of Turkey [GE2010]

1.3 Digital Images

A digital image is a matrix of elements in which each element keeps a digital value that corresponds to a specific data. The smallest element of this matrix is called picture element or pixel. The digital number associated with a pixel

represents the elevation of a terrestrial area, the spectral response for an area element or cell in object space in GIS applications.

The location or address of a pixel is defined on a discrete two-dimensional coordinate system (m,n). *m* represents the number of columns and *n* represents the number of rows. The first pixel (0,0) is located at the top left corner of the image (Figure 1.3).

Columns →

Rows ↓	0	1	2	3	4	5			m
0	28	32	56	65	55	66	56	55	6
1	54	76	67	67	65	45	45	46	45
2	23	6	4	5	6	5	9	62	32
3	25	5	5	6	7	8	33	78	77
4	56	32	33	44	4	5	43	21	23
5	76	34	65	23	21	43	32	69	67
	11	78	16	66	55	5	8	9	12
	14	45	45	65	5	7	16	77	45
n	42	22	42	33	65	55	56	66	77

Figure 1.3 - Digital image representation

The pixel values are non-negative scalars. For intensity image the values are usually referred as gray-scale and has single component. For color images the values have 3 or 4 components as red, green, blue (RGB) colors and/or alfa characteristics.

As the pixel is the smallest unit of an image, the texel (**tex**ture **el**ement) is the fundamental unit of a texture space. The texture is the matrix of the texels.

1.4 Objectives and Motivation

In this thesis we are aiming to measure the execution time costs of various interpolation methods along with the quality of the images they produce. Using these information system designers can choose the most appropriate algorithms in accordance with the quality and speed requirements of their individual projects.

1.5 Organization of the Thesis

In chapter 2 we give background information on the interpolation algorithms available in the literature.

In chapter 3 two new interpolation algorithms are proposed which are explained in detail. The user interface and capabilities of the developed program are discussed and detailed information is given about its software design. The qualities of images generated by new and existing algorithms from literature are visually compared and discussed.

In chapter 4 the execution times of each algorithm are experimentally measured for zoom-in and zoom-out operations.

Finally, in chapter 5, the conclusions and possible future research directions are discussed.

CHAPTER 2

BACKGROUND

2.1 Interpolation

What is the definition of interpolation? There are several possible definitions. Interpolation can be defined as an informed estimate of the unknown [Watson1992]. [Philippe2000] defines it as follows: "Model based recovery of continuous data from discrete data within a known range of abscissa". The latter allows a clearer distinction between interpolation and extrapolation. For resampling operations; several interpolating functions have been used such as nearest neighbor, bilinear, and bicubic.

2.2 Nearest Neighbor Interpolation

"From a computational standpoint, the easiest interpolation algorithm to implement is the so-called nearest neighbor algorithm, where each pixel is given the value of the sample which is closest to it." [Parker1983]. The closest texel location is calculated and the value of that texel is directly used in calculating the value of a pixel. The other 3 texels' values are not used (Figure 2.1). So, the output image contains all real data in pixels. This method is very fast and easy to implement.

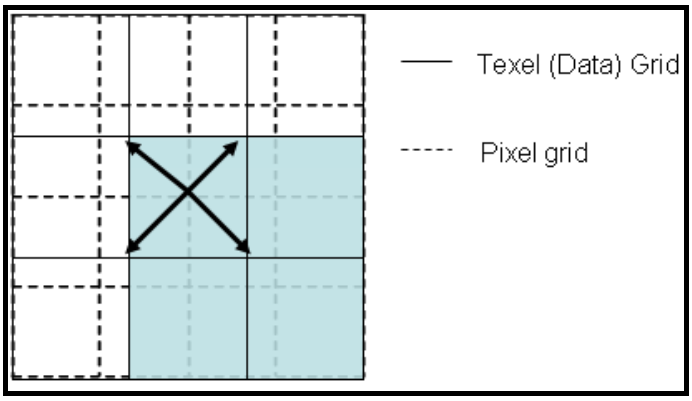


Figure 2.1 - The representation of nearest neighbor algorithm

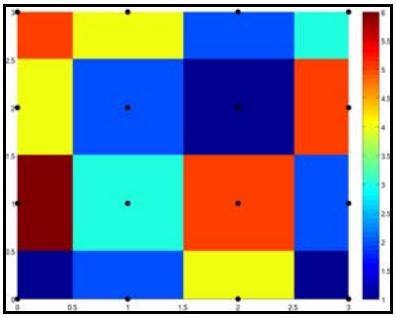


Figure 2.2 - Nearest neighbor interpolation on a uniform 2D grid (black points)[Wiki2010]

A visual representation of nearest neighbor interpolation on a uniform 2D grid is shown in Figure 2.2. The black points inside each colored square are the nearest points to any point in the same colored area.

The nearest neighbor algorithm causes the resampled image to be shifted with regard to the original image by the difference between the positions of the coordinate locations. If, for example, the locations of the resampled points are half way between the original points, the image will be shifted by one-half pixel. This shift means that the nearest neighbor algorithm cannot be used when it is necessary to preserve sub-pixel image relations. Furthermore,

the nearest neighbor algorithm fails completely when resampling to a large matrix size since the pixel values are merely replicated.

There are, however, some quite special properties of the nearest neighbor algorithm. If the resampling is done on a coordinate system with the same spacing as the original coordinate system, then, except for the shift, the resampled data exactly reproduces the original data. That is, the difference between the frequency spectrum of the original and the resampled images is a pure linear phase shift [Parker1983].

In Figure 2.3, a screen view of the 1-degree DTED file (Level 0) is given (compare with the same area in Figure 2.4). The image matrix resolution (600x600) is larger than the data matrix resolution (120x120). The applied interpolation method is nearest neighbor to generate the output image.

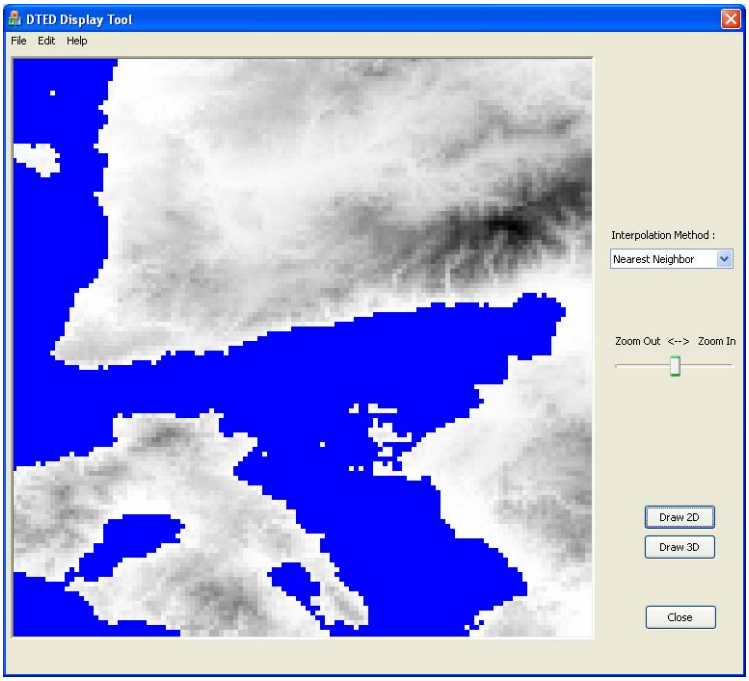


Figure 2.3 – A view of 1 degree DTED cell (using nearest neighbor)

Figure 2.4 -View of the same cell taken from VTP project [VTP2009]

2.3 Bilinear Interpolation

Traditionally, one of the most common used linear interpolations is bilinear interpolation. The concept is, first linearly interpolate in one axis (y-axis), and then second linearly interpolate in other axis (x-axis). The linear interpolation at each axis is performed by distance weighted, in which closer texel point has more effects on the output data proportionally. Four, 2x2, texel grid points, which surround the corresponding pixel location, are used in calculation of the output pixel value [Jain1988].

The bilinear interpolation is described by [Kidner1999] as follows:

> The concept of linear interpolation between two points can be extended to bilinear interpolation within the grid cell. The function is said to be linear in each variable when the other is held fixed. For example, to determine the

height h_i at x, y in Figure 2.5, the elevations at y on the vertical boundaries of the grid cell can be linearly interpolated between h1 and h3 at h_a, and h2 and h4 at h_b. Finally, the required elevation at x can be linearly interpolated between h_a and h_b.

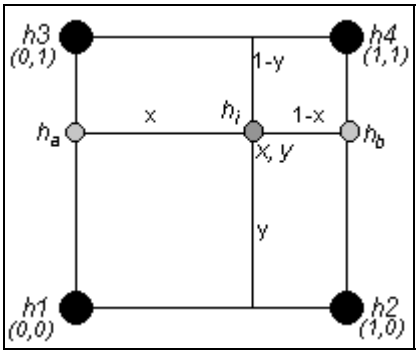


Figure 2.5 - Unit square grid cell layout for bilinear interpolation

The bilinear function is akin to fitting a hyperbolic paraboloid to the four vertices of the grid cell. It is usually written as:

$h_i = a_{00} + a_{10}x + a_{01}y + a_{11}xy$

where

$a_{00} = h1$

$a_{10} = h2 - h1$

$a_{01} = h3 - h1$

$a_{11} = h1 - h2 - h3 + h4$

In Figure 2.6, a screen view of the 1-degree DTED file (Level 0) is given. The image matrix resolution (600x600) is larger than the data matrix resolution (120x120). The applied method is bilinear interpolation to generate the output image.

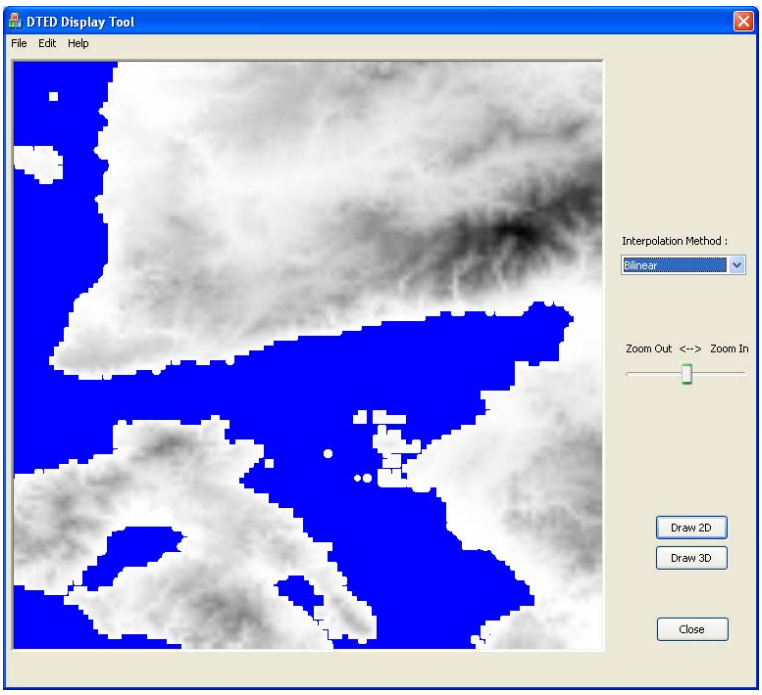


Figure 2.6 – A view of 1 degree DTED cell (using bilinear interpolation)

There are several other interpolation functions available in literature. One is the B-spline function. It is also called linear interpolation where the new point is interpolated linearly between the old points.

The next more complex function uses four nearest points. The Cubic B-Spline interpolating functions are positive everywhere and tend to smooth the resampled image. These functions were investigated by Hou and Andrews [Hou1978]. If Cubic Spline is negative in the interval (1, 2), the functions tend to preserve the original image resolution [Keys1981].

When Cubic B-Spline is not enough to magnify and reduce images and to correct spatial distortions, Cubic Convolution algorithm was developed by Rifman [Rifman1973] and Bernstein [Bernstein1976]. According to R.G. Keys [Keys1981], with the appropriate boundary conditions and constraints on the

interpolation kernel, the order of accuracy of the cubic convolution method is between that of linear interpolation and that of cubic splines.

There is also another interpolation functions; Quadratic functions. Quadratics have been largely disregarded in image resampling because of serious objections. The more serious objection to using quadratics is their filters “are space-variant with phase distortion” [Wolberg1990]. Schafer and Rabiner [Schafer1973] show that any quadratic will produce phase distortions if each quadratic piece starts and ends at the sample points.

CHAPTER 3

THE METHODS AND IMPLEMENTATIONS

In this chapter, the capabilities of the developed software with used methodologies and the implementation of the proposed algorithms are described. The view quality comparisons of the algorithms are also explained.

3.1 Software

The software DTED Display Tool is developed for the visualization of the DTED data and implementation of the most popular two interpolation algorithms used in this thesis, and implementation of the proposed algorithms 4-texel weighted average and 8-texel weighted average. A screen view from the software covering all terrestrial areas of Turkey is given in Figure 3.1. Comparing with the screen view taken from VTP (Figure 3.2), we got the same picture, except that VTP output was generated with color rendering while our software used gray-scale rendering. These two views were generated by loading the same set of DTED files.

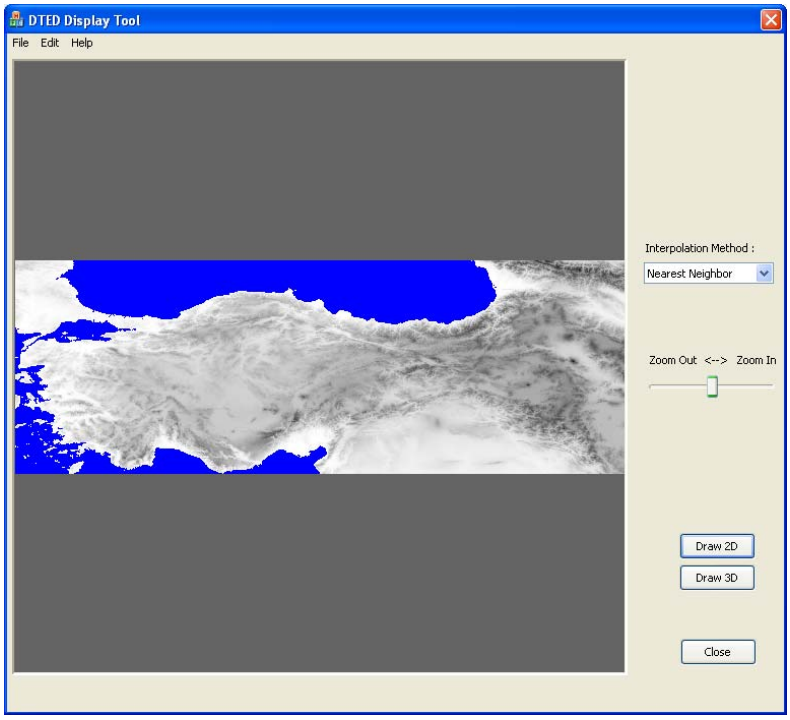


Figure 3.1 - General view from the application software

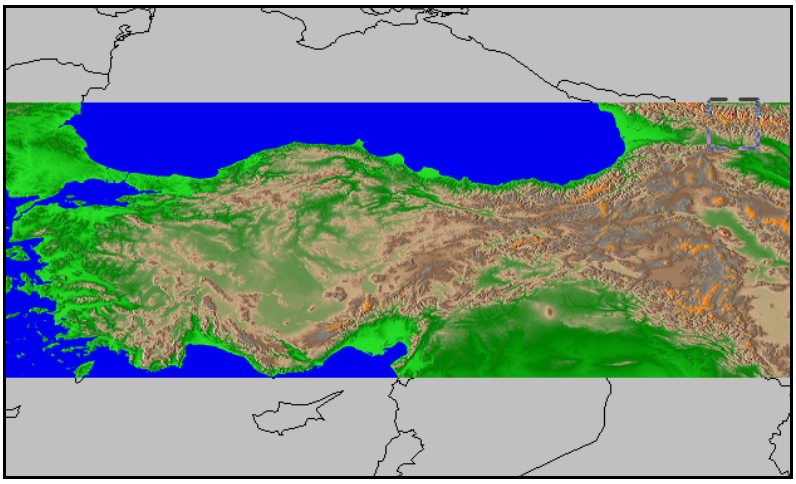


Figure 3.2 – A view from VTP that covers all areas of Turkey

The application was developed with ANSI C++, except for the user interface segment. The user interface was implemented with Microsoft Foundation Classes (MFC) application programming interfaces (API). As an Integrated Development

Environment (IDE), Microsoft Visual C++ 2008 was used. Graphics Device Interface Plus (GDI+), which provides a powerful C++ library, was used in implementing basic 2D image processing operations. The GDI+ is a Microsoft API, which is responsible for representing graphical objects in Windows operating systems [GDI2010]. For development of 3D maps, the standard OpenGL library, which is platform independent, was used including its utility toolkit GLUT [OpenGL2010].

Digital image processing requires high-speed CPU clock rate in all systems. As CPU speed is important, the type of programming language plays also an essential role. In general, usage of C++ language is more preferable with respect to Java and C# in the applications that require high speed operations. Though Java, C# languages provide easy portability into different operating systems, however, their performance is not satisfying in image processing applications that are developed with these languages. The applications that are developed with C++ provide satisfactory level performance. By using only standard C++ libraries (or reducing the number of commands that are not in standard libraries), transferring the software to different operating systems may not be as much complicated as expected. The graphical user interface (GUI) part of the applications takes some time while transferring from one operating system to another, because the GUI libraries are generally more operating system dependent.

In the concept of portability, the DTED Display Tool is developed with using standard C++ libraries. Especially developing core part of the program, the usage of platform dependent commands was avoided as far as possible. Only GUI is fully implemented using MFC library, which is for Windows operating systems.

3.1.1 The Features of the Software

The software has the features as listed below:

1. Grayscale image: In the developed software the grayscale intensity levels are used to display the images of loaded data. The number of bits to compute intensity levels is generally 8-bit, which has 256 levels. In grayscale images, the pixel value 0 represents black, while 255 is for white (Figure 3.3 - a). However, in our software it is in the reverse, the higher pixel value the lower elevation level (Figure 3.3 - b). Namely, the pixel value 255 represents lowest elevation value, and the pixel value 0 represents highest elevation in the corresponding terrain area. Elevation value 0 has special meaning in DTED data representation. The oceans and seas have height value of 0. So, if an elevation value at a grid cell is 0, it means the point belongs to ocean or sea, the corresponding pixel value is assigned to blue color.

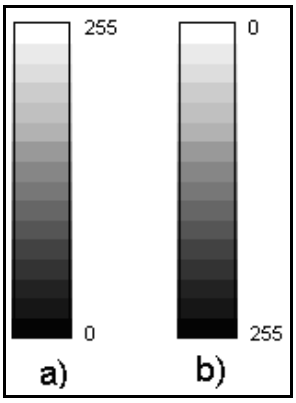


Figure 3.3 – a) Representation of grayscale values in an image,
b) Representation of height (elevation) values in grayscale in the software

2. Drawing Area: The output image is drawn in a fixed size drawing area. The size of the area is set to 600x600 pixels. The representing areas of

each loaded file are drawn on the screen, in the specified drawing area. After all files are loaded, the minimum and maximum of latitudes are calculated from all loaded DTED files. In addition, the minimum and maximum values of longitudes are calculated. After that the latitude and longitude differences, which are in degree level, are computed and the greater difference is found. By using that value the screen resolution of the drawing area, in which a DTED file data (1-degree cell) will be drawn, is calculated with dividing width/height of drawing area by the value found. The drawing area is divided into grid cells vertically and horizontally such that lower left corner has the lowest latitude and longitude values and the top right corner has highest latitude and longitude values. Each grid cell in the drawing area has a resolution that is calculated above. With that resolution value, all loaded data files are drawn on the screen in the corresponding grid cell area. In Figure 3.4, a sample distribution of the DTED grid cells is shown. As can be seen in the figure, there may be gaps in the drawing area. Those gaps (not shaded grid cells) are the areas that have no DTED file loaded.

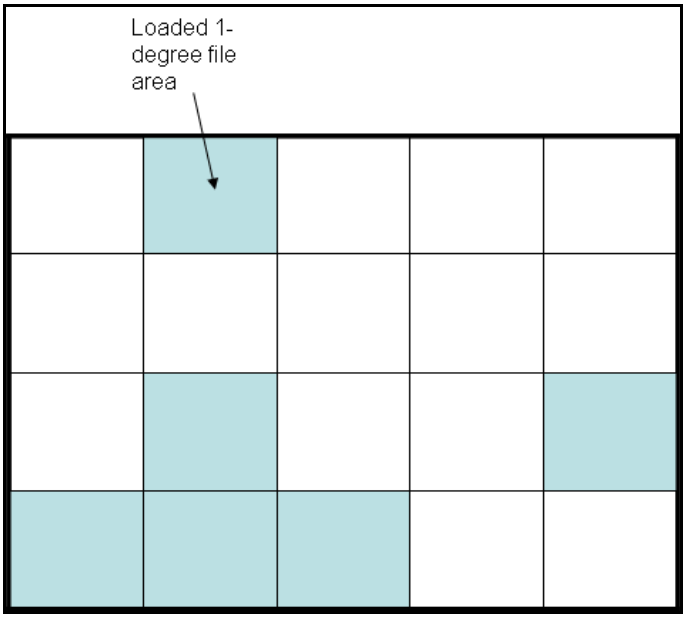


Figure 3.4 - Distribution of the DTED file tiles on the drawing area

3. File Loading: The software allows loading any number of DTED files as there is enough memory storage available. The number of files can be at most 180x360 in theory, if DTED files are available in all areas of the Earth. The file may contain data at different DTED levels. Namely, we can load DTED Level 0, Level 1 or Level 2. Furthermore, these different level files can be loaded at the same time. The program allows this functionality and it was tested in detail successfully while calculating RMSE values. In addition, the files can be selected from different locations. That is, the latitudes or longitudes in the loaded files may not be in consecutive order, as shown in Figure 3.5.

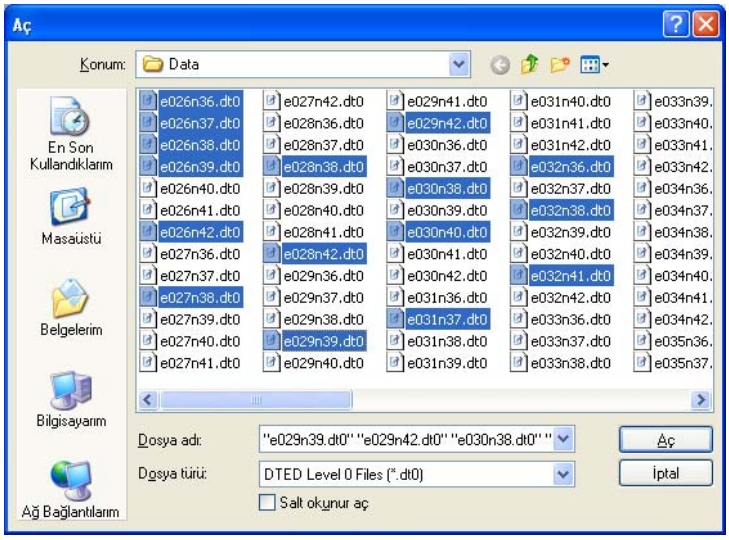


Figure 3.5 – File selection dialog box

4. Interpolation Method Selection: The implemented interpolation methods can also be changed. The default selected method is nearest neighbor interpolation. At runtime, it is possible to select different method, and the

output image is rendered according to the new selected method. This capability provides fast comparison of produced image view quality.

5. Zoom-in And Zoom-out: The software provides zoom-in and zoom-out functionalities. After resulting image is constructed from the loaded files applying one of the interpolation algorithms, it is drawn on the drawing area. It is possible to zoom in/zoom out on the image. The center of the image is taken as the reference point and zooming is performed through the center of the image segment shown on the screen. This capability is available for both 2D and 3D images.

6. Root Mean Square Error (RMSE) Calculation: In order to compare the view quality of the images generated by different interpolation algorithms, the RMSE methodology is used. Calculation of RMSE is performed by using the generated image and the original data of the same area having same resolution with the generated image.

3.1.2 Software Design

In development of the software, object-oriented programming (OOP) paradigm, which is a common approach for the applications that use C++ as programming language, was used. The OOP provides some useful features to the programming such as information hiding, data abstraction, encapsulation, modularity, polymorphism, and inheritance. More information about OOP can be obtained from Wikipedia internet sites [Wiki2010].

As a consequence of using OOP approach, the different functionalities were implemented in the separate classes. There are four main classes in the software, in which all the basic functionalities of the program were implemented. These classes are CTez_DTED_Manager, CTez_DTED_Grid, CTez_Terrain3D and

CTez_TimeMeasurement. The other classes and API's are auxiliary ones that are used to run the application in Windows environment. The class diagram of the software is given in Figure 3.6. The CTezApp class is main application class that the MFC applications have them as the start point.

As can be seen in the Figure 3.6, the only class that has interface with the GDI+ (the Microsoft Windows graphics library) is the CTezDlg. This class defines the behavior of the application's main dialog. It is a dialog based class which covers the main graphical user interface, which includes file loading/unloading menu, interpolation method selection combo box, zoom-in/zoom-out slider, as well as the 2D drawing area for the generated output images. The interface of this class with the GDI+ library includes drawing the generated images by setting pixel by pixel values that is in grayscale.

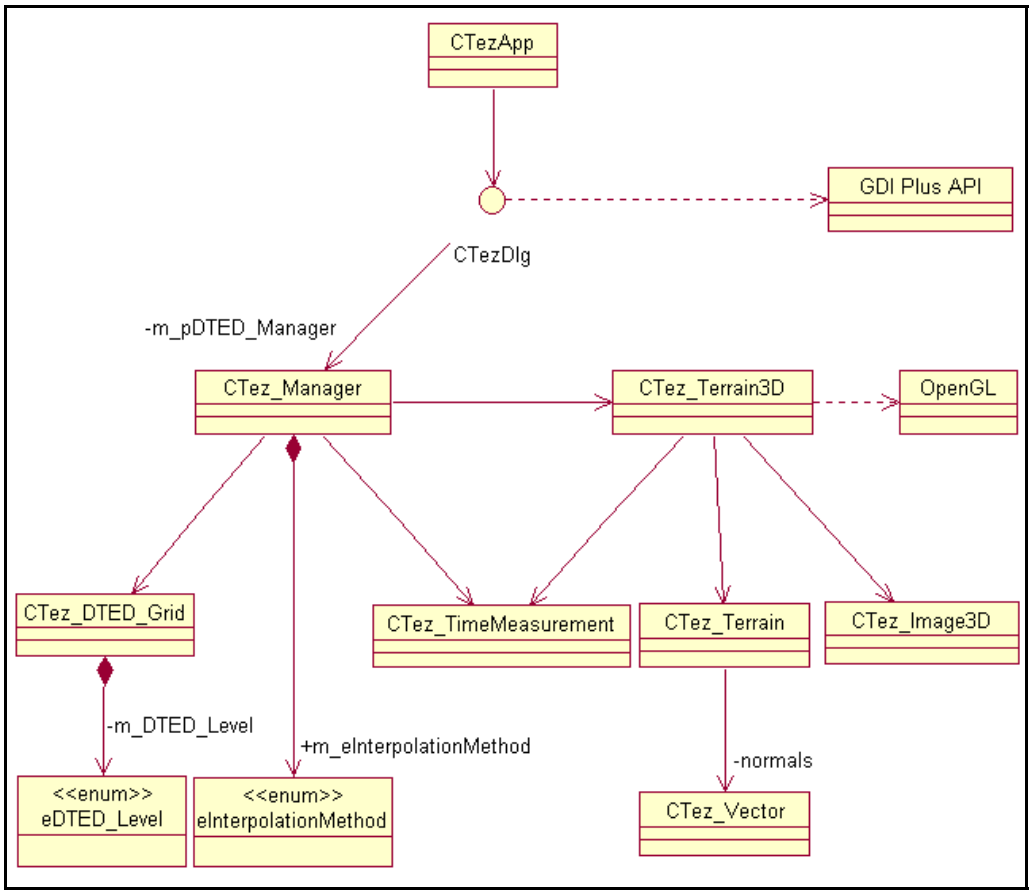


Figure 3.6 - The class diagram of the software

3.1.2.1 CTez_DTED_Grid

Since each DTED file contains the elevations of an area that has coordinates represented by longitude and latitude in degrees, there existed a need to represent the area with a grid class, which is namely CTez_DTED_Grid. This class reads the data of a single file into memory, and keeps that data in memory throughout lifetime of the program. There are some other attributes that the class has; the type of the DTED file, the origin of the location in latitude and longitude, the minimum and maximum values of the elevations, the number of latitude and longitude levels which represents the resolution of the elevation data etc.

The CTez_DTED_Grid class has mutator and accessor methods for the member variables that keep the attributes of the data. In addition to those methods, there are private functions which implement the interpolation methods. The other important functionality of the class is that it is given a resolution of any size of the image area that the data grid will be resampled, and then the value of each new point corresponding to an image pixel is calculated according to the selected interpolation method one by one.

3.1.2.2 CTez_DTED_Manager

This class constitutes the core part of the software. The main functionality of the class is to read all loaded DTED files into memory and keep them in the data structure that maintains easy insert, delete, update, and search operations on the loaded data. It also includes some attributes and their mutator and accessor methods. The attributes are as follows; the start and end latitude/longitude values of the area that is covered by all loaded files, minimum and maximum elevation values calculated from all loaded files and the resulting elevation matrix whose values are computed from each loaded files by the selected interpolation method.

In order to keep the loaded files systematically organized, a new data structure is constructed. As visualized in Figure 3.7, there are two standard C++ library maps defined to keep the data. The first map, called longitude map, has a key value defined by longitudes of the DTED files. The data part of the longitude map contains the instance of CTez_DTED_Grid class whose longitude at origin constitutes the key value for the map element. So, the real data is kept in this longitude map. The second one is named as latitude map. The key is defined by latitudes of the DTED files. The critical point is the data part of this map. It is defined by the longitude maps. Considering the structure of the latitude map, it can be called as map of longitude maps (recall the Figure 3.7). The reason for defining this data structure is that we have the ability to keep all the loaded files

that have same latitude in the same list (longitude map in this data structure). This ability provides us fast access to the desired data grid object, as well as ensuring easy search, insertion and deletion of data in runtime. The usage of the C++ maps, furthermore, yields adding any number of files that may have different latitudes and longitudes. For example, as illustrated in Figure 3.7, the longitude map does not contain data grid having latitude 36 and longitude 27, which means the DTED file is not loaded into the system. This is the flexibility resulting from using C++ maps as data container.

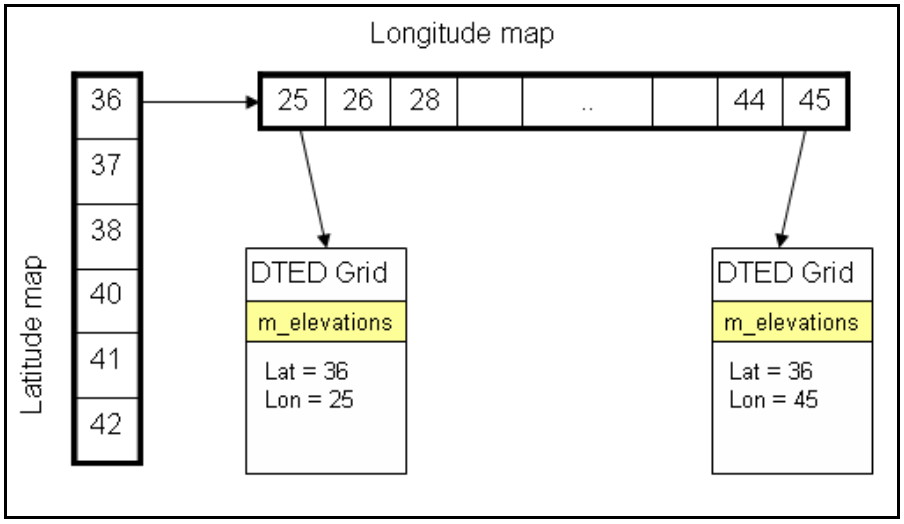


Figure 3.7 - The data structure to keep loaded file data

The other functionality that this class provides is the interface to draw 3D image of the output elevation matrix. The generated 2D grayscale image is given as an input to the 3D image generator class (CTez_Terrain3D) and a 3D image is produced from that 2D image. Therefore, in order to draw a 3D image, an elevation matrix should be created firstly.

3.1.2.3 CTez_Terrain3D

This is the main class that performs the 3D functionality of the software. It takes the image matrix as the input, assuming the elevation values are interpolated into grayscale image pixel value, namely 0 – 255 intervals. The OpenGL and GLUT configurations are set by this class. First of all, the window is configured and created, and then the OpenGL parameters are initialized. The display, resize and handle methods are registered, and lastly, the event processing loop starts. The registered methods were defined globally, because GLUT requires accessing the addresses of these functions. The normals of points are also computed while loading the image matrix into this class.

3.1.2.4 CTez_TimeMeasurement

One of the main goals of this thesis is to compare the processing time of each interpolation methods and measure the execution time of some other functionalities such as drawing time of 2D and 3D images. To get this aim, the class, CTez_TimeMeasurement, is implemented. The functionalities implemented in this class are taking the start time before executing an algorithm and then stop time after the execution ends. The time difference is calculated and if enough number of samples obtained, the average, minimum/maximum time differences and standard deviations are computed. An example of calculated time values is given in Table 3.1.

In calculation of the time difference, QueryPerformanceCounter() and QueryPerformanceFrequency() system calls are used in Windows platforms. These operations provide high-resolution performance in timing calculations that require more precise measurements. QueryPerformanceCounter() gets the current value of the performance counter. QueryPerformanceFrequency() gives the frequency of the performance counter, in counts per second [Timers2010]. By

dividing the current value of counter by frequency, we get the current time in microseconds resolution. After that it can be converted to milliseconds.

Table 3.1 – Time measurement sample

Sample No	Start Time	Stop Time	Processing Time (ms)
1	79269686.7	79269704.2	17.5
2	79269706.0	79269723.3	17.3
3	79269724.3	79269741.6	17.3
4	79269742.5	79269759.9	17.5
5	79304736.9	79304754.3	17.4
6	79304755.5	79304772.9	17.4
7	79304773.8	79304791.2	17.5
8	79304792.2	79304809.5	17.3
9	79322888.7	79322906.3	17.6
10	79322924.8	79322907.4	17.4
Min time = 17.3	Max time = 17.6	Average = 17.4,	STDEV = 0.1

3.1.3 The Sequence of Operations

There are two main steps of sequence to concern in order to use the developed application;

- Firstly; the DTED files that will be displayed should be loaded to the application.
- After that; the resulting image data is produced according to the selected interpolation method and then displayed on the screen.

The sequence of the operations that should be followed is illustrated in Figure 3.8. There is no need to pursue an order for the rest of operations after accomplishing the steps mentioned above. Namely, at any time, the interpolation

method to produce an output image can be changed, or zooming activities can be performed. Furthermore, we can load a new DTED file all times.

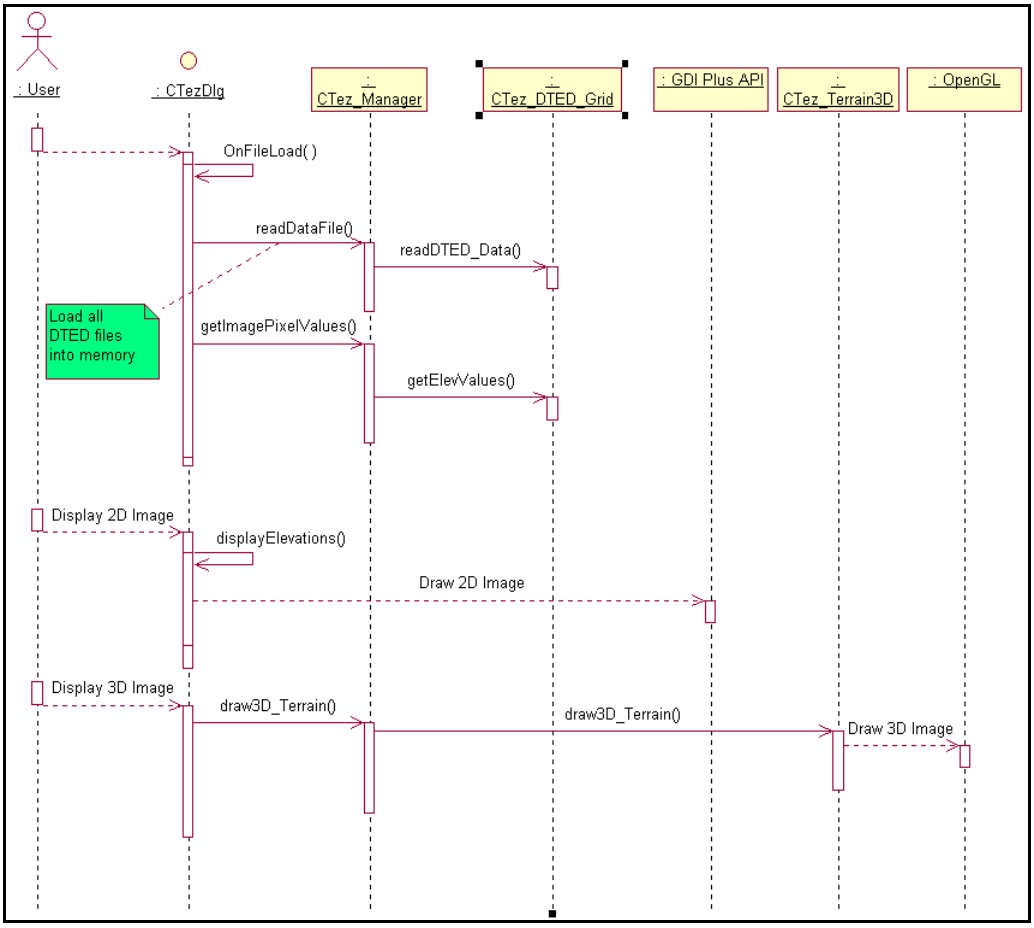


Figure 3.8 - The sequence diagram of the software

Since speed is an essential issue in applications that include image processing activities, these software require high performance systems as well as the design and implementation should be maintained via considering the speed requirements. Bearing in this requirement, the design of our software was conducted as considering performance issues such as keeping all DTED data in memory, preferring mostly usage of pointers in C++ etc.

While loading files to the software, as mentioned before, we preferred to keep all DTED data in the memory, especially in cache. This is because the disk access is always more expensive than accessing to RAM. The disk access requires physical movements of the reading head, which results in taking longer time to access data. In other words, caching the data in the memory provides rapid access (resulting in high speed) to the requested data instead of reading the file at any time needed. So, the difference between disk and memory access times is at highly considerable level.

3.2 Four Texel Weighted Average

One of the proposed interpolation algorithms is four texel weighted average. The idea behind this algorithm is that the texels that are in very high proximity of the corresponding pixel location has somewhat strong impacts on the elevation of that point. The effective texels are located at left, right, top, and bottom sides of the pixel location as well as the center texel, in which the location pixel is embodied. As the name of the algorithm implies, there are four texels around the pixel point including centered one. In this algorithm, the effect of a surrounding texel to the newly calculated pixel value is directly proportional to the closeness of that texel location to corresponding pixel location. The weighting factors are calculated from the distances of pixel location to the location of the texels in concern.

The concept of this algorithm is represented in Figure 3.9. Here is the pseudo code of the algorithm:

A. By mapping the image matrix to DTED data matrix, so that corner points exactly coincide with, find the scaling factor which is of floating point type.

B. Calculate the location, x-y coordinate values, of the pixel in texel grid environment, by multiplying the pixel coordinate values with the scaling factor found before.

C. Find the coordinates of texels, where;
   a. T_C : the texel that embodies the pixel location
   b. T_L : the texel on the left of pixel location
   c. T_R : the texel on the right of pixel location
   d. T_T : the texel on the top of pixel location
   e. T_B : the texel on the bottom of pixel location

D. Compute the distances of the texel points T_C, T_L, T_R, T_T, and T_B to the location of the pixel, let's call the distances as d_C, d_L, d_R, d_T, and d_B respectively.

E. Compute the weight of each texel point as follow;
   a. Find the sum all distances, $d_S = d_C + d_L + d_R + d_T + d_B$
   b. Compute the reverse distances as;
      i. $d_{RC} = d_S / d_C$
      ii. $d_{RL} = d_S / d_L$
      iii. $d_{RR} = d_S / d_R$
      iv. $d_{RT} = d_S / d_T$
      v. $d_{RB} = d_S / d_B$
   c. Find the sum of reverse distances, $d_{RS} = d_{RC} + d_{RL} + d_{RR} + d_{RT} + d_{RB}$
   d. Calculate the weighting values of each point as;
      i. $W_C = d_{RC} / d_{RS}$
      ii. $W_L = d_{RL} / d_{RS}$
      iii. $W_R = d_{RR} / d_{RS}$
      iv. $W_T = d_{RT} / d_{RS}$
      v. $W_B = d_{RB} / d_{RS}$

F. Calculate the pixel value (V_P) as;

$$V_P = V_C*W_C + V_L*W_L + V_R*W_R + V_T*W_T + V_B*W_B$$

where V_C, V_L, V_R, V_T, V_B are the values of the texel points T_C, T_L, T_R, T_T, and T_B respectively.

In Figure 3.10, a screen view of the 1-degree DTED file (Level 0) is given. The image matrix resolution (600x600) is larger than the data matrix resolution (120x120). The applied method is four texel weighted average interpolation to generate the output image.

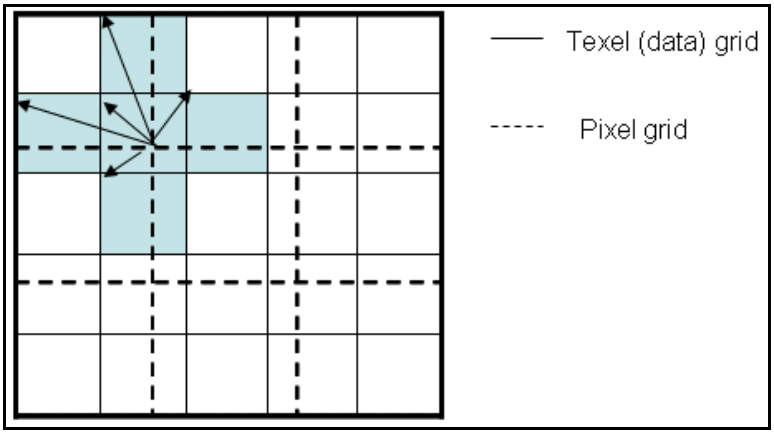


Figure 3.9 - The representation of four texel weighted average interpolation algorithm

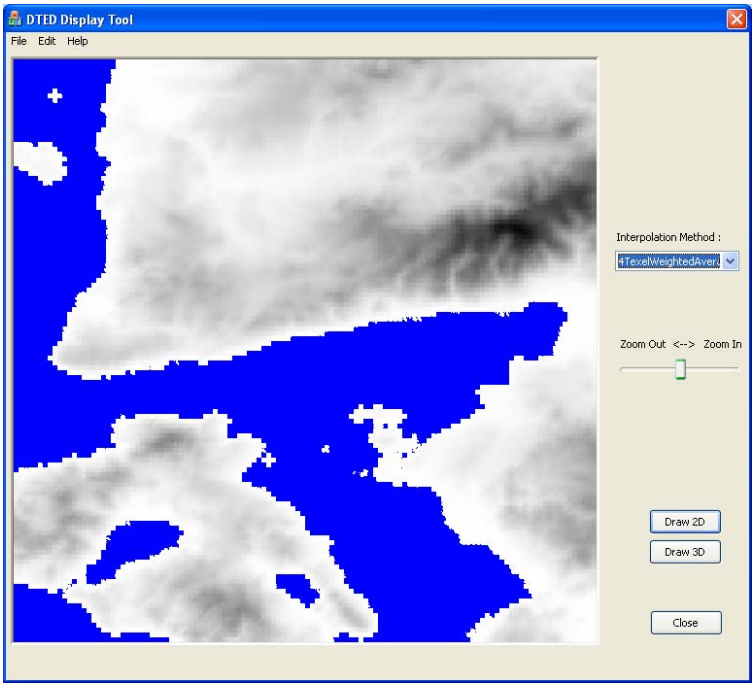


Figure 3.10 – A view of 1 degree DTED cell (using 4-texel weighted average)

3.3 Eight Texel Weighted Average

The second proposed algorithm in calculating the unknown points while performing zoom-in/zoom-out operations is eight texel weighted average interpolation algorithm. The main concept of the algorithm is similar to four texel weighted average. The difference is that the diagonal alignments of the texel values are taken into consideration. In other words, not only the texels located on the left, right, top, and bottom of the center texel, but also the ones that are located at the corners of the center texel have effects on the pixel value. As in the four texel weighted average algorithm, the closer texel has more impact on the pixel value, and the weighting factors are calculated from distances.

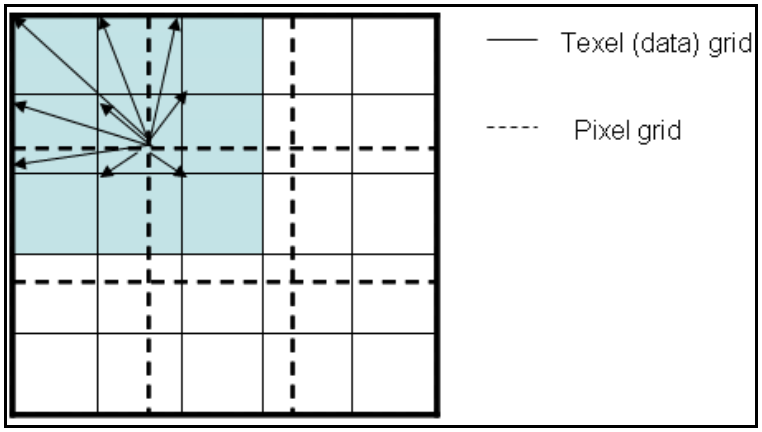


Figure 3.11 - The representation of 8-texel weighted average interpolation algorithm

The concept of this algorithm is represented in Figure 3.11. The pseudo code of the algorithm is given below:

A. By mapping the image matrix to DTED data matrix, so that corner points exactly coincide with, find the scaling factor which is of floating point type.

B. Calculate the location, x-y coordinate values, of the pixel in texel grid environment, by multiplying the pixel coordinate values with the scaling factor found before.

C. Find the coordinates of texels, where;
   a. T_C : the texel that embodies the pixel location
   b. T_L : the texel on the left of pixel location
   c. T_R : the texel on the right of pixel location
   d. T_T : the texel on the top of pixel location
   e. T_B : the texel on the bottom of pixel location
   f. T_{TL} : the texel on the top-left of pixel location
   g. T_{TR} : the texel on the top-right of pixel location

h. T_{BL} : the texel on the bottom-left of pixel location

i. T_{BR} : the texel on the bottom-right of pixel location

D. Compute the distances of the texel points T_C, T_L, T_R, T_T, T_B, T_{TL}, T_{TR}, T_{BL}, and T_{BR} to the location of the pixel, let's call the distances as d_C, d_L, d_R, d_T, d_B, d_{TL}, d_{TR}, d_{BL}, and d_{BR} respectively.

E. Compute the weight of each texel point as follow;

a. Find the sum all distances, $d_S = d_C + d_L + d_R + d_T + d_B + d_{TL} + d_{TR} + d_{BL} + d_{BR}$

b. Compute the reverse distances as;

i. $d_{RC} = d_S / d_C$

ii. $d_{RL} = d_S / d_L$

iii. $d_{RR} = d_S / d_R$

iv. $d_{RT} = d_S / d_T$

v. $d_{RB} = d_S / d_B$

vi. $d_{RTL} = d_S / d_{TL}$

vii. $d_{RTR} = d_S / d_{TR}$

viii. $d_{RBL} = d_S / d_{BL}$

ix. $d_{RBR} = d_S / d_{BR}$

c. Find the sum of reverse distances, $d_{RS} = d_{RC} + d_{RL} + d_{RR} + d_{RT} + d_{RB} + d_{RTL} + d_{RTR} + d_{RBL} + d_{RBR}$

d. Calculate the weighting values of each point as;

i. $W_C = d_{RC} / d_{RS}$

ii. $W_L = d_{RL} / d_{RS}$

iii. $W_R = d_{RR} / d_{RS}$

iv. $W_T = d_{RT} / d_{RS}$

v. $W_B = d_{RB} / d_{RS}$

vi. $W_{TL} = d_{RTL} / d_{RS}$

vii. $W_{TR} = d_{RTR} / d_{RS}$

viii. $W_{BL} = d_{RBL} / d_{RS}$

ix. $W_{BR} = d_{RBR} / d_{RS}$

F. Calculate the pixel value (V_P) as;

$$V_P = V_C*W_C + V_L*W_L + V_R*W_R + V_T*W_T + V_B*W_B + V_{TL}*W_{TL} + V_{TR}*W_{TR} + V_{BL}*W_{BL} + V_{TR}*W_{TR}$$

where V_C, V_L, V_R, V_T, V_B, V_{TL}, V_{TR}, V_{BL}, V_{BR} are the values of the texel points T_C, T_L, T_R, T_T, T_B, T_{TL}, T_{TR}, T_{BR}, and T_{BR} respectively.

In Figure 3.12, a screen view of the 1-degree DTED file (Level 0) is given. The image matrix resolution (600x600) is larger than the data matrix resolution (120x120). The applied method is eight texel weighted average interpolation to generate the output image.

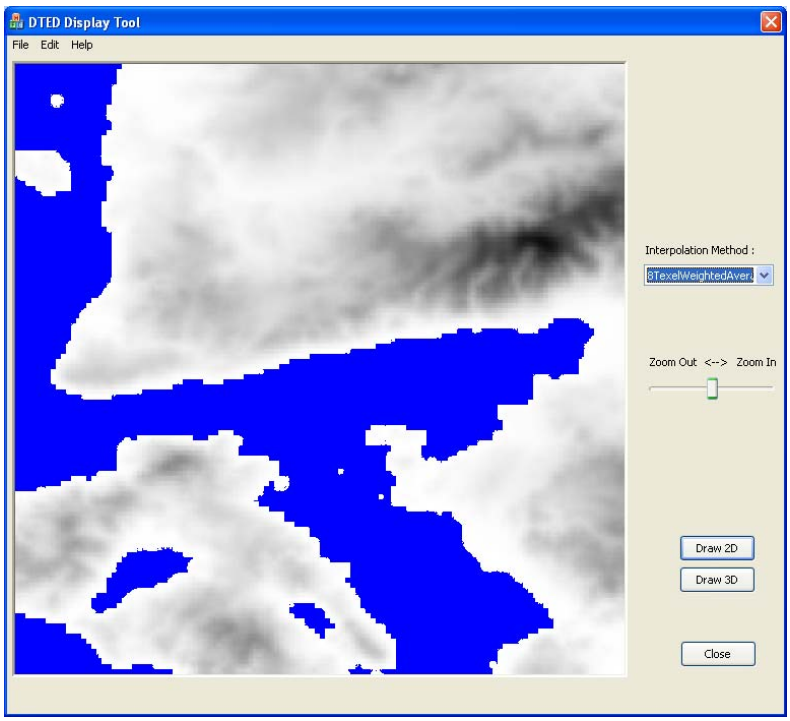


Figure 3.12 – A view of 1 degree DTED cell (using 8-texel weighted average)

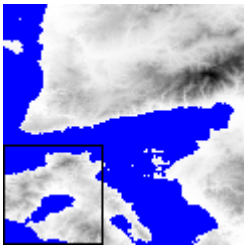
3.4 View Quality Comparisons

The conventional interpolation algorithms, nearest neighbor and bilinear (described in chapter 2), and the proposed algorithms, 4-texel weighted average and 8-texel weighted average (described in this chapter above) were all implemented in the developed software to compare the generated images. There are some viewing artifacts occurred on the output images while doing the zoom-in and zoom-out operations. These artifacts can be one or more of blocking, blurring, aliasing (jagging), and so on.

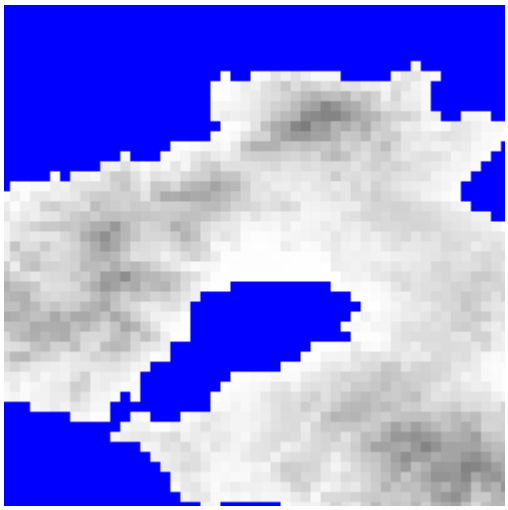
3.4.1 Zoom-In

In Figure 3.13, a list of images is given so that we compare view quality of the algorithms when the DTED grid cells are zoomed-in. The table contains 5 images; a) an image that is not resampled, b) an image that is resampled with nearest neighbor, c) an image that is resampled with bilinear, d) an image that is resampled with 4-texel weighted average, and e) an image that is resampled with 8-texel weighted average. All resampled images are zoomed in at 500% scale, and they represent the area that is enclosed by a black frame on the first image (a). The image at item (a) is a visual representation of 1-degree DTED grid cell and there is no resampling. Therefore, since the DTED Level 0 grid cell has a resolution 120x120, that image has the same resolution value too.

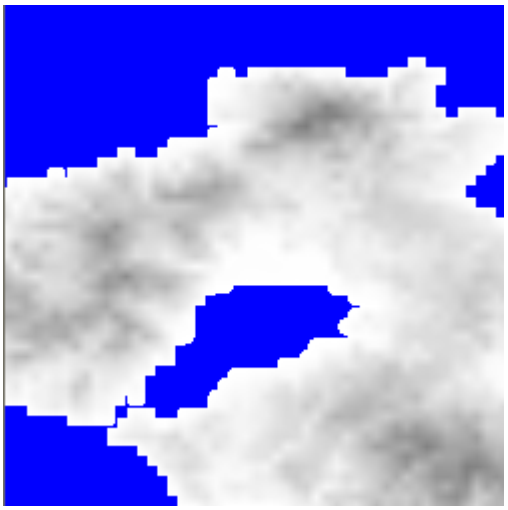
The image at item (b) is the output image that the nearest neighbor algorithm was applied. As we can see on the image, it is obvious that the image has blocking effect and aliasing. The blocking effect can be easily recognized from image, in the land area. Also, there exist aliasing, especially on the border of sea and land.



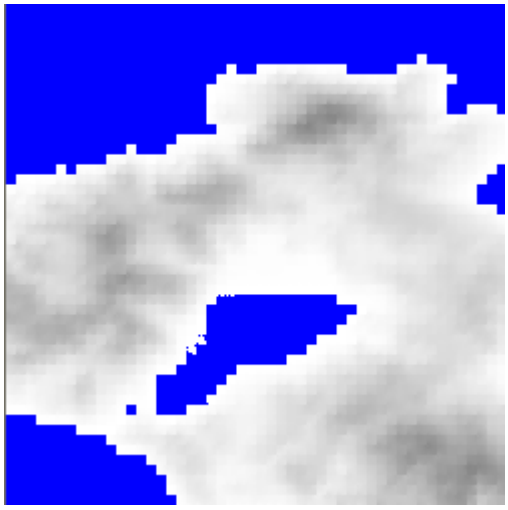
a) 120x120 resolution (50x50 in black frame area)



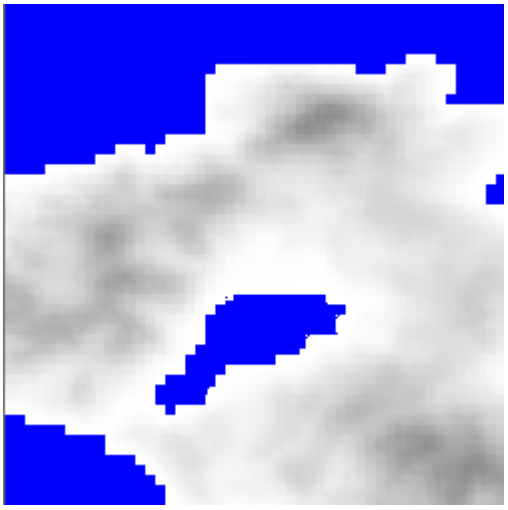
b) Nearest Neighbor (250x250)



c) Bilinear (250x250)



d) 4 - Texel Weighted Average (250x250)



e) 8 - Texel Weighted Average (250x250)

Figure 3.13 - The comparison of the interpolation methods at zoom-in

Item (c) contains an image of the same area, in which the bilinear interpolation algorithm was used to generate. Comparing with item (b), bilinear provides smoother image output, that is, no blocking effect on the image. This smoothness is even at the border of sea and land. However, the aliasing is still a problem. There is an important thing to note that, at the borders, the land area is running over the sea area, which is not the case in real image (a).

Item (d) includes the image that is produced by applying one of our proposed algorithms, which is 4-texel weighted average interpolation. This algorithm results in a blocking and aliasing effects. In case of blocking effect, this algorithm is better than nearest neighbor, because there is smoothness in the output image. Moreover, the aliasing does not exist as much as the case in the first two algorithms discussed above. Nevertheless, there is another effect in the image, which is shimmering. If we look at carefully the image, we can see the shimmers at the start points of the blocks that correspond to a texel in the DTED grid. The overlap of lands on the sea can be obviously recognized from the image such that the water gap in the south-west of the island is combined.

The other interpolation method that we proposed is 8-texel weighted average, which resulted the image at item (e). The blocking (less than nearest neighbor and 4-texel weighted average) and aliasing (less then nearest neighbor and bilinear, more than 4-texel weighted average) effects still exist in the image. In contrast to the other three, this method has more blurring effect. The invasion of lands over seas is much more than others, that is to say, for example, the bay in the island became completely a lake.

In order to compare the view quality of images generated by using the interpolation algorithms, the Root Mean Square Error (RMSE) of generated images were calculated and used. Figure 3.14 gives a sample set of RMSE results that are calculated from the grid cells covering the areas at the same latitude, i.e. 39N, and longitudes between 27E and 35E. These values are calculated for

zoomed-in images that are produced by scaling at 1000%. Comparing the RMSE values in the figure, we can conclude that 4-texel weighted average method gives the best result, whereas nearest neighbor gives the worst. For example, in the grid cell 39N27E, the RMSE values for nearest neighbor, bilinear, 4-texel weighted average and 8-texel weighted average methods are 78.57, 66.14, 53.28 and 54.84 respectively. As we can see in the figure, the sorting of interpolation methods with respect to RMSE values is the same in all areas. As a result, for zoom-in operations, the view quality of the interpolation algorithms (from best to worst) is in order of 4-texel weighted average, 8-texel weighted average, bilinear and nearest neighbor.

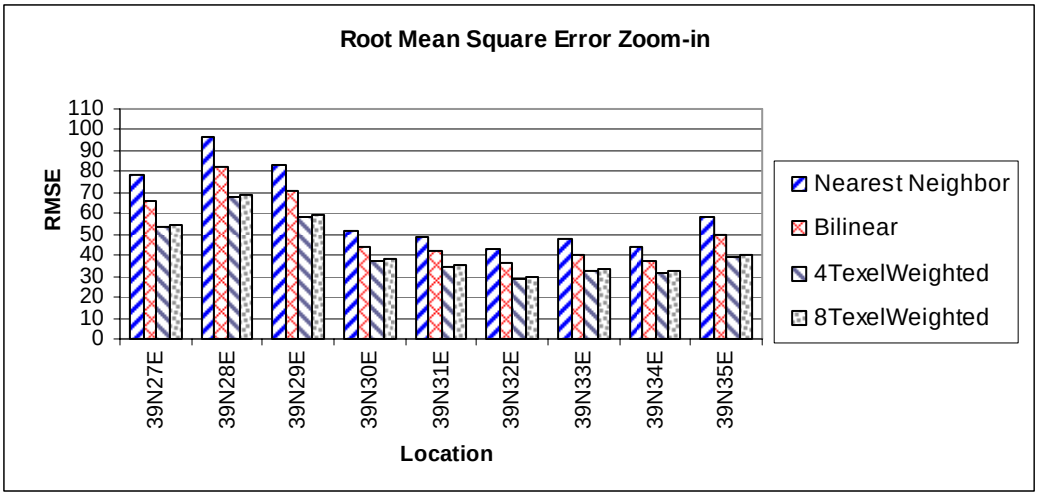


Figure 3.14 – RMSEs of the algorithms in some locations (zoom-in)

3.4.2 Zoom-Out

In Figure 3.15, there are some images depicted so that we can compare view quality of the algorithms when the DTED grid cells are zoomed-out. The table contains 5 images; a) an image that is not resampled, b) an image that is resampled with nearest neighbor, c) an image that is resampled with bilinear, d)

an image that is resampled with 4-texel weighted average, and e) an image that is resampled with 8-texel weighted average. All resampled images are minified at 70%, so they are down-sampled, and they represent the area that is given in first image (a). The image at item (a) is a visual representation of 1-degree DTED grid cell and there is no resampling as in Figure 3.13-a.

The artifacts in the down-sampled (zoom-out) images are not as clear as the up-sampled (zoom-in) images. Comparing with the zoom-in operations, there is no blocking in the images generated by zoom-out operations, so we observe smooth images for all methods. Aliasing exists in all methods, but getting disappeared when zoom-out goes further. However, there are still some degradations. Blurring effect can be considered only in the image that is produced by using 8-texel weighted average interpolation method (e). The overlap of land areas over the sea areas can be observed in all output images, in the increasing order of nearest neighbor, bilinear, 4-texel weighted average, and 8-texel weighted average respectively.

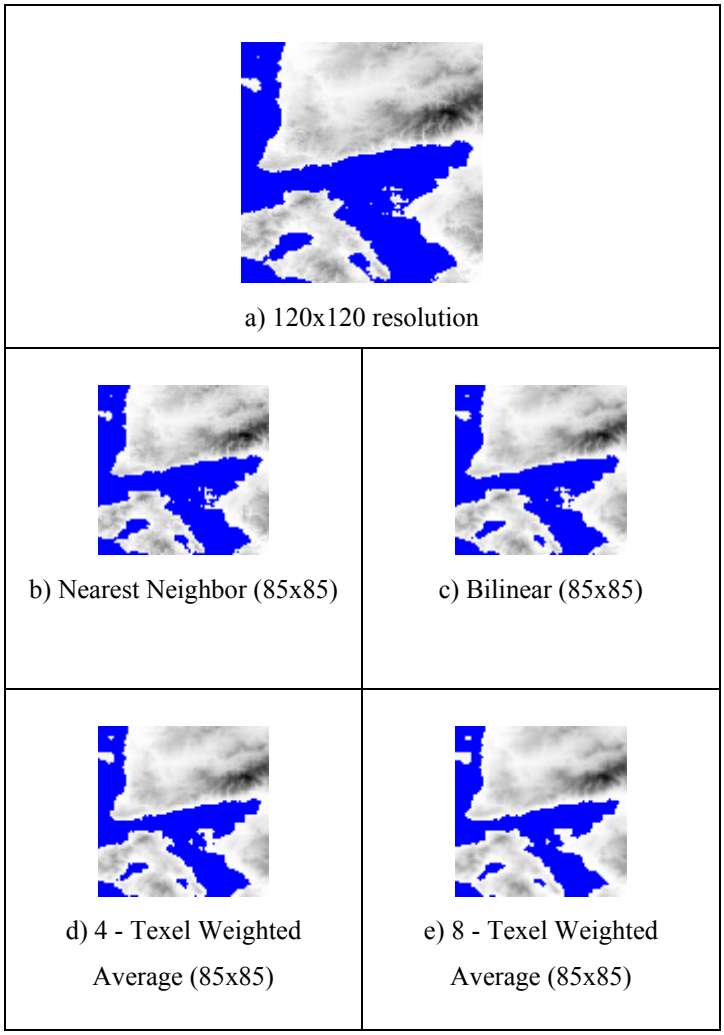


Figure 3.15 – The comparison of the interpolation methods in zoom-out

The RMSEs of the generated images were calculated and compared. Figure 3.16 contains the chart of computed RMSEs of the same areas as in Figure 3.14. In this case the images are produced for zoom-out operations, which are scaled at 1/10. For instance, in the grid cell 39N27E, the RMSE values for nearest neighbor, bilinear, 4-texel weighted average and 8-texel weighted average methods are 74,68, 74,68, 73,86 and 73.52 respectively. As we can understand from these values, and all other RMSE values from the figure, they are very close to each other. Moreover, the differences can be negligible. As a conclusion, for zoom-out operations, the algorithms give the similar results for the view quality of the produced images.

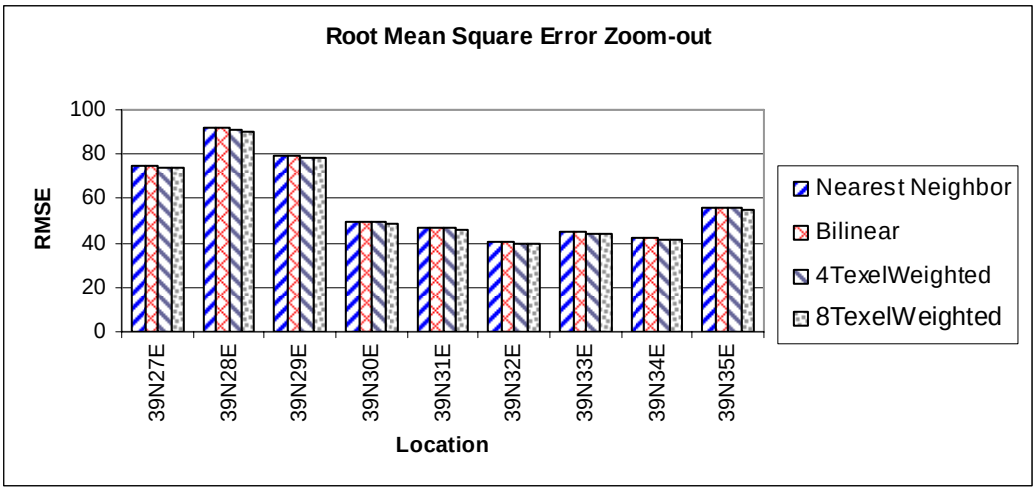
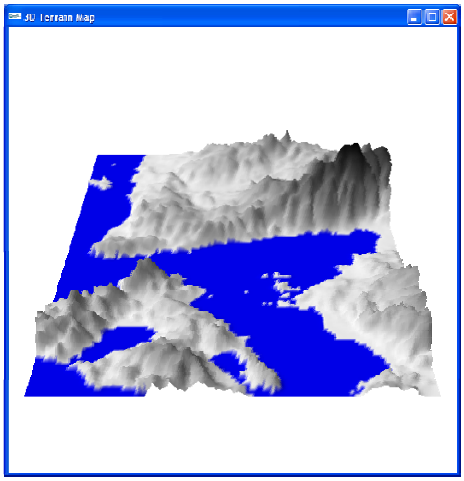


Figure 3.16 - RMSEs of the algorithms in some locations (zoom-out)

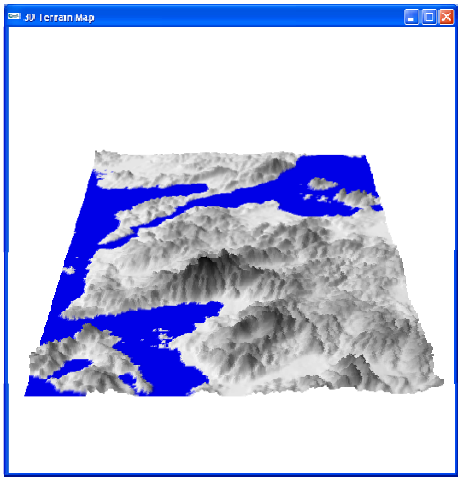
3.5 3D Image Generation

One of the main issues in this work is 3D graphical display of the output images. The standard graphic library OpenGL was used for this purposes. The window and I/O operations are maintained by GLUT, the utility toolkit of OpenGL. Since both OpenGL and GLUT are standard in all platforms, the 3D drawing part of the software is also platform independent.

The main source of the 3D programming is the 2D image generated from the DTED grid cells by applying one of the interpolation algorithms. Since grayscale value of a pixel in 2D image represents an elevation of a point, the 3D image was generated from corresponding 2D image. Figure 3.17 shows two sample 3D images; first one (a) generated from 1 DTED grid cell, and the second (b) generated from 4 DTED grid cells.



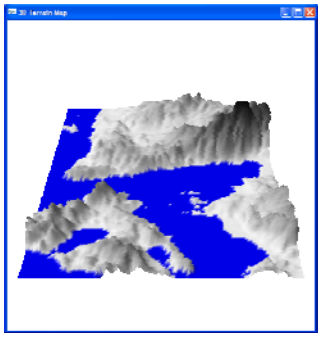
a) 3D image of one DTED grid cell



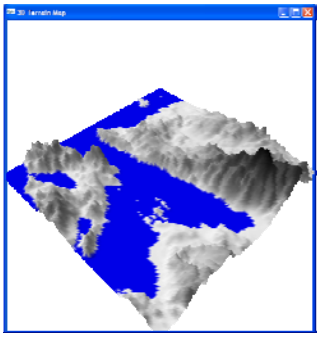
b) 3D image of 4 DTED grid cell

Figure 3.17 – Sample 3D images generated from DTED grids

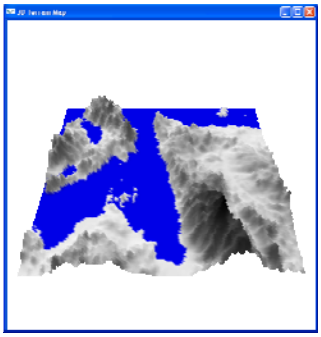
The generated 3D image can be rotated at any one of x, y and z axes. For example, the images in Figure 3.17 were rotated around x-axis about 45 degrees. This makes the images more feasible to view in depth. Furthermore, the images can be rotated around y-axis at any level of degrees. Figure 3.18 illustrates the views obtained by rotating the output image around y-axis, in addition to rotating around x-axis. The angle difference between consecutive images is 45 degrees.



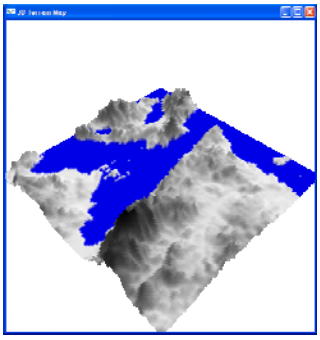
a) 0 degrees



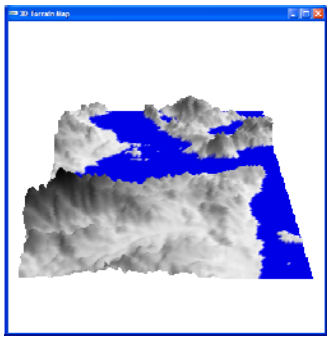
b) 45 degrees



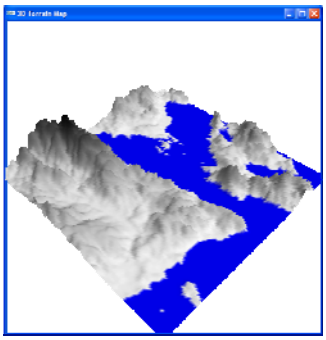
c) 90 degrees



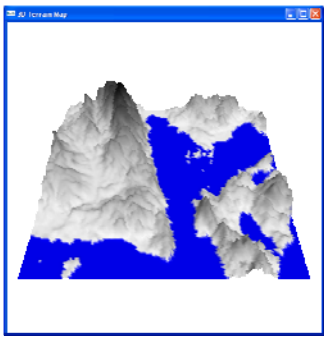
d) 135 degrees



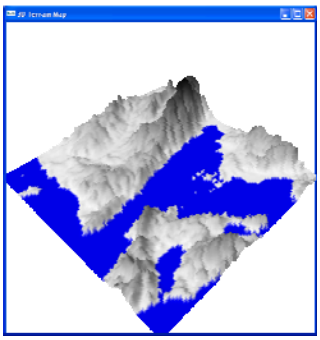
e) 180 degrees



f) 225 degrees



g) 270 degrees



h) 315 degrees

Figure 3.18 – Sample views from a 1 DTED grid cell image rotated around y-axis

3.6 How To Draw 3D Image

As mentioned before the 3D images were developed with standard OpenGL graphic library and its utility toolkit GLUT. The development steps can be listed roughly as follows:

- Generate the image matrix with x, y, and height values from the produced grayscale 2D image,
- Calculate the new height values for each point, by interpolating the heights into defined new range, to produce smoother (not steep in height) image,
- Compute the normals for each point in the image matrix,
- Initialize the GLUT, set the display modes (GLUT_DOUBLE, GLUT_RGB, GLUT_DEPTH), window size (600x600),
- Create the window to draw on,
- Initialize OpenGL rendering parameters by enabling; GL_DEPTH_TEST, GL_COLOR_MATERIAL, GL_LIGHTING, GL_LIGHT0, GL_NORMALIZE, and setting the shade model to GL_SMOOTH,
- Define the keyboard function to handle key presses such as for zoom-in, zoom-out,
- Define the timer function used for rotating the image,
- Define the reshape function to set view angle, projection matrix values etc.
- Define the display function;
    a. Set matrix mode to GL_MODELVIEW,
    b. Set the transformation matrix by assigning translation, rotation and scaling factors in all axes,

c. Set the light model to GL_LIGHT_MODEL_AMBIENT, source to GL_LIGHT0 and location, and color
d. Set the geometric primitive type to GL_TRIANGLE_STRIP,
e. For each point; find the grayscale value from height, set calculated normal, and vertex values (x, z, and height),

- Register the display, keyboard, and reshape functions,
- Enter event processing loop, displaying the drawn scene.

CHAPTER 4

EXPERIMENTAL RESULTS

In this chapter, we will explain the experiments performed to test the interpolation methods, the graphical charts of the results acquired from test cases, and the comparison of the results with respect to execution time performance of the methods. Moreover, the execution times of 2D and 3D images are also given and the results are discussed.

4.1 Description of Experiment Environment

The DTED Level 0 data is used as an input data set. This level DTED can be obtained from internet. The National Imagery and Mapping Agency (NIMA) provides all available DTED Level 0 data on its web site, which can be downloaded without any charge. We downloaded all the data covering the locations of Turkey and used to perform all experiments of this thesis.

The DTED Level 0 data file contains an area of 1-degree grid cell. Since Turkey is located between latitudes 36° - 42° North, both latitude and longitude intervals at that latitude level is 30 arc seconds (see Table 1.2). So, the resolution of the rectangular data grid for DTED Level 0 files is at 120x120.

The development environment was Microsoft Visual C++ 2008 with benefiting from MFC, GDI+ APIs and OpenGL library. The operating system was Windows

XP with Service Pack 2 installed. The timing measurements were performed on a laptop computer, which has following hardware specifications;

- Intel Core 2 T5500 1.66GHz CPU,
- 997MHz 0.99GB RAM,
- Intel GMA 950 224 MB graphic card.

The DTED data files that will be used in time measurement were first read into memory because of speed consideration, as explained in Chapter 3. After reading all file in cache memory, the time measurement was performed at runtime for interpolation methods and displayed images.

4.2 Execution Time Performances of Interpolation Algorithms

We performed the following experiments to compare the interpolation methods in terms of execution time performance. To measure the execution time, at least 10 samples are collected in all experiments. By using those samples, the average values were calculated and these averages were used as execution time for a method at a specific resolution.

4.2.1 Single Grid File Display Experiment

In this experiment, execution time of each interpolation method is calculated by working on one DTED data file at a time. The calculations are separated into two cases;

1) zoom-out operations,
2) zoom-in operations.

For zoom-out operations, the resolutions at 30x30, 37x37, 60x60, and 75x75 are used to down-resample the input data. The execution times of the interpolation methods are computed at the given resolutions. The results are depicted in Figure 4.1-a.

For the case of zoom-in operations, we used the 150x150, 300x300, 450x450, and 600x600 resolution values. The results of those operations are given in Figure 4.1-b.

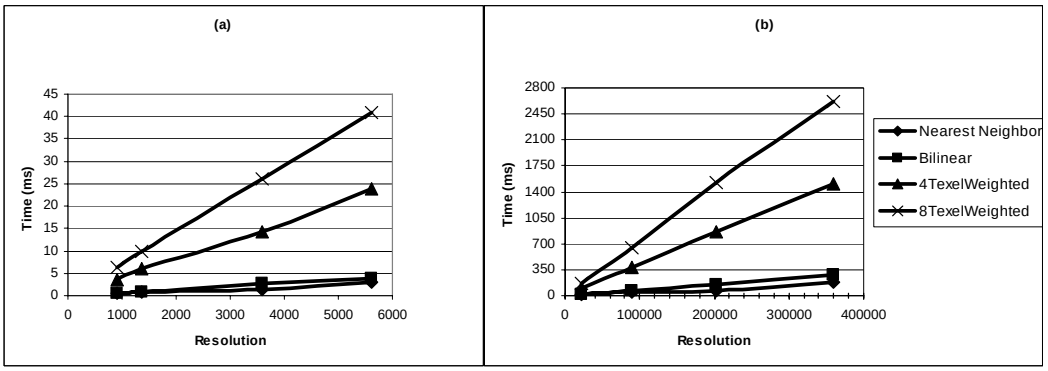


Figure 4.1 – The execution time (1-grid cell) of the algorithms, a) Zoom-out, b) Zoom-in

As can be seen from the figures, when the resolution increases the execution time also increases. This increment is directly proportional to the increase in the resolution. For example, look at the execution times of bilinear interpolation at resolutions 150x150 and 300x300 in Figure 4.1-b. The execution time at resolution 150x150 is 17.4 ms, while it is 71.3 ms at resolution 300x300. So, the latter takes approximately 4 times longer than the first one. If we consider the number of pixels at these resolutions, 300x300 is 4 times 150x150. Obviously, there is a parallelism between the increase in resolution and increase in the execution time. This is the same case in all zoom-in and zoom-out operations.

Each interpolation methods have different execution times at a specific resolution. As we can deduct from Figure 4.1, the fastest algorithm is nearest neighbor interpolation. Because of its simplicity, it is very easy to implement and it requires less number of operations compared to others, just takes the value of nearest texel.

The bilinear interpolation algorithm is a little slower than nearest neighbor. This algorithm first interpolates the values on y-axis, and then interpolates on x-axis. So, value of new point is found after those calculations. This causes more processing and results in longer execution time.

The algorithm in the third order is our first proposed one, 4-texel weighted average. Comparing to the nearest neighbor and bilinear, this algorithm takes relatively much time. The first reason is that the number of texels in the calculation is 5 in this algorithm whilst it is 4 in bilinear. Secondly, we calculate the Euclidean distance to all 5 texels in order to find the weighting factors of each one, and then compute result with using all texel values and their weightings.

The relatively worst algorithm in execution time performance is proposed 8-texel weighted average interpolation algorithm. The calculations are similar to 4-texel weighted average algorithm, but there are 9 texels used. So, this increase in the number of texels approximately doubles the execution time for this algorithm.

4.2.2 Four Grid Files Display Experiment

In this experiment, four data files were loaded at each time and the execution time computation was performed on these four files' data. We were investigating the behaviors of each algorithm when there is an increase in number of processed DTED file data at a time. The results are shown on the Figure 4.2. The execution

time values for zoom-out operations are depicted in Figure 4.2-a, and he execution time values for zoom-in operations are depicted in Figure 4.2-b as well. As in the first experiment, the fastest algorithm is nearest neighbor, the slowest one is the 8-texel weighted average. The ordering did not change in this experiment. As the resolution increases, the execution time increases with the same factor too.

If we examine the execution time of 4-file data, we can see that the time increases as expected. For example, the execution time of bilinear interpolation algorithm at resolution 75x75 is 18.5 in this experiment; it was 3.8 in the first experiment.

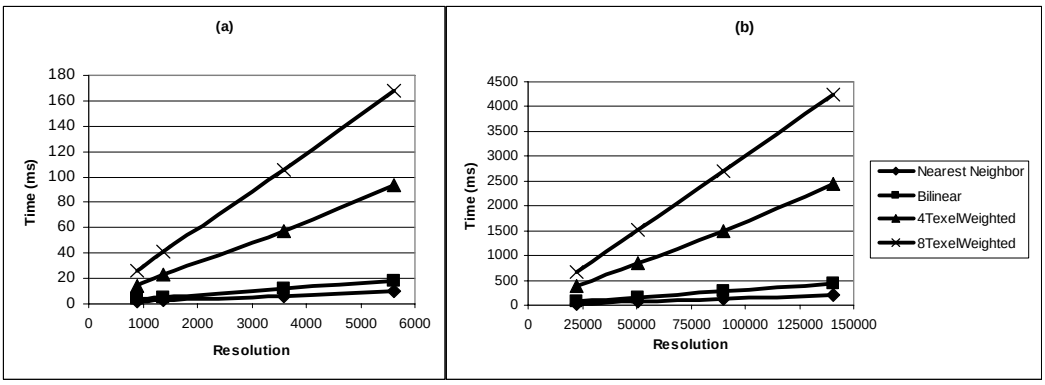


Figure 4.2 – The execution time (4-grid cell) of the algorithms, a) Zoom-out, b) Zoom-in

4.2.3 All of Turkey Grid Files Display Experiment

The third experiment was performed on the data that covers all Turkey geographic area. Since Turkey is approximately located in latitudes 36° - 42° North and longitude 26° - 45° East, there are 7x20 (140) 1-degree files containing DTED data. All those files were first loaded into program, and then the resulting images were drawn. The time values were calculated while applying interpolation

methods on all of the loaded files at the same time. Figure 4.3 gives the results of these operations for both zoom-out (a) and zoom-in (b). As we can see from this figure, the execution time increases linearly, as the resolution increases.

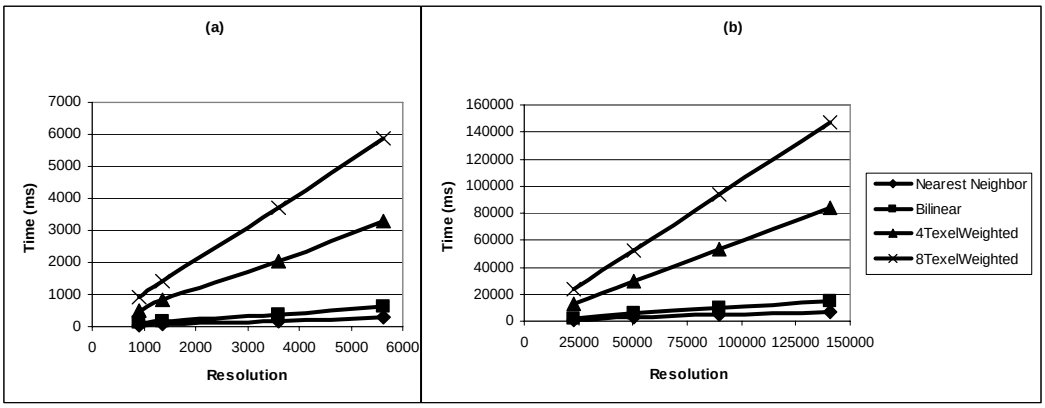


Figure 4.3 – The execution time (all of Turkey) of the algorithms, a) Zoom-out, b) Zoom-in

4.3 2D Timing Results with GDI+

We ran the software with various set of DTED grid cells and measured the time to draw the resulting image matrix with GDI+ in 2D. The results of the measurements are given in Table 4.1 below. The GDI+ can draw the image of 1-grid DTED file at the rate of about 75 frames-per-second. Also, there is a little increase in execution time when the number of grid cells increases.

Table 4.1 - 2D GDI+ drawing times for different size regions

1 Grid Drawing Time		4 Grid Drawing Time		9 Grid Drawing Time		16 Grid Drawing Time	
Avg time	Std. dev.	Avg time	Std. dev.	Avg time	Std. dev.	Avg time	Std. dev.
13.4	0.1	13.1	0.4	13.5	0,2	14.6	0,2

The execution time performance of GDI+ was measured at several zoom scales. Figure 4.4 contains the chart of zoom-in times for 1 grid, 4 grid, 9 grid and 16 grid DTED data, whereas Figure 4.5 has the chart of zoom-out times for the same data set. As we can see on the figure, the zoom-in times for different scales fluctuate around a specific value. For instance, when we have a close look at the values at different zoom-in scales for 9 DTED grid cells, we see values like 13.1, 13.4, 13.2 etc., but mostly 13.3 (each value is average of 10 runs). So, the average zoom-in time for 9 DTED grid cells can be calculated as 13.3 milliseconds. We can say that there is no regular increase or decrease in execution time when the level of zoom-in scale increases. They are around the same values, almost constant.

For zoom-out, execution time performances of the GDI+ results are different. When the zoom-out scale increases, the execution time decreases. The distribution of the values on the chart is logarithmic, indicating that the execution times are decreasing quickly as image size is reduced with zoom-out operation.

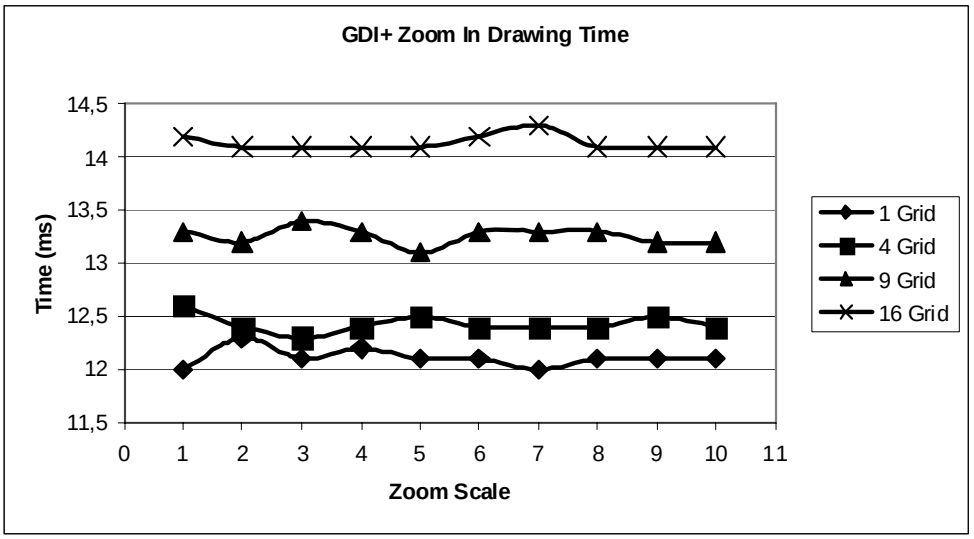


Figure 4.4 - GDI+ zoom-in time

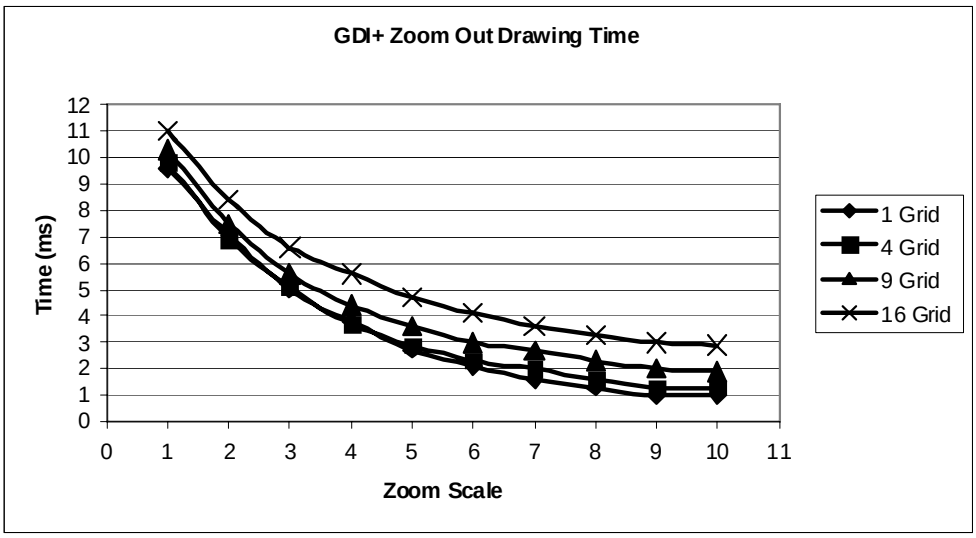


Figure 4.5 – GDI+ zoom-out time

4.4 3D Timing Results with OpenGL

The DTED 3D images were displayed using OpenGL and execution times for several resolutions were measured. Table 4.2 contains these timing results, where 1 grid, 4 grids, 9 grids and 16 grids having 26.5 ms (std = 0.1), 97.4 ms (std = 0.4), 277.4 ms (std = 11.9) and 491.9 ms (std = 1.4) execution times respectively. So, the images can be drawn on the screen with 40 to 2 frames-per-second rates.

Table 4.2 – 3D OpenGL drawing time for different size regions

1 Grid Drawing Time		4 Grid Drawing Time		9 Grid Drawing Time		16 Grid Drawing Time	
Avg time	Std. dev.	Avg time	Std. dev.	Avg time	Std. dev.	Avg time	Std. dev.
26.5	0.1	97.4	0.4	277.4	11.9	491.9	1.4

The DTED 3D image has been rotated for 360 degree using 1-degree, 10-degrees and 20-degrees increments, where average timing and standard deviations are 26.5 (std = 0.2), 26.5 (std = 0.3) and 26.4 (std = 0.3) respectively (Table 4.3).

Table 4.3 – 3D rotation times of 1 DTED grid for different degree differences

1 Grid 1° Rotation Time		1 Grid 10° Rotation Time		1 Grid 20° Rotation Time	
Avg time	Std. Dev.	Avg time	Std. Dev.	Avg time	Std. Dev.
26.5	0.2	26.5	0.3	26.4	0.3

The zoom-in and zoom-out timings for DTED 3D images were measured and the results are given in Figure 4.6 and Figure 4.7 respectively. As shown on the figures, the zoom-in times do not change when zooming scale increases. In contrasts, drawing time increases very little, 1% or 5%, in zoom-out operations.

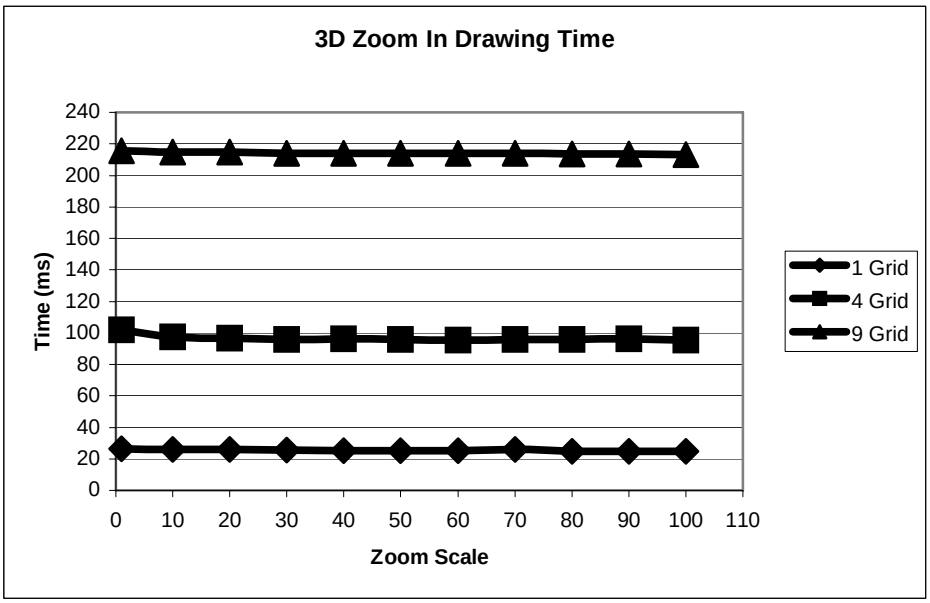


Figure 4.6 – Zoom in drawing time of 3D images

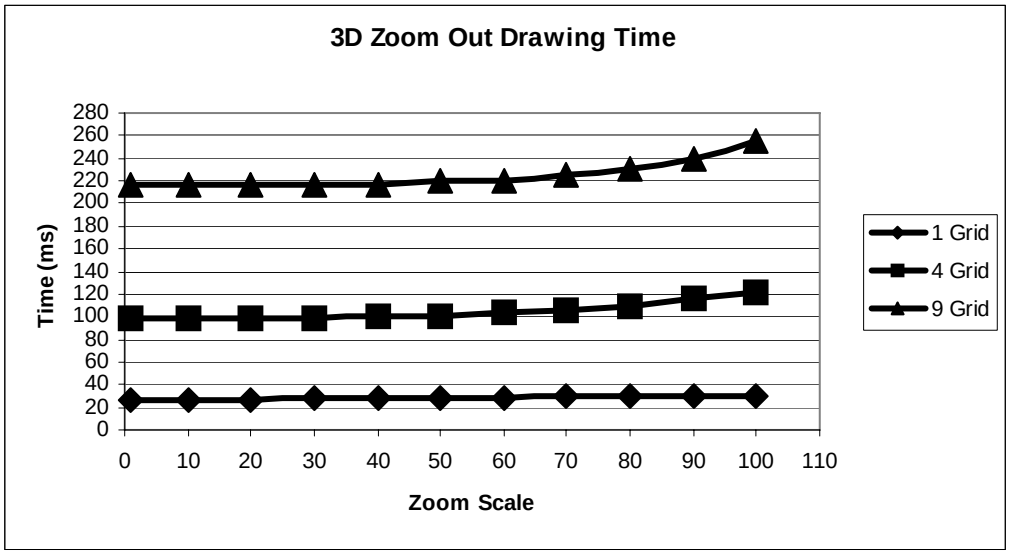


Figure 4.7 – Zoom out drawing time of 3D images

CHAPTER 5

CONCLUSION

Digital Terrain Elevation Data (DTED) is one of the raster data representations, which is utilized in almost all Geographic Information System (GIS) applications. Generally, the visualization of the DTED data is conducted by using digital images. Those images are not only displaying the DTED data, but also they provide entry point to the geospatial data representation like keeping geographical coordinates, elevation values of a point on Earth.

Each DTED file contains a 1-degree grid cell data, which corresponds to a geographical area on the Earth. This data is used to generate map of the concerned location in a GIS application. Every pixel on the map corresponds to a spatial grid location and carries an intensity level that represents an elevation value.

In order to display a DTED tile on the screen, depending on the resolution of the generated images, the data needs to be interpolated. In case of zoom-in and zoom-out operations, the interpolation should be applied on the new generated pixel point that has no correspondent in the real data. We applied the following interpolation methods and compared their performances; nearest neighbor, bilinear interpolation and the proposed 4-texel weighted average, and 8-texel weighted average.

The algorithms provide different results in terms of view quality and execution time performances, there is a trade-off choosing the better. The simplest and fastest algorithm is nearest neighbor. It provides good view quality results in zoom-out operations, especially at very small resolutions. However, there is a half pixel location shift of the texel points, and the blocking and aliasing effects are very evident in zoom-in operations. Bilinear algorithm brings out smooth images in all zoom-in and zoom-out operations. Also, it is relatively faster algorithm. Nevertheless, it is not good at differentiating sea and land in the borders; that is, land regions override sea regions. The other two proposed algorithms are slower than the first two ones. The 8-texel weighted average is the slowest among the others, but provides the smoothest output images. The 4-texel weighted average is faster than 8-texel weighted average, but it has more blocking and aliasing effects. However, looking at the RMSE calculations, the new proposed methods give more accurate results than the others.

Another thing to note in the experiment results is that when resolution of the images increases, the execution time of the algorithms increases at the same proportion. All four algorithms have the same behavior.

5.1 Recommendations and Future Work

In this thesis, the interpolation methods selected to compare are based on linear interpolations. However, there are plenty of algorithms that are based on polynomial interpolations such as Cubic B-Spline, Bezier surfaces, and Quadratics. They can be some candidates to implement and used for comparison as new alternatives to the ones that we used in this thesis. In addition, there are anti-aliasing algorithms, which can be applied on the output images to remove the aliasing effect [Tomas2008][Chan2005].

Since image processing requires high-performance computers, the graphical drawings of the output images could be handled by GPU of advanced graphics cards, which will result in higher speed performance in execution time [Lau2006].

The application could be improved by applying texture mapping of satellite images and aerial photographs on the generated 3D DTED images [Royan2007] In addition, flyover operations on the textured 3D terrain could be developed [Dias2003]. A sample flyover view taken from the animation of Candor Chasma canyons generated by using HiRISE DTM [HiRISE2010] is given in Figure 5.1 below [TechRadar2010].

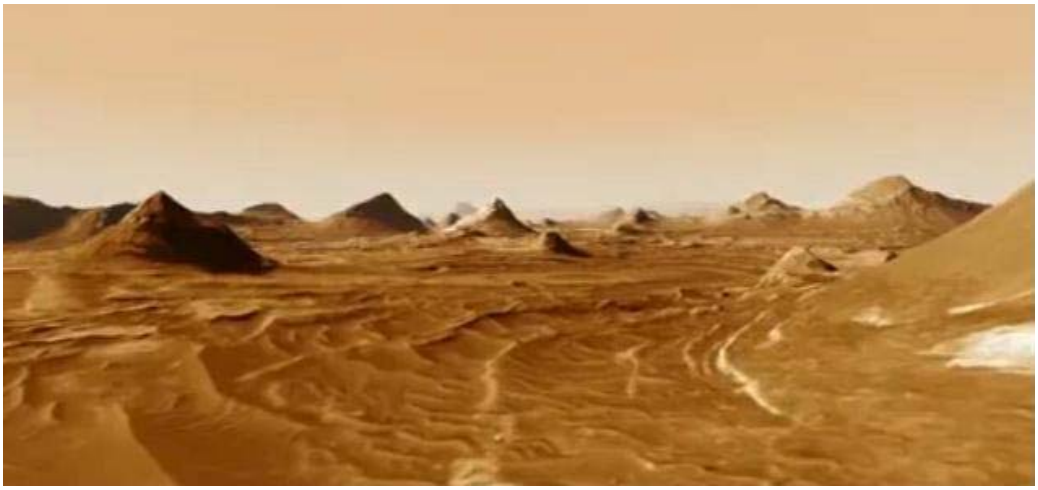


Figure 5.1 – A flyover view of 3D martian terrain

REFERENCES

[Ayhan2001] Ayhan, M.E., O. Lenk, C. Demir, A. Kılıçoğlu, M.Kahveci, A.Türkezer, M.Ocak, M.Açıkgöz, A.Yıldırım, B.Aktuğ, Y.S.Şengün, A.İ.Kurt, O.Fırat, Türkiye Ulusal Temel GPS Ağı-1999 (TUTGA-99), Technical Report, Hrt.Gn.K.lığı, Ankara, 2001.

[Bernstein1976] R. Bernstein, ‘‘Digital image processing of earth observation sensor data”, IBM J. Res. Develop., vol. 20, pp. 40-57,1976.

[Bildirici2007] Bildirici, İ.Ö., Üstün, A., Uluğtekin, N., Selvi, H.Z, Abbak, R.A., Buğdaycı, İ., Doğru, A.Ö.(2007) SRTM Data in Turkey: Void Filling Strategy and Accuracy Assessment, Middle East Spatial Technology 4th Conference & Exhibition MEST 2007.

[Chan2005] Chan, Eric, “Fast Prefiltered Lines”, GPU Gems 2: Programming Techniques for High-Performance Graphics and General Purpose Computation, Matt Pharr and Randima Fernando, Addison-Wesley, 2005.

[Dias2003] Dias, E.S., Velde, R.J. van de, Nobre, E. & Scholten, H.J., Virtual Landscape Bridging the GAP between Spatial Perception and Spatial Information, Proceedings of the 21st International Cartographic Conference, (pp. 626-635). Durban, South Africa: ICC., 2003.

[Hofmann] Hofmann-Wellenhof, B., Lichtenegger, H., Wasle, E., GNSS – Global Navigation Satellite Systems (GPS, GLONASS, Galileo and more), Springer, Wien, NewYork, 2008.

[Hou1978] H. S. Hou and H. C. Andrews, “Cubic splines for image interpolation and digital filtering,” IEEE Trans. Acoust., Speech, Signal Processing, vol ASSP-26, pp 508-517, 1978.

[Intermap2004] Intermap Technologies, Void Fill Of SRTM Elevation Data - Principles, Processes And Performance, 2004.

[Jain1988] A.K. Jain, Fundamentals of Digital Image Processing, Prentice-Hall, 1988.

[Keys1981] R. G. Keys, “Cubic convolution interpolation for digital image processing", IEEE Trans. Acoust., Speech, Sig. Proc., Vol. 29, No. 6, pp.1153-1160, Dec, 1981.

[Kidner1999] D. Kidner, M. Dorey, and D. Smith., What's the point? interpolation and extrapolation with a regular grid DEM, Proceedings of GeoComputation’99, Virginia, USA, July 1999.

[Lau2006] C. W. Lau and A. S. Parenteau, Fast zoom-adaptable anti-aliasing of lines using a graphics processing unit, Adobe Systems, Inc., San Jose, CA, US, 2006.

[NIMA2000] National Imagery and Mapping Agency (NIMA), Digital Terrain Elevation Data (DTED), Tech. Rep. MILPRF-89020B, 2000.

[Parker1983] J. A. Parker, R. V. Kenyon, and D. E. Droxel, Comparison of interpolating methods for image resampling, IEEE Trans. Med. imaging MI-2, No. 1, 1983, 31-39.

[Philippe2000] Philippe Thévenaz , Thierry Blu , Michael Unser, Image interpolation and resampling, Handbook of medical imaging, Academic Press, Inc., Orlando, FL, 2000

[Rabus2003] Rabus, B., M. Eineder, A. Roth, and R. Bamler (2003), The shuttle radar topography mission-a new class of digital elevation models acquired by spaceborne radar. ISPRS Journal of Photogrammetry and Remote Sensing. v57 i4. 241-262.

[Rifman1973] Rifman, S.S., “Digital rectification of ERTS multispectral imagery”, in Proc. Symp. Significant Results Obtained from the earth Resources Technology Satellite-1, 1973, vol -1 sec, B NASA SP-327 , pp 1131-1142.

[Royan2007] Royan, J., Gioia, P., Cavagna, R., Bouville, C., "Network-Based Visualization of 3D Landscapes and City Models," IEEE Computer Graphics and Applications, vol. 27, no. 6, pp. 70-79, Nov./Dec. 2007.

[Schafer1973] R. W. Schafer and L. R. Rabiner, “A digital signal processing approach to interpolation", Proc. IEEE, Vol. 61, No. 6, Jun, 1973, pp.692-702.

[Tomas2008] Tomas Akenine-Möller, Eric Haines and Naty Hoffman. Real-Time Rendering, 3rd edition, ISBN 978-1-56881-424-7, A.K. Peters, Ltd., 2008.

[Torge2001] Torge, W., Geodesy, 3nd Edition. W. de Gruyter, Berlin-New York, 2001.

[Watson1992] D.F. Watson, Contouring: A Guide to the Analysis and Display of Spatial Data. New York: Pergamon Press, 1992.

[Wolberg1990] G. Wolberg, “Digital Image Warping”, IEEE Computer Society Press Monograph, IEEE Computer Society Press, 10662 Los Vaqueros Circle, P.O.Box 3013, Los Alamitos, CA 90720-1264,USA; 1990.

Online References:

[GDI2010] GDI+, Last Date Accessed: 13/01/2010, Owner: Microsoft Corporation, Url: http://msdn.microsoft.com/en-us/library/ms533798(VS.85).aspx

[GE2010] Google Earth, Last Date Accessed: 19/01/2010, Owner: Google Corporation, Url: http://earth.google.com/

[HiRISE2010] High Resolution Imaging Science Experiment, Last Date Accessed: 23/03/2010, Owner: The University of Arizona, Url: http://hirise.lpl.arizona.edu/

[OpenGL2010] OpenGL – The Industry Standard for High Performance Graphics, Last Date Accessed: 13/03/2010, Owner: SGI, Url: http://www.opengl.org

[TechRadar2010] Fly across Mars in virtual 3D on Youtube, Last Date Accessed: 23/03/2010, Owner: Future Publishing Limited, Url: http://www.techradar.com/news/photography-video-capture/fly-across-the-canyons-of-mars-in-virtual-3d-on-youtube-677087

[Timers2010] Timer, Last Date Accessed: 15/01/2010, Owner: Microsoft Corporation, Url: http://msdn.microsoft.com/en-us/library/ms632592(VS.85).aspx

[VTP2009] Visual Terrain Project, Last Date Accessed: 13/01/2010, Owner: he.net, Url: http://vterrain.org

[Wiki2010] Wikipedia: The Free Encyclopedia, Last Date Accessed:10/01/2010, Owner: Wikimedia Foundation, Inc., Url: http://en.wikipedia.org