

Research on Real-Time Mobile 3D Topography System

Weihui Dai

School of Management, Fudan University, Shanghai 200433, P.R.China

Email: whdai@fudan.edu.cn

Shuyi Liang and Wei Yan

School of Management {software}, Fudan University, Shanghai 200433, P.R.China

Email: {062025047, 043053243}@fudan.edu.cn

Abstract—Real-Time Mobile 3D Topography System is a GIS (Geographic Information System) based on Client Server. In our daily life, this system could be useful in the traveling, mining, and etc.. In this paper, we developed a Real-Time Mobile 3D Topography System for embedded system. This system is consisted by a Server and a client. The client based on Intel Sitsang, can orientate the position through the GPS module and use a lot of wireless network through the Libra. The system can be applied to the domain of mountaineering, exploration, ciceroni, navigation and so on.

Index Terms—real-time, mobile, 3D, GIS

I. INTRODUCTION

The Global Positioning System (GPS) is a global navigation satellite system (GNSS) developed by the United States Department of Defense and managed by the United States Air Force 50th Space Wing. It is the only fully functional GNSS in the world, can be used freely, and is often used by civilians for navigation purposes. The establishment of the GPS brings significant change to the navigation and orientation technology, and solves the problem of navigation and orientation on earth that satisfies the needs of different users.

Topography data are complicated 3D data. So if we display it on a 2 dimension plane, we should figure it by the way that based in non-dimensional manner such as color, and contour line. But when the gurgitation is complicated, this method can not show the 3D information visually and directly. As in most daily application such as city traffic map, longitude and latitude sometimes have more information than altitude, thus this localization is not distinct in daily application.

In this paper, we design the real-time mobile 3D topography system that aims at the character and restrict of the embedded mobile system. The research includes:

Topography database: It offers data support to the

entire system, which conserves the data related to the topography and some user information. The main research contains topography data abstract and user management.

Data service software: It is the flexible route between topography browser and database, offers data alternation between the browser and database, and shields the distributing realization of topography database. We must consider system's expansibility and developing cycle during the design and realize process. The main research contains the investigation and realization of the data transport protocols.

3D topography engine: The display of the 3D topography is the core problem of the system. The embedded platform of the system doesn't install 3D accelerate hardware, so all the 3D graphics must be finished by accelerate software. Considering the particularity of this 3D entity, we must develop an graph engine that has the function of 3D topography commutation and display. The engine bases on the sand tray model, and realizes the data modeling, illumination model with parameter, and graph transformation such as circumrotating and zooming of the 3D topography.

In the system, the server is made of Oracle DBMS and topography data server, which stores the topography data and serve special data of the view point as LOD (Level of Detail) model. The middleware which we named Libra (Lite Corba) formats the topography data and transfers it. Libra supports many air-interface including GPRS, CDMA, WLAN, 3G. The client based on Intel Sitsang, can orientate the position through the GPS module and use a lot of wireless network through the Libra, in our demo version, we adopt the WLAN. The operation system of the client is Linux. The topography browser uses the LOD model to process the topography data. It has the ability of low bandwidth, hard real time and high-precision.

The browser supports many types of GIS operations, such as zoom, angular transformation, real-time walkthrough, graphics rendering, and auto orientation. The system can be applied to the domain of

Manuscript received Feb 19, 2009; revised March 3, 2009; accepted March 20, 2009. This research was supported by National High-tech R & D Program (863 Program) of China (No.2008AA04Z127) and Shanghai Leading Academic Discipline Project (No.B210).

mountaineering, exploration, ciceroni, navigation and so on.

II. DATA STORAGE AND TRANSMISSION REALIZATION OF THE TOPOGRAPHY SYSTEM

According to the holistic design of the real-time mobile 3D topography system, we will describe the data storage and transmission realization of the topography system in detail to the topography database, data transmission service software and GPS driving module.

A. The Realization of the Topography Database

1) Topography data abstract:

Topography data is a series of lattice data. To a certain partial district, we can consider the topography of the area as the gurgitation of a plane. See Fig.1 as an example:

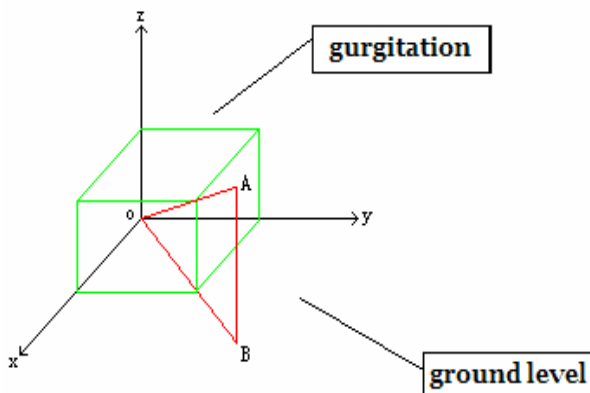


Figure 1. The Topography Description under 3D Coordinate

Based on this principle, we can scan the gurgitation of the partial district in grid. In the system, the scan space is 5 meter. See Fig.2:

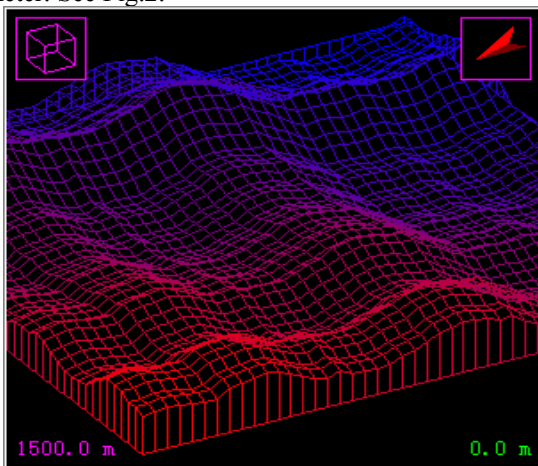


Figure 2. The Grid Scan of the Partial Topography

In this way, if the diameter is 500 meter, and the gradient is 15 degree, the maximum error is: $E_{max} = (5 \cdot 2 / 500) \cdot 100\% = 2\%$. To most grand topography, it is a minute error that can ignore. So we adopt plane gurgitation model and use grid mode to sample. When using grid mode to dispose the topography, its color is also scanned, and it is just the color of the scanning dot.

When the topography recurs, we use acme color difference mode to show the district color.

2) Topography database design

Table I shows the attributes of the sample dots in the topography data abstract model:

TABLE I. SIMPLE DISTRIBUTING TABLE

NAME	DATA SORT	DESCRIPTION
Longitude	Double Float	longitude
Latitude	Double Float	latitude
Altitude	Double Float	Height above sea level
RBG	Integer	The color of the scan dot

We can describe it in the below code:

```

Create table TTopography {
    Longitude double (not null),
    Latitude double (not null),
    Altitude double (not null),
    RBG int (not null)
};
    
```

To system operator, we must grant all the management authorization:

```

grant * on TTopography to <operator> identified by
<password>;
    
```

To common user, we just need to grant inquiry authorization:

```

grant select on TTopography to <user> identified by
<password>;
    
```

B. The Realization of the Data Service Software

1) Data service software interface

The data service software interface is realized by a Libra long-distance object. See the description as below:

```

//file: topography.idl
//describe: the interface define file of Topography
system
//author: TOMy
//time: 2007-03-04
//copy: lab
//copy right © TOMy, all rights reserved
Struct_Topography_Point {
    double longitude;
    double latitude;
    double resolve; //differentiate rate
    unsigned short size; //topography size
};
Struct_Topography_Data {
    double altitude; //height above sea level of the dot
    char r; // r heft of the dot color
    char g; // g heft of the dot color
    char b; // b heft of the dot color
};
interface_TopographyData {
    int getTopographyData (in
        struct_Topography_Point point, out
        vector<struct_Topography_Data> data);
};
    
```

This description file will be interpreted to Java object by Libra middleware tool, and be transferred by topography browser in service mode.

2) Libra middleware realization

The interface description language (IDL) of the Libra middleware abnegates a majority of characters that Corba seldom use. See the basic data sort that Libra supports in Table II:

TABLE II. SIMPLE DISTRIBUTING TABLE

DATA SORT	DESCRIPTION	BYTE
byte	Byte	1
char	ASCII character	1
short	Short integer	2
int	Integer	4
long	Long integer	8
float	Float	4
double	Double float	8

The interface description language (IDL) of the Libra middleware abnegates a majority of characters that Corba seldom use. See the basic data sort that Libra supports in Table II:

The data collect that topography database supports have two sorts: same kind data collect and different kind data collect. Same kind data collect indicate the collect composed by same data sorts that also have two sorts: array and vector. Array is the data collect that the data in it have fixed length, while vector is the data collect that the data in it have alterable length. The definition fashion of array in topography database is:

<data type> <array name>[array count];

The definition fashion of vector is:

vector<data type> <vector name>;

We call it Struct if the data collect composed by different kind of data. Struct can be composed by any legal data including base data sorts, array, vector and struct. The definition fashion of struct in topography database is:

```
Struct <struct name> {
    <data type> <data name>;
    <data type> <array name> [<array count>];
    vector <<data type>> <vector name>;
    struct <other defined struct> <name>;
    struct <other defined struct> [struct array name];
    struct <<other defined struct>> <struct vector name>;
    .....;
};
```

Struct can describe the more complicated structure.

We use interface to define the object interface in Libra:

```
interface <interface name> {
    <return type> <method name> (<param type>
    <data type> <data name>, .....;
};
```

<param type> has two options: in and out. They designate the entrance and exit parameter. The parameter can use both basic data sorts and data collect.

C. GPS Algorithm Module

GPS algorithm module solves the problem of how to comminute separate data frames from NMEA0183 data stream and distill GPS information from data frames. It has two steps:

Firstly, comminute separate data frame from data stream. NMEA0183 is a bunch data agreement based on ASCII character flow. All the data frames must begin with "\$", and end with "*"h'h'r'n'(*hh<CR><LF>). According to this rule, we can comminute integrated data frame. Fig.3 shows the algorithm flow chart. (This algorithm is realized by function: void GPSWindow::ReadSerialData())

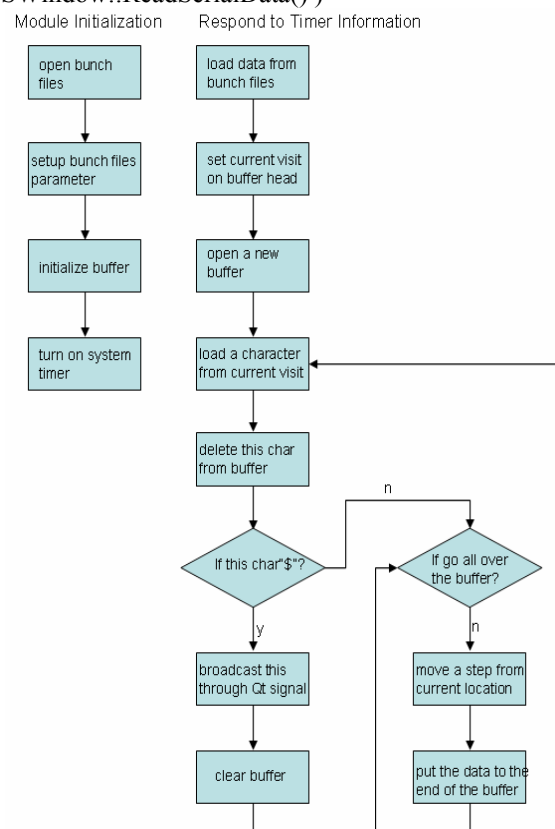


Figure 3. Flow chart of GPS algorithm module

Secondly, distill the orientation information from the frame NMEA0183. Besides the characteristic of frame NMEA0183 that mentioned formerly, it has some other important characteristic: There are a few fields in each frame, it is blocked off by ',', the first field is the mark symbol of the frame that composed by a group of capital letters, the symbol indicates the type of the frame. For example, \$GPSSV ('\$' is the frame head) expresses satellites information frame, each information frame has its fixed format. So when getting a data frame, we should estimate the type of the frame from the mark symbol first, and then gain other GPS information from the other frame's format. Table III to Table VII show the GPS frame type and its format:

TABLE III. RECOMMENDED MINIMUM SEPECIFIC GPS/TRANSIT DATA (RMC)

\$GPRMC,<1>,<2>,<3>,<4>,<5>,<6>,<7>,<8>,<9>,<10>,<11>,<12>*HH<CR><LF>	
<1>	Time UTC, hhmmss
<2>	Orientation status, 'A' for locked, 'V' for unlocked
<3>	Latitude, ddmm.mmmm
<4>	Latitude hemisphere, 'N'/'S'
<5>	Longitude, dddmm.mmmm
<6>	Longitude hemisphere, 'E'/'W'
<9>	Date UTC, ddmmyy

TABLE IV. GPS SATELLITES IN VIEW (GSV)

\$GPGSV,<1>,<2>,<3>,<4>,<5>,<6>,<7>,...<4>,<5>,<6>,<7>,*HH<CR><LF>	
<1>	The total number of GSV language
<2>	The number of current GSV
<3>	The total number of satellites in view
<4>	PRN code (01~32)
<5>	Satellite elevation (0~90 degree)
<6>	Satellite direction degree (0~359 degree)
<7>	Signal-to-Noise (00~99dB)

TABLE V. TRACK MADE GOOD AND GROUND SPEED (VTG)

\$GPVTG,<1>T,<2>M,<3>N,<4>K,<5>*HH<CR><LF>	
<1>	Ground navigate direction based on true north (0~359)
<4>	Ground speed (0000.0~1851.8 km/h)

TABLE VI. GLOBAL POSITION SYSTEM FIX DATA (GGA)

\$GPGGA,<1>,<2>,<3>,<4>,<5>,<6>,<7>,<8>,<9>M,<10>M,<11>,<12>*HH<CR><LF>	
<7>	The number of current using satellites (00~12)
<9>	Height above sea level (-9999.9~99999.9 m)

TABLE VII. GPS DOP AND ACTICVE SATELLITES (GSA)

\$GPGSA,<1>,<2>,<3>,<3>,<3>,<3>,<3>,<3>,<3>,<3>,<3>,<3>,<3>,<3>,<3>,<3>,<3>*HH<CR><LF>	
<3>	PRN code, (01~32)

In this module, we define two functions: void GPSWindow:: ProcessGPSFrame (QString CurFrame) and QSatInfoWindow:: (QString CurFrame) to incept the signal from GPSWindow::NewGPSLine(QString szText). The first one answers for distilling the information such as orientation and speed, infill to QGPSData Struct, and broadcast the information through the function void GPSWindow:: NewGPSData(QGPSData Data). The other function just distill the information related to the

satellite, protracts roof satellite sketch map in widget QSatInfoWindow.

III. THE REALIZATION OF MOBILE 3D TOPOGRAPHY

The most important module of the topography system is the mobile 3D topography engine. As there is no 3D accelerate hardware on the embedded platform, all the 3D graphics must be finished by software, CPU takes on all the calculation for 3D transformation, this requests the calculating spending of the graph engine as small as possible.

Considering the particularity of the topography, this chapter will expatiate how to realize the engine for 3D topography transform and display. The engine abandons the universal request of 3D modeling, merely aims at this especial model, and uses many predigested and specialized expert algorithms to reduce the whole calculation quantity.

A. Sand Tray Model

In the sand tray model of this paper, we consider topography as a continuous surface $z=f(x, y)$ of single value, other than a 3D object. Using surface to replace the object can reduce the complexity of the program. Many graphics warehouses including Open GL all use surface to describe 3D object other than entity. To the typical band surface structure like topography, we use lithosphere surface other than the whole earth to show it.

In the process of sand tray model protract, we should gain the world coordinate location of each sample dot firstly, that is (x_0, y_0, z_0) , and convert this world coordinate to display coordinate (DispX, DispY). According to the location relationship between the dots, we gain the figure expression of the surface.

1) The conversion from three-dimensional points to two-dimensional display coordinates at fixed sight line

In the process of sand tray model display, the first step is to convert the dots stored by world coordinate to the display coordinate on two-dimensional screen plane. Fig.4 shows the calculation process.

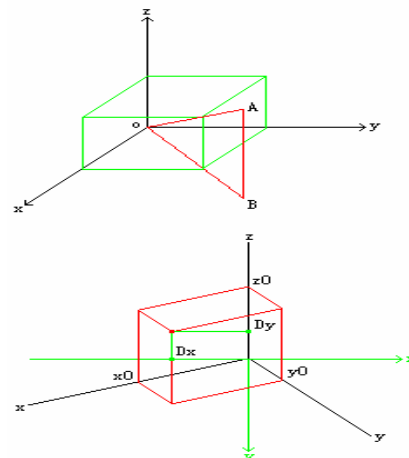


Figure 4. Grid Scan of the topography

In the upper figure, the red radial AO show the sight light direction, xyz is the world coordinate, axes X shows the true north, surface xoy is the horizontal level (base

level). BO is the vertical projection of AO. Suppose angle B-O-A is γ ($-90^\circ < \gamma < 90^\circ$), angle X-O-B is α ($0 < \alpha < 360^\circ$), then (α, γ) can confirm a sight line direction uniquely. The nether figure shows the actual graph from the sight line direction. The display plane is the plane vertical to line AO and cross the world coordinate origin. Project the world coordinate and all the dots in it, we can get the graph in the nether figure.

In the upper figure, the red dot represent a dot in the world coordinate system, its world coordinate is (x_0, y_0, z_0) , the green coordinate system represents display coordinate system, suppose its coordinate in display coordinate system is (D_x, D_y) , then according to the space analytic geometry we can get:

$$D_x = y_0 * \cos \alpha - x_0 * \sin \alpha \quad (1)$$

$$D_y = x_0 * \cos \alpha * \sin \gamma + y_0 * \sin \alpha * \sin \gamma - z_0 * \cos \gamma \quad (2)$$

We can convert dots in the world coordinate system (x, y, z) to the display coordinate (D_x, D_y) that can be displayed on the screen. If the model is circumrotated, we just need to adjust the sight line direction, that is alter the value of α and γ , and recalculate the projected coordinate according to the below function.

2) The estimation of the dots location relationship

The estimation of sheltering relationship is an important problem in the 3D graph. There exist a lot of mature algorithms for this aspect, considering the particularity of the current problem we adopt comparatively easy thought in this module: we protract the chart farthest to the observing dot first, and protract later to the near, in this way the nearer chart will keep out the farther part. According to the discussion on the sand tray model, the intersecting problem between surface and lines on the continuous single value surface doesn't exist. There exists the problem of wasting calculating time on the protracting for some completely sheltering surfaces in this algorithm. While using other algorithms, we should estimate the visibility of the surface in the first place, which also consumes a mass of calculating time. Thus we can see the advantage of our algorithm.

The calculation of the distance between each dot and the observing dot is the key in this algorithm. The display projection surface is a plane in the world coordinate system, this indicates a supposition: the observing dot is very far from the world coordinate system. Thus the measurement of the distance between each dot and the observing dot is meaningless. So the dots location relationship can be defined by the distance between the dots and the display plane (the plane cross the coordinate origin O and vertical to line OA). Suppose L is the distance between dot (x_0, y_0, z_0) and the display plane. Then we get:

$$L = x_0 * \cos \alpha * \cos \gamma + y_0 * \sin \alpha * \cos \gamma + z_0 * \sin \gamma \quad (3)$$

L indicates the location relationship of the dots, and is the estimation basic of sheltering relationship to lines and surfaces.

3) The estimation of lines' sheltering relationship

We can protract the topography through connecting the dots. For example, we connect (x_0, y_0, z_0) , (x_{0-1}, y_0, z_1) , (x_{0+1}, y_0, z_1) and (x_0, y_{0+1}, z_1) to get a 3D object composed by the line circle. So we can get the projection of 3D object by connecting the two dimension projection of the above dots in the display plane. As the line has no width, the sheltering relationship between lines doesn't exist. But in the practical protracting, the width of line is at least one pixel, thus we should consider the sheltering problem between lines especially after coloring up. We use the L value to mark the location of the string. This sheltering estimation can avoid the sheltering disorder on vision through practice proving.

4) The estimation of surfaces' sheltering relationship

We can easily extend the line's estimation method to the surface protracting method. We use a polygon that contains a filling brush to replace the above 4 margin between the 4 dots. We can use the L value of the surface's 3 acmes to estimate the surfaces' location relationship. Considering the asymmetry, the local sheltering disorder may exist, so we should revise the estimation algorithm combining this special protracting method.

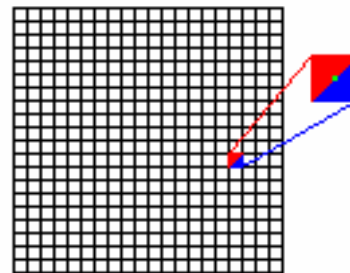


Figure 5. Grid Scan of the parcel topography

The grid in Fig.5 expresses the plane XOY (the basic level) after dispersing. According to the particularity of the surface filling algorithm, we should integrate a pair from the two planes of the four adjacent acmes, and then use the diagonal intersection point's L value of the quadrangle's projection to mark the sheltering order of this pair of the plane. After the arrangement of the sheltering from all the pairs composed by two planes, we confirm the order of the two planes according to the location of the two planes of each pair, thus we finish all the location estimation.

B. Surface Filling Algorithm

To protract the image element of the triangle plane, the most direct thought is to confirm the projection location on the display plane of the three acmes, and then use function QPainter::drawPolygon to protract it. But there exist some problem in the algorithm. Firstly, due to the setup of QT/Embedded, the working status of QPainter::drawPolygon on the embedded equipment is similar to QPainter::drawPolygonLine, not filling the polygon. This problem can be settled by clipping the rectangle protracting by QRegion, but the enwind algorithm or odd-even checkup algorithm has more calculation quantity when creating polygon field, which misfits the condition of embedded equipment.

In order to solve the surface filling problem, we build a memory field according to Microsoft Bitmap File (BMP) in this module, and setup the corresponding location of each triangle image element in the memory while protracting, then use function QPixmap:: loadFromData to write the memory block to the display BMP. Because the protracting polygon is simple during the memory setup, we don't use the traditional algorithm that needs more calculation quantity. we estimate the inner dots of the triangle to fill in according to the geometry principle instead.

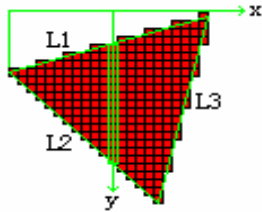


Figure 6. The estimation and filling of triangle's inner dots

We use scan line algorithm while filling the triangle. Calculate the line function of L₁, L₂ and L₃, scan the object through the increasing of coordinate x from left to right row after row, and then calculate the coordinates of the intersection points between the scan line and L₁, L₂ or L₃ to estimate whether the dots belong to the triangle field. Repeat this process to setup all the inner points in the pixel matrix, we finish the protracting of the triangle image element.

C. Coloring up Algorithm

The engine provides three coloring up modes: perspective mode, altitude mode and custom mode. The effects of these three modes differ from each other, but realizing principle is similar. The perspective coloring up algorithm decides the color of the dots by the location relationship. The farthest dot is blue, the nearest is red, the dot between the two dots is the linear interpretation of blue and red. The altitude algorithm is similar to the perspective algorithm, it regards the altitude of dot as the coloring up basis, and is also flat coloring up. The custom coloring up uses the coloring up policy decided by the user. When the user needs to display the map data, he needs to appoint the color of each dot. And if the user wants to abandon the custom coloring up scheme, he could setup the dot color as black (RGB=0), at this time the mode transfers to altitude mode.

D. Illumination Model

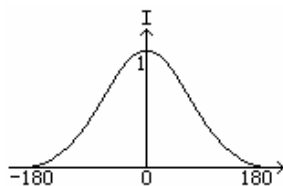


Figure 7. The Space distribution of the reflecting light intensity

Fig.7 shows the space distribution of the intensity of mirror reflecting light due to air's dispersion. The

abscissa expresses the angle between view sight and reflecting light, the vertical coordinate expresses the relative light intensity. We can use function $I_0 = (1 + \cos \beta) / 2$ to denote the curve, and we use constant I_1 to denote the diffuse reflection light. Suppose the intensity of the incident light is I , then $I = I_0 + I_1$, β is the angle between view sight and reflecting light, then the following formula shows the observing intensity:

$$L_c = I * [(1 + \cos \beta) * \tilde{p} + \tilde{p}] \tag{4}$$

Suppose the incident light direction is same to the view sight, then β is twice of the angle between surface normal and view sight. Suppose the three acmes of one triangle image element are (x, y, z_0) , $(x+1, y, z_1)$ and $(x, y+1, z_2)$, then the normal vector can be expressed by the following formula:

$$A = -(z_1 - z_0) * i - (z_2 - z_0) * j + k \tag{5}$$

Equation (6) shows the view sight vector:

$$B = \cos \gamma * \cos \alpha * i + \cos \gamma * \sin \alpha * j + \sin \gamma * k \tag{6}$$

According to (4), (5), (6), we can get (7):

$$\cos \beta = 2 * [A \cdot B / (|A| * |B|)]^2 - 1 = 2 * \{[\sin \gamma - \cos \gamma * \sin \alpha * (z_2 - z_0) - \cos \gamma * \cos \alpha * (z_1 - z_0)] / \sqrt{[1 + (z_1 - z_0)^2 + (z_2 - z_0)^2]}\}^2 - 1 \tag{7}$$

According these equations, we can calculate the intensity of each surface of the 3D object when the view sight of a certain angle.

E. Working Flow Figure

The 3D figure engine of the whole system is completed hereto. Fig.8 shows the working flow:

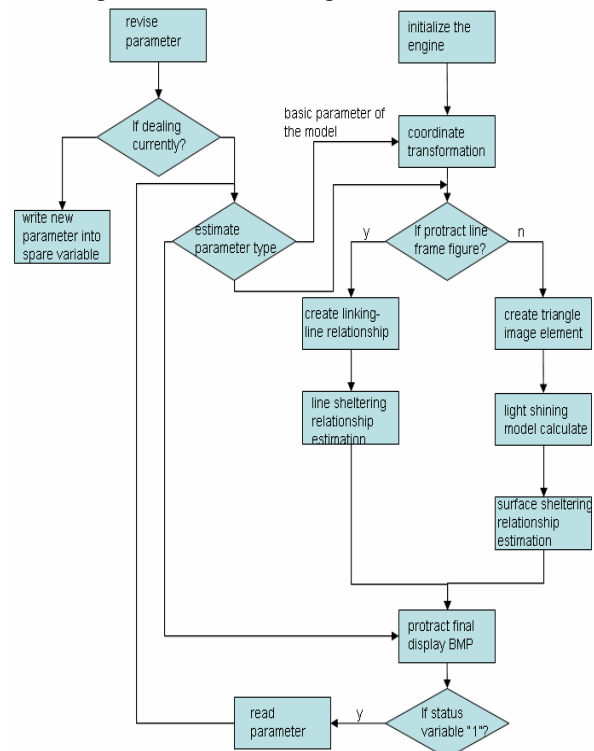


Figure 8. The Working Flow Chart

IV. SYSEYEM TESTING

We design a serious of testing scheme to test the system on function and capability aimed at the below designing and realization.

A. Testing Conditions

The hardware platform of this system is Intel-Sitsang Evaluation Board. Intel-Sitsang is based on Intel XScale processor and is an embedded hardware developing platform. The hardware parameter of this equipment is: microprocessor with XScale truss and main frequency is 400MHz, 64MB RAM, E2PROM and 32MB FLASH MEM. The equipment also has 10M Ethernet network card, a Bluetooth-UART, a Full Function-UART, two USB Host Ports and a touching screen.

The testing software is embedded LINUX based on ARM-LINUX kernel and QTopia windows system, the kernel is 2.4.18.

The server is realized by PC, its hardware parameters are: 1700MHz Celeron CPU, 256M DDR, 40G hard disk, 10M/100M Ethernet network card, RS-232. The operation system is Red Hat 9.0 LINUX.

B. Module Testing Scheme and the Result

The 3D topography system's browser is composed by 4 modules: GPS, WLAN, 3D image engine and user GUI.

1) Testing scheme I

We run the simulation server which can bring stochastic 3D topography data on PC, to test the stability of the network transmission on the status of large data amount and long runtime circulating. The testing result shows, in 61minutes and 15 seconds, the system finishes 735 times of data update, the total data incept amount is 7.4097MB, the total data send amount is 18.6621KB, average transmitting speed is 124.5KB/s, largest transmitting time is 150ms, smallest is 50ms. The result shows the strong stability of the network transmission, and doesn't appear data accumulating status on long time running.

2) Testing scheme II

We run the circulating time test of 3D display, rotation and zooming for different resolution ratio and data amount through the setup of configuration files' parameter to adjust resolution ratio and transmission amount of the 3D topography, thus to test the display, rotation, zooming efficiency. Repeat the test 20 times under same condition for different data, the result shows: when model resolution ratio is 31×31 , the average time of 3D display is 1.9s; when model resolution ratio is 21×21 , the average time of 3D display is 0.86s; when model resolution ratio is 51×51 , the average time of 3D display is 4.9s. We get the conclusion that: the 3D display time is proportional to the square of the resolution ratio. The higher the resolution ratio is, the more diaphanous the 3D topography is, and the longer the running time is. So we choose 31×31 as the better scheme.

3) Testing scheme III

We turn off the network transmission and cease the 3D display commutation operation, to test the GPS module single. We choose the open field in the suburb, test time

is 22:00 to 22:30, and the weather condition is good. The test shows: In 30 seconds (average value) after the program starts we can get the accurate orientation, the satellite in view is 9. The result shows: The system can gain the orientation information accurately in most outdoor environment.

C. The Overall Testing Scheme and the Results

Overall testing scheme: Run the data server program on PC, and run the client server program on embedded equipment, setup the update time as 55 seconds through configuration files, the resolution ratio is 31×31 , use WLAN to transmit data outdoors, do the client test to the system. The test time is 15 minutes, the function status is good, meet the designing request entirely.

1) Code Testing and Tools Detecton

- Pass Numega's DevPartner test.
- Pass the Logiscope software static test by Telelogic Corporation.
- Pass the TeamTest by Rational Corporation and WebStress Software black box test by Microsoft.
- Use TestBytes to optimize the application performance.
- Use DevPartner Tools to test the covering rate which is over 80%.

2) The System Operation Capability Index

- The database supports 5000 supervision, the inquiring time is no more than 50ms each time.
- The data service software supports 1000 supervision connect, for every data request, the responding time is no more than 50ms.
- Under the resolution ratio 31×31 , the time of finishing an entire 3D transform is no more than 2.1 seconds.
- The backup time of the program interface is no more than 0.25 second in all instances.
- The orientation precision of GPS: 5 to 25m on horizontal direction, no more than 50m at elevation.
- The maximum network transmission delay is no more than 150ms.

3) Test Conclusion and Analysis

Through the single module test and the overall test of the system, we see the good performance of each module, the total operating efficiency and the effects all meet the design requests. 3D display module restricts from Intel-Sitsang Evaluation Board, we can just choose a suitable balance point for the operating speed and display effect, and it can be accepted as a whole.

V. CONCLUSION AND EXPECTATION

We developed a Real-Time Mobile 3D Topography System for embedded system. We tackle the key problem of topography data abstract, topography database design,

Libra middleware realization, GPS driving module and 3D figure engine, and all gain the good effect.

The 3D topography data engine's function is very strong, it supports 3D coordinate transform, rotation, illumination commutation, zooming, field depth effects, and so on. The display effect is good, as well as the efficiency and display quality. We can also use it in military affairs, scientific review field, architecture, urgent service, archaeological site protracting, and entertainment industry. Through certain technology fruit conversion and corporation hatch, it will play its corresponding role.

ACKNOWLEDGMENT

This research was supported by the National High-tech R & D Program (863 Program) of China (No.2008AA04Z127) and Shanghai Leading Academic Discipline Project (No.B210).

REFERENCES

- [1] Turner K., "What is the difference among 2D, 2.5D, 3D and 4D," *GIS World*, vol.10(3), pp.54, 1997.
- [2] Molenaar M., "A formal data structure for three-dimensional vector maps," in *Proceedings of the Fourth International Symposium on Spatial Data Handling*, 1990.
- [3] Molenaar M., "A topology for 3-D vector maps," *ITC Journal*, vol.(1), pp.25-33, 1992.
- [4] Victor J. D., "Delaunay triangulations in TIN creation: an overview and a linear-time algorithm," *Int. J. Geographical Information Systems*, vol.7(6), pp.501-524, 1993.
- [5] Pilout M., Tempfli K., and Molenaar M., "A tetrahedron-based 3D vector data model for geoinformation," in *Advanced Geographic Data Modelling*, vol.(40), Netherlands Geodetic Committee, Publications on Geodesy, 1994, pp.129-140.
- [6] Chen Xiaoyong and Ikeda Kozo, "Three-dimensional modeling of GIS based on delaunay tessellations," *International Archives of Photogrammetry and Remote Sensing*, pp.132-139, 1992.
- [7] Bak P. and Mill A., "Three dimensional representation in a geoscientific resource management system for the minerals industry," in *Three Dimensional Applications in Geographic Information Systems*. Taylor & Francis, 1989, pp.155-182.
- [8] Li R., "Data structures and applications issues in 3D geographic information systems," *Geomatica*, vol.18(6), pp.209-224, 1994.
- [9] Egenhofer M., "A model for detailed binary topological relationships," *Geomatica*, vol.47(3), pp.261-273, 1993.
- [10] Pigot Simon, "A topological model for a 3D spatial information system," *Auto Carto*, vol.(10), pp.268-392, 1992.
- [11] Randell D., Cui Z. and Cohn A., "A spatial logic based on regions and connection," in *Proc. 3rd Int. Confon Knowledge Representation and Reasoning*, Boston, 1992, pp.165-176.
- [12] Cui Z, Cohn A. G., and Randell D., "Qualitative and topological relationships in spatial databases in design and implementation of large spatial databases," in *Third International Symposium, SSD'93, LNCS692*, Abel D., Qoi B. C., Eds. Pisa: Springer-Verlag, 1993, pp.396-415.
- [13] Chen Jun, Li Chengming, Zhilin Li, and Gold C. M., "Improving 9-intersection model by replacing the complement with voronoi region," in *Proceedings of Inter. Workshop on Dynamic of Multidimensional GISs*, HongKong, 1997.
- [14] Openshaw S., "What is GIS able spatial analysis," in *EUROSTAD3D: New Tools for Spatial Analysis*, Luxembourg, 1994, pp.36-44.
- [15] K.Jiang, L.D.Seneviratne, and S.W.E Earkes, "Finding the 3D shortest path with visibility graph and minimum potential energy," *Proceedings of the 1993 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Yokohama, Japan July 26-30, 1993.
- [16] Sun M, Chen J, and Zhou Q.M., "Cloverleaf junction expression in three-dimensional city model," *The International Archives of Photogrammetry and Remote Sensing*, vol.32(12), pp.109-114, 1999.
- [17] Zhang H. Y., Li J. W., and Zeng X. G., "The study of digital terrain model in the 3D visualized system of computer aided railroad alignment and design," *The International Archives of Photogrammetry and Remote Sensing*, vol.32(12), pp.319-324, 1999.
- [18] Du P.J., Guo D.Z., Sheng Y.H., "Data Structures and Visualization in 3D-GIS Taking into Account the Properties and Applications in Mines," *The International Archives of Photogrammetry and Remote Sensing*, vol.32(12), pp.281-284, 1999.

Weihui Dai received his BS degree in Automation Engineering in 1987, his MS degree in Automobile Electronics in 1992, and his PhD in Biomedical Engineering in 1996, all from Zhejiang University, China. He is currently an Associate Professor at the Department of Information Management and Information Systems, School of Management, Fudan University, China. He has published more than 100 journal papers and conference articles in Software Engineering, Information Systems, Service Management, Net-ecology, e-Business, etc. Dr. Dai became a member of IEEE in 2003, a senior member of China Computer Society in 2004, and a senior member of China Society of Technology Economics in 2004.

Shuyi Liang received her BS degree in Applied Mathematics in 2006 from Fudan University, China. She is current a master student at School of Management, Fudan University, China.

Wei Yan received his MS degree in Software Engineering in 2007 from Fudan University, China.