TEXTURE BASED SATELLITE VISIBILITY DETECTION

(DOKU TABANLI UYDU GÖRÜNÜRLÜĞÜ ALGILAMA)

by

Can GÖÇMENOĞLU, B.S.

Thesis

Submitted in Partial Fulfillment

of the Requirements

for the Degree of

MASTER OF SCIENCE

in

COMPUTER ENGINEERING

in the

INSTITUTE OF SCIENCE AND ENGINEERING

of

GALATASARAY UNIVERSITY

June 2014

TEXTURE BASED SATELLITE VISIBILITY DETECTION

(DOKU TABANLI UYDU GÖRÜNÜRLÜĞÜ ALGILAMA)

by

Can GÖÇMENOĞLU, B.S.

Thesis

Submitted in Partial Fulfillment

of the Requirements

for the Degree of

MASTER OF SCIENCE

Date of Submission : May 21, 2014

Date of Defense Examination : June 19, 2014

Supervisor : Assoc. Prof. Dr. Tankut ACARMAN

Committee Members : Prof. Dr. Bernard LEVRAT

Asst. Prof. Dr. Murat AKIN

Acknowledgements

First and foremost, I wish to thank my supervisor Assoc. Prof. Dr. Tankut Acarman for his extensive guidance through every step of the process. While he influenced me with his helpful comments and directions, he also encouraged me to try my own ideas and explore new areas. For this I am sincerely grateful.

I would like to thank my parents for teaching me the importance of science since I was a child and for all their continuing support.

I am also grateful to Infotech for providing invaluable digital map data for our experiments.

Finally, I would like to thank my friends and colleagues, for all their help.

Table of Contents

List of Symbols

3D	Three Dimensional
BSP	Binary Space Partitioning
CPU	Central Processing Unit
DOP	Dilution of precision
DOP	Dilution of precision
GLSL	OpenGL Shading Language
GNSS	Global Navigation Satellite System
GPU	Graphics Processing Unit
ITS	Intelligent Transportation Systems
IV	Intelligent Vehicle
LOS	Line-Of-Sight
NLOS	Non-line-of-sight
OBU	On-board Unit
PVRTC	PowerVR Texture Compression
ROI	Region of Interest
SNR	Signal-to-Noise Ratio
TBSVD	Texture Based Satellite Visibility Detection
VANET	Vehicular Ad-hoc Network

List of Figures

List of Tables

Abstract

Limited satellite visibility, multipath and non-line-of-sight (NLOS) signals greatly reduce GNSS positioning accuracy in urban environments. Results of various research have shown that 3D representations of these environments can be helpful for determining NLOS signals and multipath, therefore they can be used to improve positioning accuracy which is invaluable to ITS and VANET applications. State-of-the-art methods use computationally expensive raytracing algorithms on 3D polygon-based maps to determine satellite visibility.

In this thesis, we introduce Texture Based Satellite Visibility Detection (TBSVD) method which uses texture-based algorithms to calculate visibility information using height maps of urban structures, contributing to the application of these methods in low-cost receivers and on-board units.

Real road tests and measurement campaigns in the business district of the metropolitan city shows that our method, which significantly reduces computational costs and storage requirements compared to raytracing, can be effectively used to improve GNSS positioning accuracy.

Résumé

La précision de positionnement par Système de Positionnement Par Satellite, GNSS (Global Navigation Satellite System) dans les environs urbains réduit considérablement à cause de visiblité limitée de satellite, multipath (multi-trajet) et sans ligne de vision signaux, NLOS (non-line-of-sight). Les résultats des divers recherches ont démontré que les représentations 3D de ces environs peuvent être utilisées pour améliorer la précision de positionnement qui porte assez d'importance pour les Systèmes de Transport Intellligent, ITS (Intelligent Transportation Systems) et pour les Réseaux Ad-Hoc de Véhicules, VANET (Vehicular Ad-Hoc Network). Les méthodes présentes se serve du technique le lancer de rayon sur les cartes 3D polygone-basé pour déterminer la visibilité de satellite.

Pour contribuer l'utilisation de ces méthodes à faible coût et aux unités à bord on introduit la méthode que l'on appelle La Texute-Basée Détection de Visibilité Satellite - Texture Based Satellite Visibility Detection (TBSVD). Cette méthode traite les algorithmes basés image pour calculer l'information de visibilité utilisant les cartes d'altitues des structures urbain.

Les tests réalisés dans le quartier des affaires aux vraies routes métropoliennes ont confirmés l'éfficacité de la méthode proposée, TBSVD qui réduit les coûts computationels et spatiales comparée au lancer de rayon et peut être utilisée effectivement pour bonifier la précision de positionnement GNSS.

Özet

Kısıtlı uydu görünürlüğü ve uydu sinyallerinin çokyollu yayılması, GNSS (küresel uydu yön bulma sistemleri) pozisyonlama hata oranlarını etkileyen en önemli faktörlerdendir. Pozisyonlama hatalarının azaltılması, özellikle akıllı ulaşım sistemleri (ITS) ve araçlar arası ağlar için oldukça önemlidir. Çeşitli araştırmalarda, şehirlerin 3 boyutlu gösterimlerinin, görüş alanında bulunmayan bu tip sinyallerin tespit edilmesinde, dolayısıyla pozisyonlama hatalarının azaltılmasında yardımcı olabileceği belirtilmiştir. Ancak, araştırmalarda sözü edilen 3 boyutlu modelleme tabanlı algoritmalar, hesaplama karmaşıklı açısından oldukça verimsiz olan ışın izleme yöntemini kullanmaktadır.

Bu tez kapsımda, yöntemlerin düşük maliyetli araç içi üniteler ve diğer yön bulma cihazlarında kullanılabilmesine katkı sağlamak amacıyla, Doku Tabanlı Uydu Görünürlüğü Algılama yöntemini sunuyoruz. Bu yöntem bilgisayar grafikleri alanında da kullanılan doku ve görsel tabanlı algoritmaları şehirlere ait yükseklik haritalarına uygulayarak, verilen bir pozisyon için uydu görünürlüklerini hesaplamaktadır.

Şehiriçinde gerçekleştirdiğimiz gerçek yol testleri ve deneylerle elde ettiğimiz ölçümler, bu yeni yöntemin, eski ışın izleme yöntemlerine göre hesaplama karmaşıklığını azalttığını ve GNSS pozisyonlama performansını iyileştirdiğini göstermektedir.

1 Introduction

Intelligent Transportation Systems and Intelligent Vehicle Technologies introduce safety-critical and liability-critical applications to enhance mainly road traffic safety, road capacity and road tolling by all means of connected vehicles. Position measurement with respect to some absolute coordinate system is usually done using GNSS receiver, and possibly matching with a digital road map data. Position information, its accuracy and its reliability becomes crucial towards fusion of sensor data and decision making.

Major causes of the local degradations of the GNSS signals in urban environments are mainly due to reduced satellite visibility, multipath and non-line-of-sight signals. Multipath mitigation with special hardware is technologically feasible but expensive. For instance, a high level setup with choke ring antenna costs 15.000 Euro to mitigate multipath effects whereas the cost of Ublox development kit with a single frequency antenna receiver is approximately 300 Euro Bauer et al. (2012). Since urban GNSS receivers with lane-level accuracy is desired for any IV and ITS applications, receiver cost becomes important for multipath mitigation methods. Thus, many commercially available GNSS software use signal-based metrics such as Signal-to-Noise ratio to adjust weights of the satellites in the mutual positioning algorithm.

These degradations can also be mitigated in software by giving receivers information about their surroundings. 3D map-based methods give equivalent or more accurate results, especially when combined with 3D map matching, position accuracy can be significantly improvedPeyraud et al. (2013). Using 3D data, it is possible to determine satellite visibility and even calculate reflected signals for any given location. Theoretically, using an exact representation of GNSS receiver's local environment, multipath delays can also be simulated and mitigatedBauer et al. (2012).

On the other hand, computational complexity and storage requirements for 3D map based methods are considerable. Several performance aspects for leveraging 3D information on low-cost receivers exist: accuracy and resolution of 3D data, its storage and distribution, and computational complexity of the algorithm can be stated as the preliminary performance issues. Processing 3D data in order to determine Line-Of-Sight condition of the satellite with respect to the GNSS receiver requires storing buildings and other structures in the form of polygonal models, and applying raytracing, which means using triangle-ray intersection

algorithms to calculate signal occlusions. If a space-partitioning algorithm is not used, each ray will have to be tested against all triangles around the local map. Use of a space partitioning algorithm, such as binary space partitioning Torres (1990), reduces time complexity but increases space complexity and makes generation, storage and dissemination of 3D maps increasingly complex.

To overcome the performance issues of polygonal raytracing methods, researchers have looked into image and texture based methods. In one study, shadow mapping for Non Line-Of-Sight (NLOS) detection is used to exclude NLOS satellites from the multilateration solution, and better positioning accuracy has been achieved Bauer et al. (2013). It is an improvement over polygonal raytracing methods, but it is still infeasible for mass market devices, as the method requires frequent updates and transmission of shadow maps for each individual satellite, which may not be an efficient and scalable solution for land vehicles in urban area. Shadow matching is another distinct method which uses 3D maps Groves (2011). This method tries to match the number of NLOS satellites calculated from the 3D map to the number of NLOS satellites calculated at the receiver. Areas where number of NLOS satellites match the receiver values constitute a set of positions where the real position might probably be.

As a solution to these existing problems with both polygonal and image-based methods, we propose a novel method called Texture Based Satellite Visibility Detection (TBSVD). It is an application of a Computer Graphics algorithm, called relief mapping or steep parallax mappingMcGuire & McGuire (2005); Risser et al. (2005), to the case of satellite visibility detection.

This CG algorithm uses image-based ray marching to find the collision point of eye-vector directly on 2D height maps, and it is an improved version of the previous methods on bump mapping and normal mapping. TBSVD follows the same approach: 3D height data are stored on 2D maps and image-based ray marching is applied to determine visibility status of satellites. Storing urban 3D information on height maps is a well known method, but novelty lies along this algorithm for calculating the raytracing directly on 2D data. Height maps are 2D gray scale images where height values for each position are stored at each pixel. Each pixel of the image corresponds to a fixed size area on the real world. Firstly, the ray marching method is applied on the image to determine satellite signal collision with the structures in the vicinity. Then, satellites are classified into line-of-sight, obscured and multipath categories. Using satellite geometry and collision information with respect to the pixel of the image in the neighborhood of the GNSS receiver, it is possible to calculate various GNSS measurement metrics such as dilution-of-precision (DOP), vertical dilution-of-precision (VDOP), time dilution-of-precision (TDOP) and horizontal dilution-of-precision (HDOP). Also, with

receiver-specific information, integrity metrics such as protection levels can be estimated. If a GPU is used, all of these calculations can be performed on a real-time basis in parallel for each pixel (each tile in the grid) providing not only the visibility or integrity information on a single point but on a larger area whose size is determined by the texture size supported by the GPU (usually more than 2048x2048 texels).

TBSVD is suitable for its implementation on a CPU with low processing capability, but it is better suited for implementation on a Graphical Processing Unit . In recent years, GPU, which has been already an invariable part of a personal computer, becomes as much common as CPUs on mobile devices such as smart-phones and tablets. For instance Imagination, producer of PowerVR low-cost low-power GPUs which are used in tablets and smart-phones, sells GPU chips to deploy automotive applications e.g., 3D navigation and infotainment units (see for instance (Imagination, 2014, http://www.imgtec.com/markets/automotive.asp)). We believe that the demand for graphically advanced displays and interfaces in vehicle systems, will make GPUs an integral part of vehicle OBUs and their computational power can be harnessed for vehicular technologies.

Use of TBSVD for GNSS positioning brings some additional advantages for IVs and ITS applications. Firstly, height map can be installed on or disseminated to the OBUs of vehicles at the upper layer of local dynamic map on the navigation system. Secondly, height map can be disseminated at varying resolutions. A low resolution height map will give less accurate collision information, but processing and distribution will be faster. Thirdly, our approach enables deployment and calculation of integrity on a real-time basis on the OBU. Additionally, developers can use 3D volumetric data of nearby area for other aiding purposes, such as displaying the driver a 3D view of the area or determining LOS path between vehicular nodes to route data packets in VANET. Finally, height maps can be easily generated, as building shapes and their heights (or number of floors they have) are already stored in city information databases.

In this study, we first present our methodology and algorithm in detail, discussing complexity, requirements and related options. We also compare it to the existing methods and algorithms. Then we validate our method with observation data gathered from real-world experiments conducted in various urban locations in İstanbul, Turkey.

2 Methodology

In an urban area, tall structures, i.e., buildings, are the main obstructions for satellite signals to reach to the receiver's antenna. These 3D structures can be easily projected on 2D maps with their 2D shape from top view and their depth information, which is far more important than their fine details from the view point of this study (as seen in Figure 2.1).

Figure 2.1: Height map for Kadıköy, Istanbul. This is generated using vector shape and height information. This image is inverted for visual clarification, thus color black (0.0) represent highest points, while white (1.0) represents lowest point.

Texture-based ray marching methods are mainly used in computer graphics for real-time calculation of various special effects such as relief mapping and ambient occlusion. In the presented approach, ray marching algorithm is applied on the height map to check whether the GNSS receiver is line-of-sight with respect to the satellite or not, and the need for polygonal storage and processing of 3D representations of road-side structures is eliminated. To the best of our knowledge, generating realistic representations and dissemination of structural information to ITS stations have not been addressed in open literature.

2.1 Height Map Format

Height map is a gray scale image where each pixel corresponds to a small tile on the real world. Each pixel has a value ranging from 0.0 to 1.0 where 0.0 is the ground level and 1.0 is the given maximum height. For the proposed implementation, the minimum surface area represents the image resolution, and the number of floors or height steps represents the image depth. Throughout this study, a resolution of 1 square meter area is used for each pixel and 8-bits per pixel depth represents 256 levels of height, which is sufficiently rich for digital meaning of current urban structures. Also, image compression algorithms may store 2 bits per pixel (bpp) images, which gives depth information with good accuracy, for instance interested readers may further investigate PVRTC Ogniewski et al. (2011). Data size of height map stored as image while using compressed textures can be drastically reduced,

for example a 42 square kilometers area can be stored in 10 megabytes. Textures are also suitable for streaming over the data network, a 100 meters square tile with a high resolution of 1 meters per pixel has the data size of 25 bytes.

2.2 Ray Marching on Height Maps

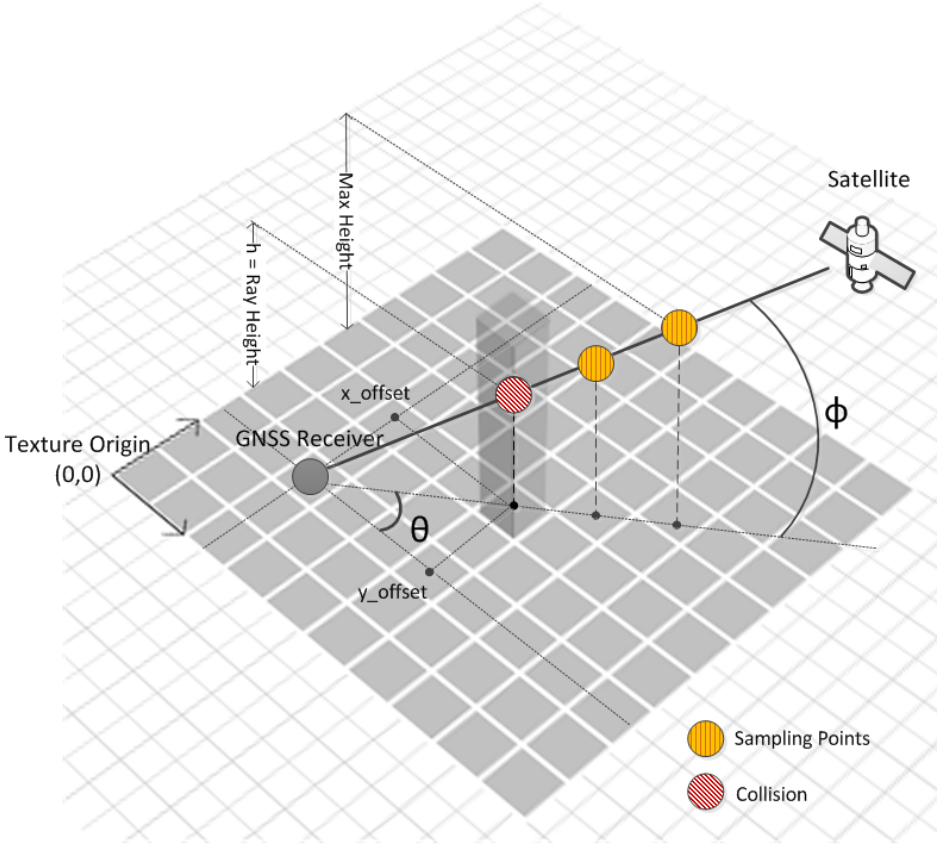

Figure 2.2: Illustration of ray marching algorithm to convert 3D model of the city into elevation map. Along the ray starting at the satellite and ending at the GNSS receiver, it is marched subject to a fixed number of sampling texture look-ups to determine possible collision with an existing 3D structure on the digital map.

For a given distant object in the sky level, represented by its azimuth and elevation angle, and a given point on the map, sampling is performed by descending from the maximum height to the ground level along a straight line at a constant sampling step. If a collision is detected at any sampling point, the point is marked as NLOS with respect the the individual satellite, i.e., occluded. By repeating this checking procedure for each pixel, a binary bitmap representing the occluded areas from the view point of the satellite is generated as illustrated in Fig. 2.2. A similar approach can be used to determine signals reflected from the ground to calculated reflected multipath. If a ray reaches the ground level, the ray marching is continued in the

same direction. However, instead of lowering ray height, we increase it. Reflected ray can be checked for collisions the same way we check for the incident ray.

Translation of map coordinates into texture coordinates and calculation of texture look-up offsets based on satellite position constitutes two main stages of the algorithm. "Offsets" denote the pair of $x_{offset}$and y_{offset} derived for a given individual satellite at a sampling point. These offsets are used for texture look-ups from height maps. When sampling 2D textures, texture coordinates are not necessarily mapped into individual pixels, hence the word "Texel" is used for a sampled part of a textureHeckbert (1989).

The value of maximum height and pixel width in meters is the parameter of the generated height map. Once again, maximum height represents the height at 1.0 value for a given pixel or texel, and pixel width denotes the width of the square area on the ground. Based on these values, the height value is mapped into texel size,

$$h_{pixel} = \frac{h \times maxheight}{pixelwidth} \tag{2.1}$$

where h denotes the ray height for the given texel. Relative distance between the given position of the GNSS receiver and the sampling point on the height map is calculated at a given height on the ray based on the elevation, which is denoted byθ:

$$distance_{pixel} = \frac{h_{pixel}}{\tan\theta} \tag{2.2}$$

Then, the sampled point texel offsets are calculated by using the azimuth angle ϕ : $x_{offset} = distance_{pixel} \times \sin\phi$ and $y_{offset} = -distance_{pixel} \times \cos\phi$, (the y offset is negated due to inversion of y axis in digital images).

The flowchart of algorithm is given in Fig. 2.3, and it is described as follows:

1. Latitude and longitude coordinates of the GNSS receiver are measured, height map is stored as a gray scale bitmap image, elevation and azimuth angle with respect to the satellite's position, whose signal line-of-sight condition will be checked, is taken as input,

2. Latitude and longitude coordinates are converted to texture values ranging between 0.0 and 1.0. Following this conversion, work on pixel level is converted to work on texel level.

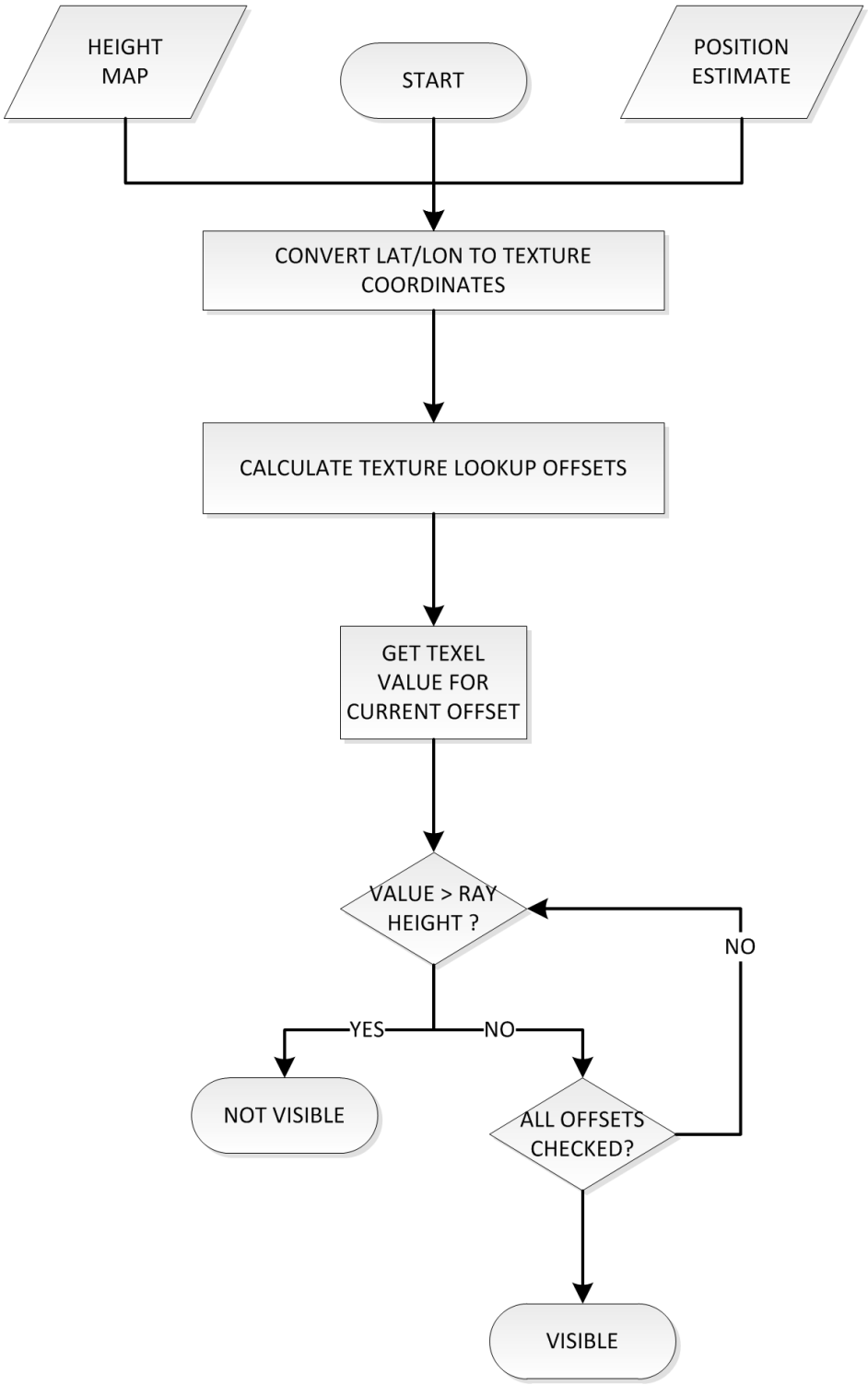

Figure 2.3: Flowchart for TBSVD.

3. A fixed number of sampling points are defined to check collision with the buildings in the vicinity. At each sampling point, a texture offset is calculated with respect to the GNSS receiver's texture coordinates. These offsets are added to the ground object's texture coordinates to check collision along the ray. Number of sampling points should be high enough so that the space between each step must be narrower than the width of any building, otherwise possible collision in a large distance can not be detected. In this study, number of sampling points with equivalent length are chosen to be 25 and this choice has not caused any false detection in a real urban area.

4. Starting at the maximum height, which is 1.0, and incrementally decreasing the ray height value about one sampling height resolution, corresponding texel offsets are calculated as described in the previous step of this algorithm. Then, the height value of the texel is compared with respect to the current height value. If the height value of the texel is less than the image value, ray is being occluded by the underlying structure and a collision is detected. The core of this algorithm is to transform sampled points on the ray to the texels on the height map.

5. Algorithm is terminated after all samplings are evaluated without detection of a collision, then the satellite is marked visible for the given azimuth and elevation angle.

A sample GLSL fragment shader code is presented in Listing 1.

```
// inputs
uniform float az,el;
sampler2D heigthmap;
varying vec2 position;
uniform float maxheight,pixelwidth;
uniform float stepsize;

vec2 calcOffsets(float h){
        float hpixel = (h*maxheight)/pixelwidth;
        float distpixel = hpixel/tan(el);
        return distpixel*vec2(sin(az),-cos(az));
}

bool isSatVisible(){
        for( float h = maxheight ; h>0 ; h-= stepsize){
                float hpixel = (h*maxheight)/pixelwidth;
                vec2 off = calcOffsets(h);

                float mapheight = texture2d(heightmap,position+off);
                if(mapheight > hpixel)
                        return false;
```

```
        }
        return true;
}
```

Listing 1: GLSL code for calculating single satellite visibility on GPU

2.3 Forward DOP Calculation

The presented ray marching algorithm calculates the visibility and multipath condition of individual satellite in the constellation from the point-of-view of a given point, hence this method can be used to enhance GNSS position measurement accuracy on a real-time basis and to approximate DOP values more realistically. DOP is a significant source of error in position estimate calculation as it indicates how well the satellite geometry is at a given moment. Without using a 3D map data or namely local obstructing effects, DOP values can be calculated in advance for a given location at a given time, however it is not possible to make sure that these DOP values will be accurate locally. A more realistic DOP calculation has to take into account satellite visibility and multipath constraints. The proposed method contributes to the calculation of a more realistic DOP value for any given location on map and for any given time in a computationally efficient manner. These values may be used to compute reliability of GNSS measurement in the areas on the map and time interval. A time-varying DOP or visibility map can be generated and distributed on the network to OBUs, which in turn can be used to help with local integrity calculation. For instance, in a GNSS based road user charging application, if we can determine for a given road segment, for a given time period, the positioning error will exceed the tolerance specified earlier, the application can use fallback methods or completely seize operation to prevent erroneous charging.

Satellites can be classified according to their visibility, this information can be used to calculate a more realistic DOP value and estimate positioning error. Our novelty image based method is suitable for calculation of GNSS error mitigation and estimation in parallel, very fast and deployment on a GPU. Visibility of each satellite and path condition of its traveling signal towards to the destination can be classified as illustrated in Fig. 2.4:

- Line-of-sight: Satellite is visible from the ground point,
- Multipath: Satellite is not visible from the ground point, but its signal can be reflected by nearby structures,
- Occluded: Satellite nor its reflections are not visible from the ground point.

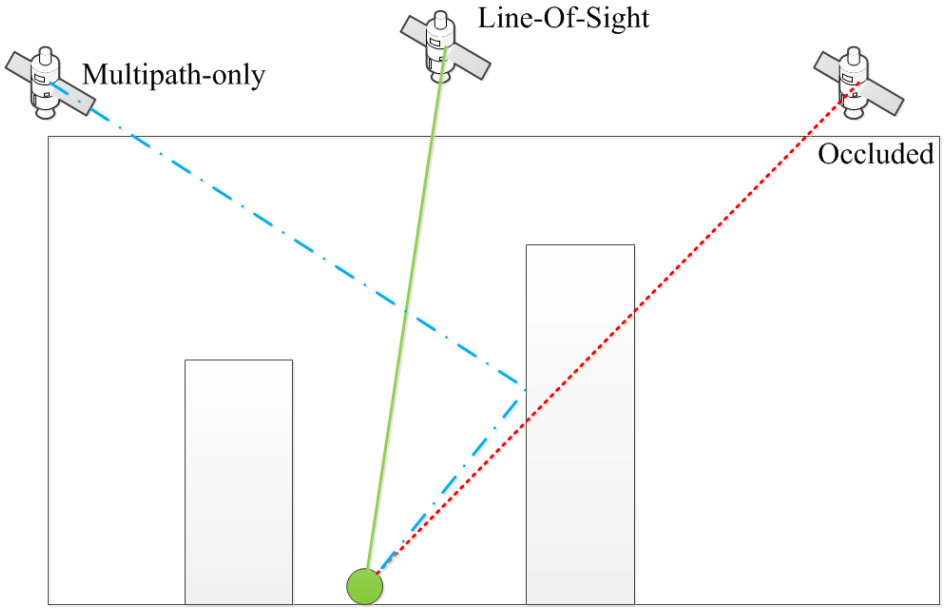

Figure 2.4: Satellite reception classes.

Following the classifications, some performance metrics for GNSS receiver can be approximately calculated as well. For instance, after checking the visibility status, 4 visible satellites are detected (i.e. azimuth ϕ_n elevation angle θ_n for each individual satellite is known, for $n = 1,2,3,4$), DOP values can be derived for the given location and time. Based on the relative positioning information of the GNSS receiver with respect to the 4 satellites, the matrix in 2.3 can be constituted Sturza (1983),

$$M = \begin{bmatrix} \cos\theta_1 \sin\phi_1 & \cos\theta_1 \cos\phi_1 & \sin\theta_1 & 1 \\ \cos\theta_2 \sin\phi_2 & \cos\theta_2 \cos\phi_2 & \sin\theta_2 & 1 \\ \cos\theta_3 \sin\phi_3 & \cos\theta_3 \cos\phi_3 & \sin\theta_3 & 1 \\ \cos\theta_4 \sin\phi_4 & \cos\theta_4 \cos\phi_4 & \sin\theta_4 & 1 \end{bmatrix} \tag{2.3}$$

A symmetric matrix in 2.4 in terms of the matrix M is calculated to asses the estimation errors for the corresponding GNSS measurement in terms of the well known variables such as $EDOP^2, NDOP^2, VDOP^2, TDOP^2$.

$$\left(M \times M^T\right)^{-1} = \begin{bmatrix} EDOP^2 & . & . & . \\ . & NDOP^2 & . & . \\ . & . & VDOP^2 & . \\ . & . & . & TDOP^2 \end{bmatrix} \tag{2.4}$$

For example, horizontal error estimate (HDOP) value, which can vary the precision of map matching algorithms significantly, can be calculated as follows: $HDOP = \sqrt{EDOP^2 + NDOP^2}$.

HDOP is useful in calculating estimated position error (EPE) which is the 95% limit for the horizontal positioning error.

$$EPE = HDOP \times \sqrt{URE^2 + UEE^2} \times 2 \tag{2.5}$$

User Range Error (URE) and User Equipment Errors (UEE) are the estimates of other errors in GNSS position tracking depending on various conditions from the receiver being used or atmospheric conditions at given moment.

After calculating visible satellites, it is straight-forward to apply these methods to calculate DOP values for a given map segment from the GPU. For example, a function of height map, time and sky geometry as input generating an image representing estimated error levels on each pixel is presented in Listing 2.

```
// inputs
uniform float time;
uniform float ure;
uniform float uee;
sampler2D heigthmap;
varying vec2 position;
uniform int numsatellites;
uniform float[30] az;
uniform float[30] el;

// variables
bool[30] satelliteVisibility;
uniform int[4] selectedSatellites;
mat4 M;

void calcSatelliteVisiblity(){...}
void selectSatellites(){...}

void main(){
    calcSatelliteVisibility();
    selectSatellites();

    int[4] s = selectedSatellites;

    M[0] = vec4(cos(el[s[0]])*sin(az[s[0]]),cos(el[s[1]])*sin(az[s↩
        [1]]),cos(el[s[2]])*sin(az[s[2]]),cos(el[s[3]])*sin(az[s[3]])↩
        );
    M[1] = vec4(cos(el[s[0]])*cos(az[s[0]]),cos(el[s[1]])*cos(az[s↩
        [1]]),cos(el[s[2]])*cos(az[s[2]]),cos(el[s[3]])*cos(az[s[3]])↩
        );
```

```
    M[2] = vec4(sin(el[s[0]]),sin(el[s[1]]),sin(el[s[2]]),sin(el[s↩
        [3]]));
    M[3] = vec4(1.0);

    mat4 dopMat = matrixCompMult(M,transpose(M));
    dopMat = inverse(dopMat);

    float edop2 = dopMat[0][0];
    float ndop2 = dopMat[1][1];

    float hdop = sqrt(edop2+ndop2);

    float rangeError = hdop*sqrt(pow(uee,2)+pow(ure,2))*2;

    gl_FragColor = vec4(rangeError);
}
```

Listing 2: GLSL code for parallel DOP calculation in GPU

2.4 Improving Positioning Accuracy

Previous work indicates two distinct method that use the 3D representations of urban areas to improve GNSS accuracy:

- The first method is to eliminate NLOS signals by reducing their weights in the equation of mutual positioning.
- The second method, which is called shadow matching Groves (2011), divides satellites into two sets of NLOS and LOS satellites by using a threshold SNR value, then these sets are matched to the locations on the 3D shadow maps, which is created by the satellite view for a given moment. After calculating a set of solutions subject to the given satellite visibility, closest solution with respect to the estimated position is selected as the matched position output.

We have experimented with both of these methods using our height map algorithm instead of polygonal raytracing.

For both of these methods, we first calculate a rough position estimate using Bancroft's Method (Bancroft (1985)). Using this position estimate, we define a region-of-interest (ROI).

For the weight-based method, for each pixel in this ROI, we calculate the weight as $w = \frac{\sum p_i}{n}$ where p_i the value of each pixel and n number of pixels in ROI. We multiply this weight to

the actual weight based on SNR value and feed it to the Weighted Least Square algorithm to finalize the solution.

In this thesis, we compared SNR based weights to 3D height map based weights for weighted least square positioning solutions. Parameters and final weight equations can be seen in Table 2.1.

Table 2.1: Satellite weights used in our tests

Method	SNR	TBSVD
Input	SNR: Signal-to-noise ratio	p_i:value of pixel
Parameters	$SNR_a, SNR_A, SNR_0, SNR_1$: Signal power constants (Li & Wu (2009))	n: number of pixels in ROI
Weight	$10^{\frac{SNR_1 - SNR}{SNR_a}} \times \left(\frac{\frac{SNR_A}{10^{SNR_1 - SNR_0}} - 1}{SNR_0 - SNR_1} \times (SNR - SNR_1) + 1 \right)$	$\frac{\sum p_i}{n}$

For the shadow matching method, we use the SNR value calculated by the receiver to compare it to our visibility maps. Estimated position is than refined using the set of probable points given by the algorithm.

2.5 Comparison to Other Methods

When compared to conventional raytracing over polygonal 3D data, our method has better space and time complexity. A 3D vertex requires 3 dimension components of 32 bits each and for each triangle 3 vertices have to be stored. In our tests, we found out that for the same amount detail, a small tile 256 meters wide requires about 800 triangles. Our method represents the same data with a 256x256 raster image with 2 bits per pixel storage. Also, raytracing requires a ray-triangle intersection algorithm, such as Möller-Trumbore method (Möller & Trumbore (1997)), which is computationally more expensive compared to texture look-ups (see Table 2.2).

Table 2.2: Comparison of polygonal raytracing and our method for a small area

Method	Polygon Raytracing	TBSVD
Storage	800 triangles (28 KB)	256x256 pixels (16 kb)
Algorithm	Triangle-Ray intersection	Texel look-up
Worst case	Tests each triangle (800 tests)	Constant number of look-ups (20 in our case)

Main advantage of texture-based methods such as our algorithm is that specialized hardware (GPU) can do all instructions for each pixel in parallel and most linear algebra instructions are exceedingly fast. As we discussed on previous sections, low-cost GPUs are becoming

more and more common and affordable. On-board units and navigation devices will use GPUs more and more in the future to provide better and more responsive user interfaces. Texture-based algorithms are a by-product of graphical power used for these user interfaces. Even if a GPU is not available, texture look-ups are faster to do in memory than raytracing in traditional way.

Another benefit from the extra performance gained by using our algorithm is that the methods used for improving GNSS positioning can be extended to create a more realistic simulation of signal propagation in real-time for urban areas. In this study, we only check for NLOS satellites, but more complex methods can be used. For example, rays can be bounced, reflected or refracted using the height map information on GPUs.

3 Results

We have used our method with a height map of 1 pixel per 1 square meter density and 8-bit per pixel depth values. We have generated our height maps using building outline information and number of floors registered in digital map database for Kadıköy Area in Istanbul, Turkey.

As it can be seen in Figure 3.1, height map at this configuration gives a good representation of the volumetric data of urban structures.

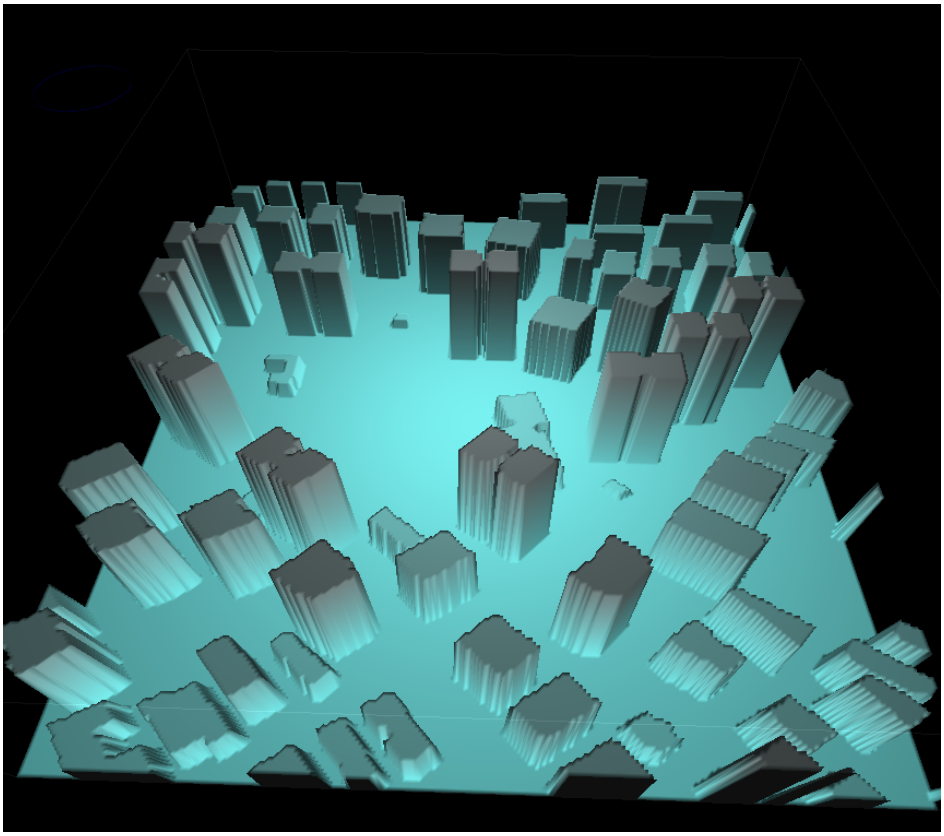

Figure 3.1: Portion of the height (elevation) map data represented in 3D. Elevation maps can be used to store urban city models more efficiently.

Running the algorithm for a single satellite creates a binary occlusion map as seen in Figure 3.2. These binary maps are then combined into a categorized occlusion or visibility map, where for each pixel or in our case every square meter, number of visible satellites, number of multipath-only satellites are known, as well as which satellites are visible and their visibility classes.

Using our method we can calculate detailed satellite visibility heat maps for any given time. Figure 3.3 shows the values generated by our method at different points of time. There are

Figure 3.2: Binary occlusion map generated from a single satellite. White areas can see the satellite, black areas can't. Notice the artifacts around shadows caused by limited number of steps used in our algorithm. Increasing the number of steps improves the results.

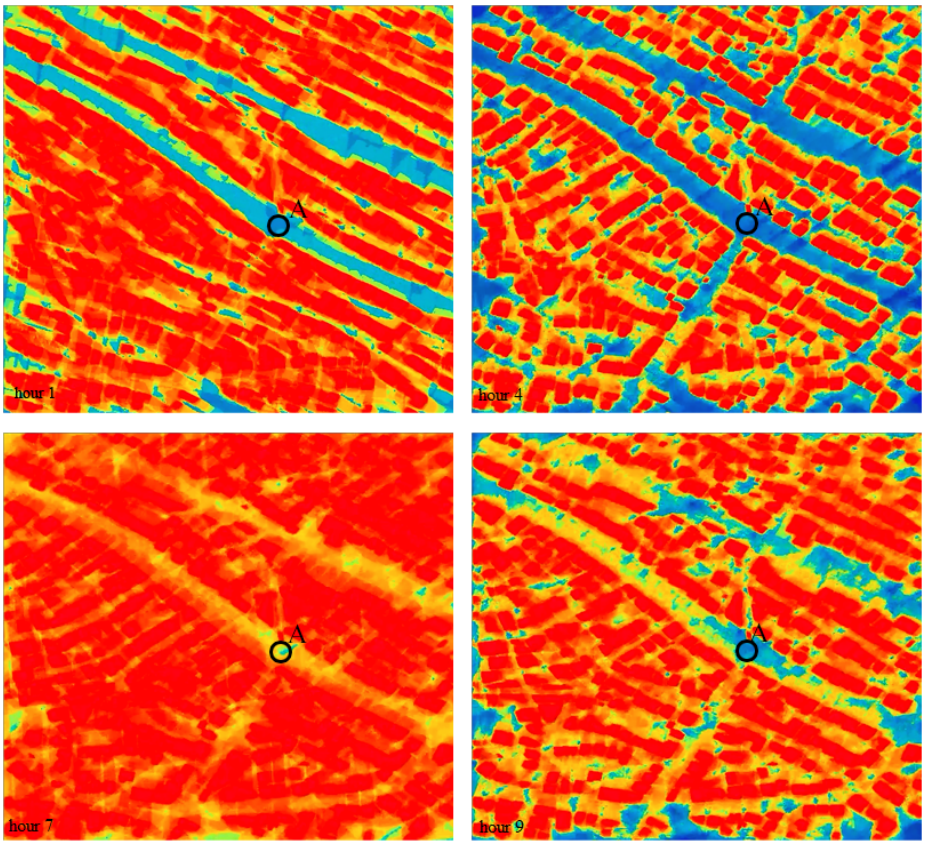

Figure 3.3: Satellite visibility heat maps at 4 different times. Blue: High visibility (>4 satellites), red low visibility (<4 satellites). Our method can be used to process height maps to generate satellite visibility information for any given time quickly by processing each tile (texel) in parallel on a GPU. For instance, we found that the junction at point A, while it has fairly good visibility throughout the day, at 0700 along with all neighboring streets and junctions, has low visibility.

huge differences in satellite visibility for some locations, hence our method can be used to determine those areas and time segments where integrity of the service is compromised. For instance, inspecting randomly selected junction (A), differences in satellite visibility at different hours can be seen. In our 24 hour test, while point A maintained high visibility (more than 4 satellites) compared to other locations for most of the day, at 7 it dropped to low visibility. Number of satellites visible for point A are 5, 6, 3, 4 at 1, 4, 7 and 9 o'clock respectively. While this information could be easily calculated in GNSS simulators before, using our method it is possible to generate and disseminate the 3D volumetric information using already existing technologies to receivers and calculate local visibility information on-board with ease.

For instance, point A in the Figure 3.3, is a junction with high satellite visibility throughout the day. However, heat maps generated using our method showed that at some hours of the day, it loses visibility for most of the satellites. Using raytracing, these calculations could only be done on high performance processors. With our method, a low cost OBU with a mobile GPU can do this calculations real-time on large texture maps, up-to 4096x4096 pixels at one time.

In our main experiments, we have done urban road tests and measured errors of the commercial receiver and our own software solution (with same observation data) which uses signal-to-noise ratio weights for its least square solver. Than in our software solution we have switched the weighting algorithm to one which uses our ray-marching method on height maps to give lower weights to low visibility satellites.

We have done these real-world experiments using U-Blox Series 6 receiver and compared results to the ground truth. We used the raw data gathered from the device using propriety UBX protocol (U-blox (2011)). A generalized flow of our method can be seen in Figure 3.4.

First experiment was done in an urban track in Kadıköy İstanbul. As shown in Table 3.1, refined weights have shown significant improvements over SNR based weighting method. CDF graph (see Figure 3.6) shows the improved distribution of refined weights over both SNR based method and receiver itself. However, it also shows that although very few there are some solutions with very high errors compared to other methods. These errors are mainly caused by mismatches between 3D data and real environment. As height maps are only a 2D representation of very complex structures, visibility information extracted from them is not exact.

A graphical comparison of tracks in Figure 3.5 shows that height map based weighting refined the actual solutions in many areas, except in some places where resolution of the 3D maps and limited number of steps used in shadow map calculation gave noisy results. Apart

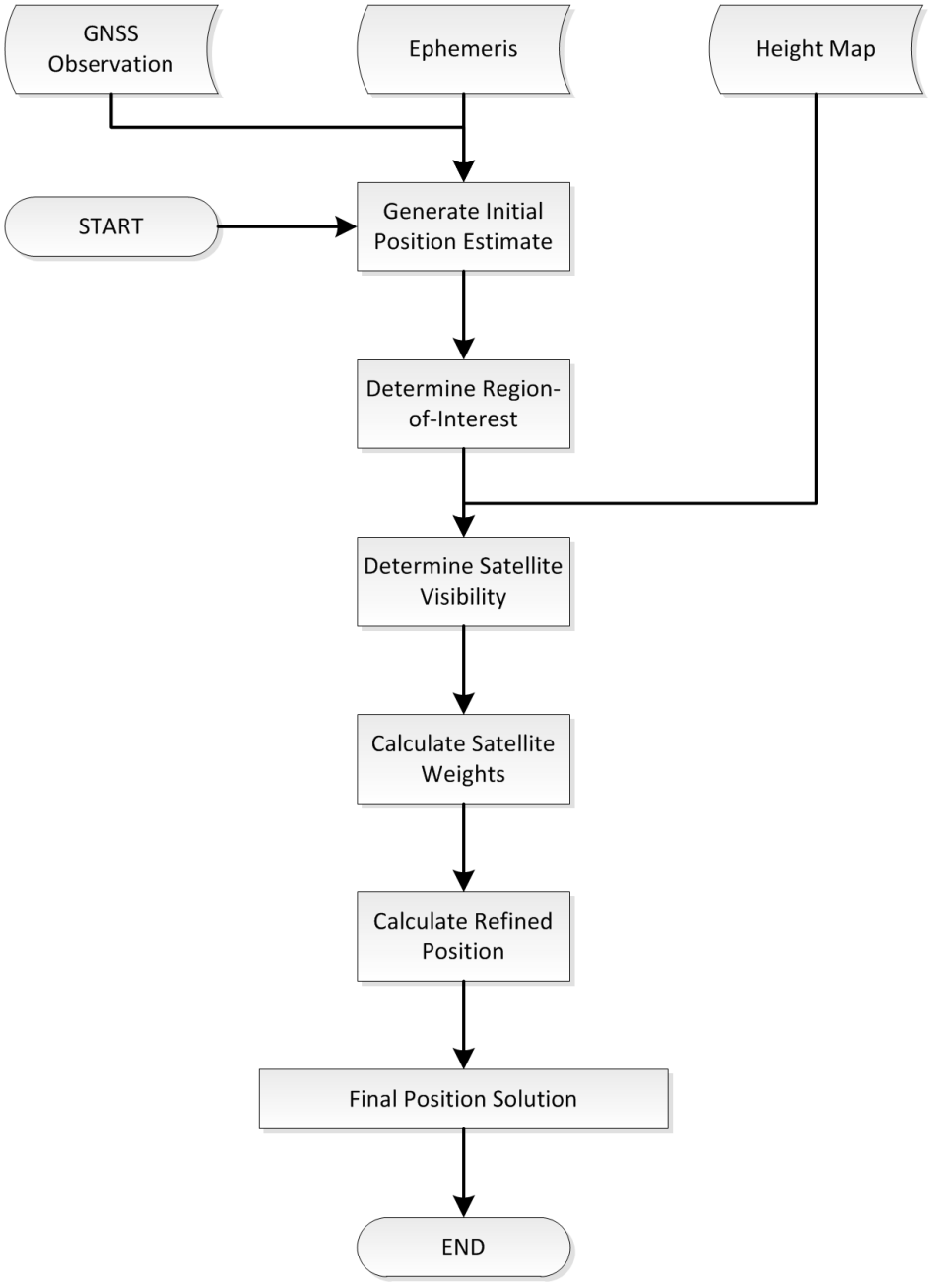

Figure 3.4: Flowchart for refining GNSS positioning with TBSVD

from the U-Blox solution, all tests used epoch-by-epoch solutions and no filtering was used. Shadow matching, while generating less accurate results in average, gave better solutions in some areas. We conducted another experiment but this time on a longer circuit in İstinye, İstanbul. This track is 4 kilometers long and has both open sky areas and urban canyon areas. As seen in Table 3.3 , our refined weights gave significantly improved accuracy over SNR based weights.

Another experiment in same area was conducted, but this time with a stationary receiver. Our software solver could not match the smooth results generated by the U-Blox receiver in stationary test, but as seen in Table 3.2 weights computed with our algorithm proved better than SNR weights.

Table 3.1: Position Accuracy (m) for Kadıköy Experiment (mobile)

Configuration	Median	Max	DRMS	95%	CEP
U-Blox	5.31	13.04	6.19	12.38	5.94
SNR Weights	4.67	20.02	6.55	13.10	6.29
TBSVD Weighs	3.48	16.85	5.22	10.43	5.00
TBSVD Shadow Match	6.68	15.42	7.73	15.46	7.42

Table 3.2: Position Accuracy (m) for Kadıköy Experiment (stationary)

Configuration	Median	Max	DRMS	95%	CEP
SNR Weights	4.82	7.24	4.99	9.99	4.79
TBSVD Weights	4.49	8.02	4.92	9.84	4.72

Table 3.3: Position Accuracy (m) for İstinye Experiment

Configuration	Median	Max	DRMS	95%	CEP
SNR Weights	5.88	75.55	18.39	36.78	17.65
TBSVD Weights	2.38	71.0	21.48	42.96	20.62

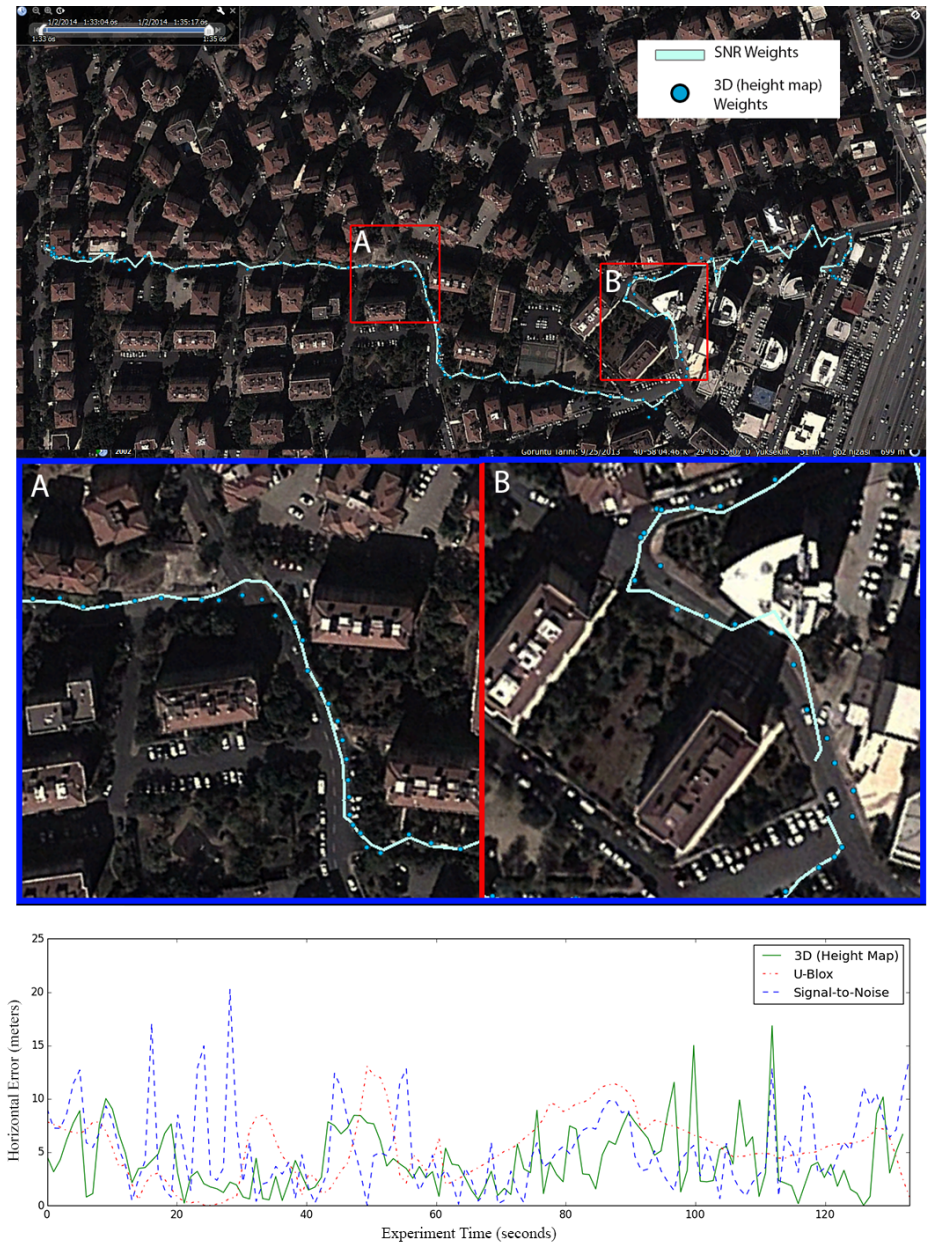

Figure 3.5: On top, tracks generated by both methods from our first Kadıköy experiment overlaid over the digital map. Especially in some areas (e.g. A and B) , TBSVD significantly improved the path towards the ground truth compared to SNR based weighting. Bottom, comparison of horizontal position errors for each method over time.

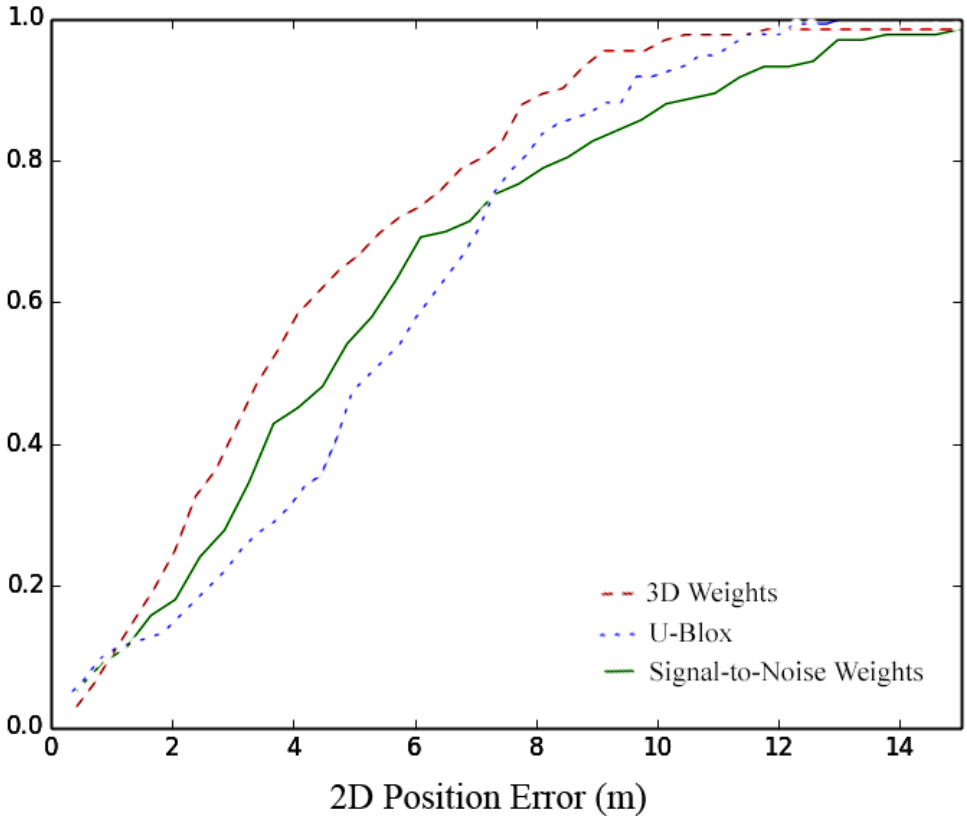

Figure 3.6: CDF for Kadıköy Experiment (mobile) shows how our height map based weights improved positioning.

4 Conclusion

Several methods exist for using 3D data in urban areas to detect degraded signals and studies indicate that these methods do improve GNSS positioning accuracy. However, their application in real-world ITS station receivers or other low-cost receivers is problematic, because they rely on polygonal raytracing algorithms which are computationally expensive and require high end processors. In this study, we present a novel approach called Texture Based Satellite Visibility Detection for leveraging digital map data for GNSS positioning improvement. It uses texture-based algorithms inspired by high performance Computer Graphics techniques, which uses 2D height maps instead of 3D models and replaces raytracing with a much more simple 2D ray-marching. Therefore, it is better in both time and space complexity compared to polygonal raytracing methods.

TBSVD alone is used for satellite visibility detection. Combining it with other techniques has many advantages. In this study, for instance, we have explored two different approaches to use the information TBSVD generates for improving GNSS accuracy. First method, we used was using TBSVD based weights compared to Signal-to-Noise based weights for the Weighted Least Squares solution. Other method was shadow matching, a technique where the position solution is refined matching the set of visible satellites from the receiver to the set of visible satellites calculated on the TBSVD map. We have conducted real-world experiments in Istanbul, Turkey. Both methods showed that TBSVD can be used to refine GNSS solutions better than SNR-based weighting. Visual examination of refined tracks shows that TBSVD weighting provides better results than SNR-based method, especially in problematic areas where large parts of the sky is occluded by tall buildings.

Both previous work and our experiments show the potential of using 3D information for improving GNSS positioning. 3D urban data gives the devices context-awareness unlike before and we believe that with increased processing power of GNSS receivers and OBUs, new ways to mitigate degraded signals can be found in this area. Currently, mobile GPUs are getting even more common and affordable then before, which will further focus interest in texture and shading based algorithms to be used in problems unrelated to computer graphics. Also, low-cost graphical processing units are being more and more common in navigation and entertainment systems used in vehicles. For such systems texture-based algorithms like ours will work significantly faster, leaving processing power for more complex algorithms.

We believe that graphical methods and simulations can be further improved and leveraged in GNSS navigation. GNSS signal propagation can be more accurately simulated and this information can be used to mitigate NLOS signal errors. Also, height maps can be used

to determine areas in urban environments where GNSS performance is limited beforehand, thus adjusting importance of other methods in hybrid positioning solutions.

References

Bancroft, S. (1985). An algebraic solution of the gps equations. *Aerospace and Electronic Systems, IEEE Transactions on*(1), 56–59.

Bauer, S., Obst, M., Streiter, R., & Wanielik, G. (2013). Evaluation of shadow maps for non-line-of-sight detection in urban gnss vehicle localization with vanets, the gain approach. In *at ieee conference on vehicular technology.*

Bauer, S., Obst, M., & Wanielik, G. (2012). 3d environment modeling for gps multipath detection in urban areas. In *Systems, signals and devices (ssd), 2012 9th international multi-conference on* (pp. 1–5).

Groves, P. D. (2011). Shadow matching: A new gnss positioning technique for urban canyons. *Journal of Navigation*, *64*(03), 417–430.

Heckbert, P. S. (1989). *Fundamentals of texture mapping and image warping*. Unpublished master's thesis, Citeseer.

Imagination. (2014). *Imagination powervr automotive page.* Retrieved 2014-01-31, from `http://www.imgtec.com/markets/automotive.asp`

Li, J., & Wu, M. (2009). The improvement of positioning accuracy with weighted least square based on snr. In *Wireless communications, networking and mobile computing, 2009. wicom'09. 5th international conference on* (pp. 1–4).

McGuire, M., & McGuire, M. (2005). Steep parallax mapping. *I3D 2005 Poster*, 23–24.

Möller, T., & Trumbore, B. (1997). Fast, minimum storage ray-triangle intersection. *Journal of graphics tools*, *2*(1), 21–28.

Ogniewski, J., Karlsson, A., & Ragnemalm, I. (2011). Texture compression in memory- and performance-constrained embedded systems. In *Computer graphics, visualization, computer vision and image processing 2011* (pp. 19–26).

Peyraud, S., Bétaille, D., Renault, S., Ortiz, M., Mougel, F., Meizel, D., & Peyret, F. (2013). About non-line-of-sight satellite detection and exclusion in a 3d map-aided localization algorithm. *Sensors*, *13*(1), 829–847.

Risser, E. A., Shah, M. A., & Pattanaik, S. (2005). Interval mapping. *University of Central Florida Technical Report. Available online at http://graphics. cs. ucf. edu/IntervalMapping/images/IntervalMapping. pdf*.

Sturza, M. A. (1983). Gps navigation using three satellites and a precise clock. *NAVIGATION: Journal of the Institute of Navigation*, *30*(2).

Torres, E. (1990). Optimization of the binary space partition algorithm (bsp) for the visualization of dynamic scenes. In *Eurographics* (Vol. 90, pp. 507–518).

U-blox. (2011, Apr). *u-blox 6 receiver description.* Including Protocol Specification. Retrieved from `http://dlnmh9ip6v2uc.cloudfront.net/datasheets/Sensors/GPS/760.pdf`

Wang, L., Groves, P. D., & Ziebart, M. K. (2012). Multi-constellation gnss performance evaluation for urban canyons using large virtual reality city models. *Journal of Navigation*, *65*(03), 459–476.

Biographical Sketch

The author, Can Göçmenoğlu, was born on February 15, 1987 in İstanbul, Turkey. He graduated from Galatasaray High School in 2006. In 2012, he received his Bachelor of Science in Computer Engineering from Galatasaray University.

In 2012, he was admitted at the Galatasaray University under the Master of Science in Computer Engineering program, specializing in Intelligent Transportation System Applications and Simulations. Aside from his post-graduate studies, he is participating in research activities for GLOVE project under EU 7th Framework program.

Publications

1. Uluer, P., Göçmenoğlu, C., & Acarman, T. (2012, July). Evaluation of Drivers Authority in a Structured Set of Driving Tasks and Decisions: Preliminary Results on Vehicle Simulator Study. In *ASME 2012 11th Biennial Conference on Engineering Systems Design and Analysis* (pp. 803-811). American Society of Mechanical Engineers.