T.C.
KONYA TEKNİK ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

3 BOYUTLU COĞRAFİ VERİLERDEN OTOMATİK VE HİYERARŞİK OLARAK, OGC STANDARDI 3D TILES'LARIN OLUŞTURULMASI

Rümeysa KABAKCI

YÜKSEK LİSANS

Bilgisayar Mühendisliği Anabilim Dalı

Mayıs-2024
KONYA

TEZ KABUL VE ONAYI

Rümeysa KABAKCI tarafından hazırlanan "3 Boyutlu Coğrafi Verilerden Otomatik ve Hiyerarşik Olarak, OGC Standardı 3D Tiles'ların Oluşturulması" adlı tez çalışması 24/05/2024 tarihinde aşağıdaki jüri tarafından oy birliği ile Konya Teknik Üniversitesi Lisansüstü Eğitim Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı'nda YÜKSEK LİSANS olarak kabul edilmiştir.

Jüri Üyeleri	**İmza**
Başkan Prof.Dr. Erkan ÜLKER (Konya Teknik Üniversitesi)	……..
Danışman Prof.Dr. Harun UĞUZ (Konya Teknik Üniversitesi)	……..
Üye Doç.Dr. Hüseyin HAKLI (Necmettin Erbakan Üniversitesi)	……..

Yukarıdaki sonucu onaylarım.

Prof. Dr. Mevlüt UYAN
Enstitü Müdürü

.

TEZ BİLDİRİMİ

Bu tezdeki bütün bilgilerin etik davranış ve akademik kurallar çerçevesinde elde edildiğini ve tez yazım kurallarına uygun olarak hazırlanan bu çalışmada bana ait olmayan her türlü ifade ve bilginin kaynağına eksiksiz atıf yapıldığını bildiririm.

DECLARATION PAGE

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.

Rümeysa KABAKCI

Tarih:

ÖZET

YÜKSEK LİSANS

3 BOYUTLU COĞRAFİ VERİLERDEN OTOMATİK VE HİYERARŞİK OLARAK, OGC STANDARDI 3D TILES'LARIN OLUŞTURULMASI

Rümeysa KABAKCI

Konya Teknik Üniversitesi
Lisansüstü Eğitim Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı

Danışman: Prof. Dr. Harun UĞUZ

2024, 90 Sayfa

Jüri

Prof. Dr. Erkan ÜLKER
Prof. Dr. Harun UĞUZ
Doç. Dr. Hüseyin HAKLI

3D Tiles, coğrafi bilgi sistemlerinde 3 boyutlu verilerin temsil edilmesi ve depolanması için geliştirilmiş bir standarttır. Geleneksel 2 boyutlu harita ve görüntülerin yanı sıra, artan taleplerle birlikte 3 boyutlu verilerin kullanımı da hızla artmaktadır. Ancak, büyük ve karmaşık 3 boyutlu verilerin etkili bir şekilde depolanması, işlenmesi ve aktarılması zor olabilir. 3D Tiles, bu zorlukların üstesinden gelmek için geliştirilmiştir. Bu standart, büyük miktarda 3 boyutlu veriyi hiyerarşik olarak düzenler ve parçalara ayırır, böylece verilerin daha hızlı yüklenmesi ve işlenmesi mümkün hale gelir.

Tez çalışması kapsamında, 3 boyutlu coğrafi verilerin OGC (Open Geospatial Consortium) standardı olan 3D Tiles formatına otomatik ve hiyerarşik bir şekilde dönüştürülmesi incelenmiştir. OGC, coğrafi bilgi sistemleri ve lokasyon tabanlı hizmetler için uluslararası standartlar geliştiren bir organizasyondur. Çalışma kapsamında hem açık kaynaklı hem de verinin 3D Tiles setlerine dönüşüm aşamasına kadar hiçbir müdahale gerektirmeyen bir uygulama geliştirilmiştir. Bu uygulama, veri yoğunluğuna göre hiyerarşik bir bölümleme yapabilme yeteneğine sahiptir. Tez çalışmasında, hiyerarşik bölümleme probleminin çözümü için QuadTree, Octree, Kd-Tree, R-Tree ve önerilen Optimize QuadTree algoritmaları uygulanmıştır. Bu algoritmaların kullanımıyla elde edilen sonuçlar; oluşturulma süreleri, ağaç yapısı ve dosya boyutları açısından karşılaştırılarak en ideal bölümleme yöntemi belirlenmiştir.

Sonuç olarak, bu çalışma, 3 boyutlu coğrafi verilerin 3D Tiles formatına otomatik ve hiyerarşik bir şekilde dönüştürülmesi sürecinde literatüre önemli bir katkı sağlamıştır. Elde edilen sonuçlar, büyük ölçekli üç boyutlu coğrafi verilerin verimli bir şekilde akışı ve 3D Tiles setlerine dönüştürülmesi aşamasında başarılı değerler elde edilmiştir. Bu dönüşüm süreci; açık kaynaklı, müdahale gerektirmeyen ve veri yoğunluğuna göre hiyerarşik bir bölümleme yapabilme yeteneğine sahiptir ve oluşturma süreleri ile dosya boyutu açısından daha verimli sonuçlar sunmaktadır. Bu çalışma, gelecekteki kentsel planlama ve modelleme çalışmaları için önemli bir referans niteliği taşımaktadır.

Anahtar Kelimeler:3D Tiles**,** Hiyerarşik Bölümleme, 3D Coğrafi Veri, Optimize QuadTree

ABSTRACT

MS THESIS

CREATING OGC STANDARD 3D TILES AUTOMATIC AND HIERARCHICALLY FROM 3D GEOGRAPHICAL DATA

Rümeysa KABAKCI

Konya Technical University
Institute of Graduate Studies
Department of Computer Engineering

Advisor: Prof. Dr. Harun UĞUZ

2024, 90 Pages

Jury
Prof. Dr. Erkan ÜLKER
Prof. Dr. Harun UĞUZ
Doç. Dr. Hüseyin HAKLI

3D Tiles is a standard developed for representing and storing 3D data in geographic information systems. In addition to traditional 2D maps and images, the use of 3D data is also rapidly increasing with increasing demands. However, large and complex 3D data can be difficult to effectively store, process and transfer. 3D Tiles was developed to overcome these challenges. This standard hierarchically organizes and segments large amounts of 3D data, making it possible to load and process data faster.

Within the scope of the thesis, automatic and hierarchical conversion of 3D geographical data into 3D Tiles format, which is the OGC (Open Geospatial Consortium) standard, was examined. OGC is an organization that develops international standards for geographic information systems and location-based services. Within the scope of the study, an application that is both open source and does not require any intervention until the data conversion stage into 3D Tiles sets has been developed. This application has the ability to perform hierarchical partitioning according to data density. In the thesis study, QuadTree, Octree, Kd-Tree, R-Tree and the proposed Optimized QuadTree algorithms were applied to solve the hierarchical partitioning problem. The results obtained by using these algorithms; The most ideal partitioning method was determined by comparing creation times, tree structure and file sizes.

As a result, this study has made a significant contribution to the literature in the automatic and hierarchical conversion of 3D geographic data into 3D Tiles format. The results obtained were successful in the efficient flow of large-scale three-dimensional geographical data and their conversion into 3D Tiles sets. This transformation process; It is open source, does not require intervention and has the ability to perform hierarchical partitioning according to data density and offers more efficient results in terms of creation times and file size. This study serves as an important reference for future urban planning and modeling studies.

Keywords: 3D Tiles**,** Hierarchical Partitioning, 3D Geographic Data, Optimize QuadTree

ÖNSÖZ

Bu çalışmanın yürütülmesinde beni bilgi ve tecrübesiyle yönlendiren, destek ve yardımcı olan, danışman hocam Prof. Dr. Harun UĞUZ' a, Konya Teknik Üniversitesi Mühendislik ve Doğa Bilimler Fakültesi Bilgisayar Mühendisliği Bölümünün tüm öğretim elemanlarına, değerli katkılarıyla çalışmama destek veren çalışma arkadaşım Talha ÖZTÜRK'e, çalışmalarımda gösterdiği ilgi, anlayış ve bana verdiği destek için eşim Barış KABAKCI'ya, bugünlere gelmemde büyük pay sahibi olan annem Fatma AKDOĞAN ve babam Ziya AKDOĞAN'a teşekkür ederim.

Rümeysa KABAKCI
KONYA-2024

İÇİNDEKİLER

SİMGELER VE KISALTMALAR

Kısaltmalar

GLTF : Grafik Kitaplığı İletim Formatı (Graphics Library Transmission Format)

3D : Üç Boyutlu (Three Dimensional)

BV : Sınırlayıcı Hacim (Bounding Volume)

B3DM : Toplu Üç Boyutlu Model (Batched Three Dimensional Model)

I3DM : Örnek Üç Boyutlu Model (Instanced Three Dimensional Model)

LOD : Ayrıntı Düzeyi (Level of Detail)

HLOD : Hiyerarşik Ayrıntı Düzeyi (Hierarchical Level of Detail)

SSE : Ekran Alanı Hatası (Screen-Space Error)

FME : Özellik Manipülasyon Motoru (Feature Manipulation Engine)

OGC : Açık Coğrafi Konsorsiyum (Open Geospatial Consortium)

IFC : İnşaat Endüstrisi Temel Sınıfları (Industry Foundation Classes)

SH : Sınırlayıcı Hacim

3DCM : Üç Boyutlu Şehir Modeli (Three Dimensional City Model)

1. GİRİŞ

1.1. 3D Tiles Nedir?

3D Tiles, masaüstü, web ve mobil uygulamalarda büyük boyutlu heterojen 3D jeo-uzamsal içeriği paylaşmak, görselleştirmek, kaynaştırmak ve bunlarla etkileşim kurmak için kullanılan açık kaynaklı bir özelliktir.

3D Tiles, veri sağlayıcıların ve uygulama geliştiricilerinin büyük ve karmaşık 3D bilgilerini her türlü araç ve uygulamada daha erişilebilir, birlikte çalışabilir ve kullanışlı hale getirmelerine olanak tanır. 3D Tiles, çalışma zamanında verimli bir şekilde seçme, sorgulama, filtreleme ve stil oluşturma gibi etkileşimlere izin vermek için özellik başına meta verileri korur.

Bu tez çalışmasında, farklı bölümleme algoritmaları kullanarak en ideal olanını tespit etmek, tespit edilen bölümleme algoritmasını açık kaynak "Node.js" modülü olan "obj23dtiles.js" içine entegre ederek tek bir obje sınırlayıcı hacim (SH), oluşturmak yerine bölgesel hiyerarşiyi sağlayarak tile yapısını açık kaynaklı bir modülle başarmak hedeflenmektedir.

1.2. Neden 3D Tiles'a İhtiyaç Duyulmaktadır?

Teknolojinin sürekli ilerlemesiyle birlikte, 3D veri ve görselleştirmenin kullanımı giderek daha geniş bir yelpazede yaygınlaşmaktadır. Mimarlık, şehir planlama, jeo-uzamsal analiz, artırılmış gerçeklik ve birçok başka alanda bu teknolojilerin kullanımı artmaktadır. Ancak, büyük 3D veri kümeleriyle uğraşmak, bu verilerin iletilmesi, depolanması ve etkili bir şekilde işlenmesi açısından zorluklar içerebilir (Xu ve dig., 2020). Tile yapısı, büyük bir 3D veri kümesini daha yönetilebilir hale getirmek için kullanılan bir yaklaşımı temsil eder. Bu yaklaşım, veriyi ihtiyaç duyulduğunda aşamalı olarak yüklenebilen ve işlenebilen daha küçük bölgelere bölerek büyük veri kümesini yönetilebilir parçalara ayırmayı amaçlar. Bu yaklaşımın birkaç önemli avantajı bulunmaktadır. İlk olarak tile yapısının kullanılması, büyük 3D veri kümelerini daha verimli bir şekilde internet üzerinden iletmeyi sağlar. Tüm veri kümesini tek seferde yüklemeye ve işlemeye çalışmak yerine, veri kümesini daha küçük parçalara bölerek daha hızlı iletilip işlenebilmesine imkan tanır (Chaturvedi, 2015). Bu özellik, özellikle yavaş veya güvenilmez internet bağlantılarıyla karşılaşıldığında önemlidir. Bu sayede, büyük 3D veri kümeleri daha etkili bir şekilde yönetilebilir ve kullanıcılar, ihtiyaç duydukları veriyi daha hızlı ve verimli bir şekilde elde edebilirler.

Veri iletimini daha verimli hale getirmenin yanı sıra, tile yapısını kullanmak 3D veri kümelerini depolamayı ve almayı da kolaylaştırabilir. Veri kümesinin tamamını tek

bir dosyada veya veri tabanında depolamak yerine, veri kümesi bağımsız olarak depolanıp alınabilen daha küçük parçalara bölünebilir. Bu, büyük veri kümelerine dayanan 3D uygulamaların ölçeklenebilirliğini ve performansını artırabilir ve veri kümesinin zaman içinde güncelleme veya değiştirme sürecini basitleştirebilir.

Tile kullanmanın bir diğer önemli avantajı, 3D verilerin daha verimli bir şekilde işlenmesine olanak tanımasıdır. Uygulamalar, yalnızca ihtiyaç duyulan tileları anında yükleyip işleyerek, daha yavaş veya güç bakımından sınırlı cihazlarla bile 3D verilerin daha sorunsuz ve hassas bir şekilde işlenmesini sağlayabilir. Bu özellik, özellikle sanal ve artırılmış gerçeklik gibi gerçek zamanlı oluşturma ve etkileşimin kritik olduğu uygulamalarda büyük bir avantaj sağlar. Bu sayede, kullanıcılar daha düşük sistem gereksinimlerine sahip cihazlarda bile etkileyici 3D deneyimler yaşayabilirler (Chen ve dig., 2018). Tile yapısı, uygulamaların gereksinimleri doğrultusunda dinamik bir şekilde veri yüklemesine ve işlemesine olanak tanıdığından, performans artırıcı bir rol üstlenir ve kullanıcı deneyimini olumlu yönde etkiler.

Tile yapısı, 3D verilerle çalışmayı ve bunları değiştirmeyi de kolaylaştırabilir. Bir veri kümesini daha küçük parçalara bölerek, verileri daha ayrıntılı bir düzeyde analiz etmek ve işlemek mümkün hale gelir. Bu, genellikle 3D verilerin ayrıntılı analizinin gerekli olduğu şehir planlama ve jeo-uzamsal analiz gibi çeşitli uygulamalar için yararlı olabilir (Hu, 2020).

Genel olarak, 3D verilerin ve tile yapısının kullanımı, çeşitli alanlarda giderek daha önemli hale gelmektedir. Bu durum, büyük veri kümelerinin daha verimli ve etkili bir şekilde işlenmesine, daha gerçekçi ve sürükleyici görselleştirmelere, 3D verilerin daha ayrıntılı analizine olanak tanımaktadır. Teknoloji ilerlemeye devam ettikçe, 3D verilerin ve tile yapısının öneminin artarak birçok farklı alanda yenilikler getirmesi beklenmektedir.

1.3. 3D Tiles'ın Avantajları

Modülerlik; 3D Tiles, karmaşık modellerin ve sahnelerin daha küçük, yönetilebilir parçalara bölünerek oluşturulmasına olanak tanır. Bu büyük ve karmaşık modellerle çalışmayı ve gerektiğinde değişiklik veya güncelleme yapmayı kolaylaştırır.

Verimli işleme; 3D Tiles, verimli işleme için optimize edilebilir; bu alt uç donanımlarda bile hızlı ve sorunsuz bir şekilde görüntülenebilecekleri anlamına gelir. Bu, özellikle akıcı ve sürükleyici bir deneyim için yüksek kare hızlarının kritik olduğu coğrafi bilgi sistem analizleri, video oyunları gibi uygulamalarda önemlidir.

Yeniden kullanılabilirlik; 3D Tiles birden çok modelde ve sahnede yeniden kullanılabilir, bu da zamandan ve kaynaklardan tasarruf sağlar. Bu aynı zamanda geliştiriciler arasında daha kolay iş birliğine olanak tanır, çünkü bireysel tilelar diğer geliştiriciler arasında oluşturulabilir ve paylaşılabilir.

Sanal ve artırılmış gerçeklik; 3D Tiles, sanal ve artırılmış gerçeklik ortamlarında sürükleyici deneyimler oluşturmak için kullanılır. Bu, özellikle 3D görselleştirmenin bir alanın tasarımını ve yerleşimini daha iyi anlamayı sağlayabildiği mimari ve şehir planlama gibi uygulamalarda kullanışlıdır.

Uyumluluk; 3D Tiles, çeşitli araçlar ve platformlar tarafından geniş çapta desteklenerek, modellerin ve sahnelerin farklı platformlar ve cihazlar arasında aktarılmasını ve kullanılmasını kolaylaştırır.

Etkileşim; 3D Tiles verileri etkileşimli hale getirilerek, kullanıcıların etkileşimli ve dinamik animasyonlarla daha zengin deneyimler yaşamasına olanak tanır.

3D Tiles'ın birincil amacı, büyük heterojen veri kümelerinin akış ve işleme performansını iyileştirmektir. 3D Tiles temeli, hiyerarşik ayrıntı düzeyini (HLOD) etkinleştiren bir uzamsal veri yapısıdır, böylece yalnızca görünür tile'lar için tüm veriyi aynı anda işlemeye çalışmak yerine, belirli bir 3D görünüm için en önemli olan tile'ların veri akışını sağlar. (Cozzi ve Lilley, 2022).

Şekil 1. 3 Boyutlu Bina Verileri (Cozzi ve Lilley, 2022)

Şekil 2. Seyrek bir ızgara düzeninde kutucuklar için sınırlayıcı hacim (Cozzi ve Lilley, 2022)

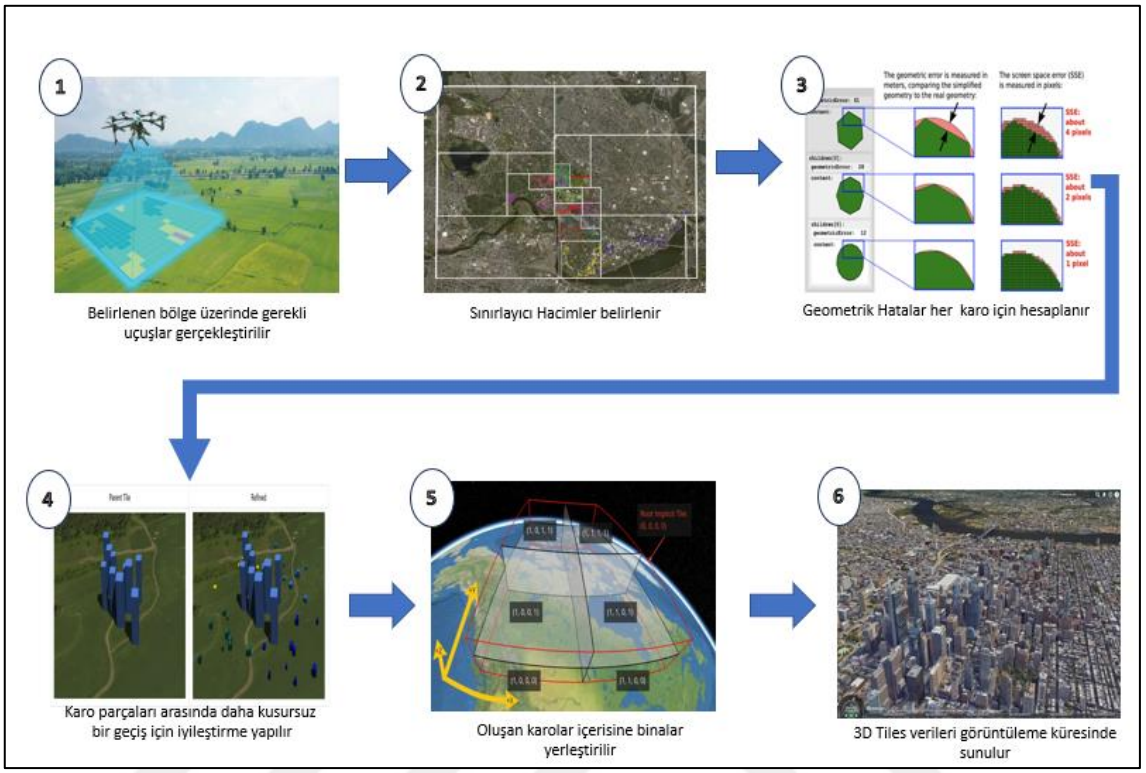
3D Tiles, Kd-Tree, QuadTree, Octree gibi uzamsal veri yapıları ve ızgaralar gibi unsurları içeren çeşitli uzamsal veri yapılarına dayanarak, 3D alanı uyarlanabilir uzamsal alt bölümlere ayırmak için kullanılabilir (Usta, 2021). Dönüştürme araçları, katı uzamsal alt bölümler yerine, bir veri kümesini daha esnek ve uyarlanabilir alt bölümlere bölme yeteneğine sahiptir. Bu, örneğin her bir modelin oluşturulma maliyeti ve modellerin dağılımına göre veri kümesini dengeli bir şekilde bölme olanağı sağlar. Bu uyarlanabilir uzamsal alt bölümler, verinin doğası ve dağılımına daha iyi uyum sağlayarak işleme süreçlerini optimize edebilir ve daha etkili bir veri yapısı oluşturabilir. Özellikle büyük ve karmaşık 3D veri kümeleriyle çalışırken performansı artırabilir ve işleme süreçlerini daha etkin hale getirebilir (Usta, 2021). Çeşitli uzamsal veri yapıları, 3D veri analizi ve görselleştirmesi alanında daha iyi ölçeklenebilirlik ve verimlilik sağlamak için kullanılabilecek önemli araçlardır.

1.4. 3D Tiles Aşamaları

3D Tiles verilerinin oluşturulma süreci ve görüntülerin işlenmesi, yüksek bir teknik beceri ve bilgi gerektirir. Stereoskopik eşleştirme, fotogrametri ve diğer görüntü işleme teknikleri karmaşık olabilir. 3D modelin doğru bir şekilde dönüştürülmesi ve tiles setleri haline getirilmesi, veri kaybını önlemek ve verimli bir tileset oluşturmak için dikkatli bir çalışma gerektirir.

"tileset.json" dosyasının doğru bir şekilde yapılandırılması ve gereksinimlere uygun olarak düzenlenmesi önemlidir. Bu, verilerin doğru bir şekilde sunulmasını sağlar. Küre üzerinde etkileşimli bir şekilde sunulan 3D modelin performansı ve kullanıcı deneyimi, web uygulamasının optimize edilmesi ve uygun bir 3D görüntüleme

motorunun kullanılmasıyla ilgilidir. Elde edilen bir görüntünün 3D Tiles formatına dönüştürülüp küre üzerinde sunulmasına kadar olan sürecin işlem adımları Şekil 3'te verilmiştir.



Şekil 3. 3D Tiles işlem adımları

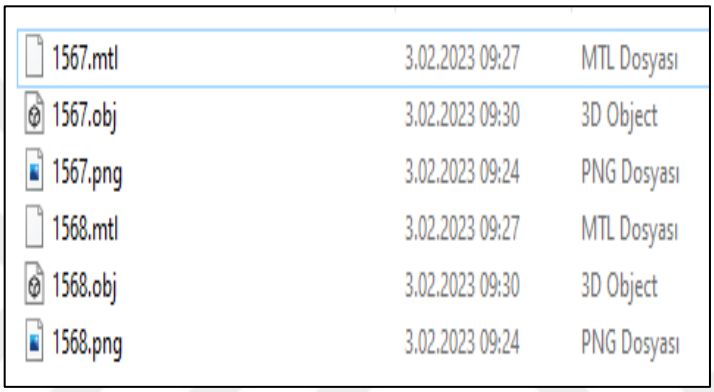
İlk adım, hedef alanın dronlar, uçaklar veya diğer uydu sistemleri aracılığıyla görüntülenmesidir. Bu, genellikle yüksek çözünürlüklü kameralar veya sensörler kullanılarak gerçekleştirilir.

Elde edilen görüntüler, fotogrametri veya görüntü işleme yazılımları kullanılarak işlenir. Bu işlemde, görüntülerin bir araya getirilmesi, düzeltilmesi ve gerektiğinde yeniden boyutlandırılması gibi adımlar yer alır. Bu işlem adımı sonucunda her bir model için 4 farklı dosya türü elde edilir (.mtl, .obj, .png, .coi veya .map.txt). Oluşturulan ".mtl", ".obj", ".png" uzantılı üç dosyaya ait bir örnek Şekil 4'te gösterilmektedir.

.mtl (material) dosyası, bir 3D modelin malzeme bilgilerini içerir. Bu dosya, .obj dosyasıyla birlikte kullanılır ve modelin görünümünü belirler. Her model için renk, yansıma özellikleri, saydamlık gibi özellikler .mtl dosyasında tanımlanır. .obj dosyası, bir 3D modelin geometrik verilerini içerir. Bu dosya, modelin köşe noktalarını, yüzeylerini ve kenarlarını tanımlar. Modelin şeklini, boyutunu ve konumunu belirleyen temel verileri içerir. .png dosyası, 3D modelin yapısal ve yüzeysel özelliklerini veya renk haritasını içerir. Bu dosya, modelin yüzeylerine uygulanan dokuları veya renkleri

tanımlar. Görüntü dosya formatı olduğu için, modelin daha gerçekçi ve ayrıntılı görüntülenmesini sağlar. Bazı görüntü işleme yazılımları koordinat indekslerini içeren dosya türü olarak her bir model için ayrı .coi dosyası oluştururken, bazı yazılımlar tüm modellerin koordinat indekslerini içeren tek bir "map.txt" dosyası üretmektedir. Bu dosyalar, 3D modelin belirli bir koordinat sistemi içindeki konumunu tanımlar. Modelin yerini, dönüşünü ve ölçeğini belirleyen koordinat bilgilerini içerir. Bu dosya, modelin düzgün bir şekilde konumlandırılmasını sağlar ve farklı koordinat sistemlerine dönüştürülmesine yardımcı olabilir.

Şekil 5'de içeriği de mevcut olan her bir modelin id, lon(boylam), lat(enlem) ve height(yükselik) bilgileri map.txt dosyasında tutulmaktadır.



Şekil 4. Veri üretimi sonucu oluşan dosyalar

```
id,lon,lat,height
0,26.9979880968821,38.4062617971816,2.007253623751626
1,26.9981046125142,38.4062734825171,2.0082834314767046
10,26.9993928641882,38.4063412665514,2.0699845230703438
100,26.998111987294,38.40520891638,2.8811210697806096
1000,26.9969138152247,38.3989208747779,9.037241931428806
1001,26.9969720041263,38.3989607770041,9.03542460513354
1002,26.9900795069398,38.401570345845,4.829842385577305
1003,26.9900782134472,38.4016238369075,4.7908913825575175
1004,26.9901010485982,38.4015276735426,4.860915407910662
1005,26.9942958050036,38.4005338529412,5.746291279588901
```

Şekil 5. map.txt dosya içeriği

İkinci adım sınırlayıcı hacimlerin oluşturulma aşamasıdır. Bir 3D modelin sınırlarını tanımlayan sanal kutular, küreler veya diğer geometrik şekillerdir.

Tileset.json dosyası oluşturulurken, her bir 3D model için sınırlayıcı hacimleri belirlenir. Bu, modelin boyutunu, konumunu ve dönüşünü kapsayan bir kutu veya küre olabilir. Sınırlayıcı hacimler, modelin sınırlarını tanımladığından, verimli bir şekilde yüklenmesini ve görüntülenmesini sağlar.

Üçüncü adım geometrik hataların hesaplanması aşamasıdır. 3D model oluşturma sürecinde veya dönüştürme aşamasında geometrik hatalar oluşabilir. Bunlar, yüzeyler

arasındaki boşluklar, kenarların eşleşmemesi veya yüzeylerin iç içe geçmesi gibi sorunları içerebilir. Geometrik hatalar, modelin doğru bir şekilde görüntülenmesini engelleyebilir veya performansı olumsuz yönde etkileyebilir. Bu hatalar, modelin veri formatı dönüştürülürken veya sonraki bir aşamada tespit edilir ve giderilir.

Geometrik hataların tespit edilmesinin ardından, uygun düzeltme teknikleri kullanılarak iyileştirme işlemi yapılır. Bu iyileştirme işlemi, yüzeylerin düzeltilmesi, kenarların hizalanması veya boşlukların kapatılması gibi adımları içerebilir. İyileştirme işlemi, modelin daha doğru ve estetik bir şekilde görüntülenmesini sağlar. Bu işlem adımı sayesinde karo parçaları arasında daha kararlı bir geçiş sağlanmış olur.

Beşinci adım oluşan karoların içerisine uygun şekilde modellerin yerleştirmesi aşamasıdır. Tileset.json dosyasında, 3D modelin konumu, ölçeği ve dönüşü gibi yerleştirme bilgileri belirtilmelidir. Modelin düzgün bir şekilde yerleştirilmesi için koordinat bilgileri doğru bir şekilde belirlenmelidir. Bu, modelin dünya koordinat sistemi içinde doğru konumda ve yönde görüntülenmesini sağlar.

Son aşama sunum aşamasıdır. Tileset.json dosyası ve ilişkili 3D model, bir web uygulamasında veya 3D görüntüleme ortamında kullanılır. Model, kullanıcılar tarafından etkileşimli bir şekilde görüntülenebilir ve keşfedilebilir hale getirilir. Sunma aşaması, modelin doğru bir şekilde yüklendiğinden, performansının optimize edildiğinden ve kullanıcı deneyiminin iyileştirildiğinden emin olmayı içerir.

1.5. Tezin Amacı

Web ortamında çalışmanın gereksinimlerinden biri, büyük ölçekli verilerin daha küçük parçalara bölünerek yönetilmesidir. Bu nedenle, veriler daha küçük parçalara ayrılarak işlenebilir hale gelmiştir, bu parçalar "tile” olarak adlandırılmaktadır. 3D Tiles, web üzerinden transfer edilecek görüntülerin yalnızca görünür bölümlerini sağlayan hiyerarşik detay seviyelerine dayanmaktadır. Ancak, literatürdeki birçok çalışma, tile yöntemlerini kullanmamakta veya düzenli tile kullanmaktadır.

Bu tez çalışması, QuadTree, Octree, R-Tree ve Kd-Tree gibi veri yapıları kullanarak bölümlendirme performanslarını karşılaştırmayı ve ayrıca optimize bir bölümlendirme algoritması geliştirmeyi amaçlamaktadır. Ayrıca, 3D Tiles verilerinin oluşturulması için tamamen otomatik ve açık kaynaklı bir çözüm geliştirilmiştir.

Bu çalışma, büyük 3D veri kümelerinin etkili bir şekilde yönetilmesini ve web üzerinden daha verimli bir şekilde iletilmesini sağlayacak veri yapıları ve farklı algoritmaların incelenmesini hedeflemektedir. Bu, özellikle web tabanlı uygulamalarda daha iyi performans ve kullanıcı deneyimi sağlamak için önemli bir adım olacaktır.

1.6. Literatür Katkısı

Tez çalışması, 3D Tiles'ın geliştirilmesi, 3D jeo-uzamsal verilerin temsili ve görselleştirilmesine ilişkin literatüre önemli ölçüde katkıda bulunmuştur. Büyük ölçekli 3D coğrafi verilerin verimli akışına ve görselleştirilmesine yeni bir yaklaşım getirmiştir. Farklı yazılım bileşenleri ve sistemleri arasında birlikte çalışabilirliği mümkün kılarak 3D coğrafi verilerin temsilinin standartlaştırılmasına yardımcı olmuştur.

Genel olarak literatüre bakıldığında konunun kapsamlı bir çalışmasının eksik olduğu görülmektedir. Yapılan çalışmaların eksik olmasının bir sebebi açık kaynak bileşenlerinin olmamasıdır (Usta, 2021). Çalışmaların çoğunda ise düzenli bir bölümleme uygulanmış ve veri yoğunluğu gözetilerek hiyerarşik bir bölümleme yapılmamıştır

Yöntemlerin hepsi dağıtık bir biçimde bulunmakta her bir işlem adımı için farklı fonksiyonların kullanıcı müdahale ile çalıştırılması gerekmektedir (Anon(2019-a), 2019). Bu çalışma sonunda hem açık kaynaklı, hem de verinin 3D Tiles setlerine dönüşüm aşamasına kadar hiçbir müdahale gerektirmeden otomatik olarak oluşturan ve veri yoğunluğuna göre hiyerarşik bir bölümleme yapabilen uygulama geliştirilmiştir.

1.7. Tezin Organizasyonu

Yapılan tez çalışması 6 başlıkta ele alınmıştır. İlk başlık olarak 3D Tiles hakkında; 3D Tiles nedir, süreçleri nelerdir, neden ihtiyaç duyulmaktadır gibi genel bilgiler verilmiştir. Ardından 3D Tiles oluşturulmasından sunulmasına kadar olan sürecin zorluklarından yola çıkarak tez çalışmasının içeriği, amacı ve literatürdeki yeri gibi bilgilere değinilmiştir. İkinci bölümde 3D Tiles ve bölümleme algoritmaları hakkında literatür taramalarına ve web üzerinde 3DCM'leri tile yapısına kavuşturmak ve görselleştirmek için mevcut olan yazılım bileşenlerine yer verilmektedir. Üçüncü bölümde ise tez çalışmasında kullanılan araziler hakkında bilgiler verilmiş, kullanılan programlama dilleri ve yöntemlere değinilmiştir. Dördüncü bölümde kullanılan algoritmalar anlatılmış, probleme nasıl uygulandığı hakkında bilgiler verilmiştir. Beşinci bölümde ise örnek bir bölgede uygulanan algoritmalar için elde edilen ağaç yapıları, örtülü döşeme yapısı ve verinin uzay düzleminde 3D görüntülerine yer verilmiştir. Altıncı bölümde, tez çalışması kapsamında kullanılan tüm veri kümelerinden elde edilen sonuçlar incelenmiş, karşılaştırmalar yapılmış ve genel bir değerlendirme sunulmuştur.

2. KAYNAK ARAŞTIRMASI

Bu bölümde 3D Tiles için yapılan akademik çalışmalar, bölümleme algoritmaları için yapılan kaynak araştırmaları ve son olarak web üzerinde 3DCM'leri tile yapısına kavuşturmak ve görselleştirmek için mevcut olan altı adet yazılım bileşeni anlatılmaktadır.

2.1. 3D Tiles Kaynak Araştırması

Gesquiere ve Manin'in yaptıkları çalışmada (Gesquière ve Manin, 2012), 3 boyutlu şehir ve manzara model verilerini web üzerinde görselleştirmek için bir istemci sunucu mimarisi geliştirmiştir. Bu çalışmada hiyerarşi olmadan düzenli tile kullanılmıştır. Bu yöntemlerin en büyük dezavantajı, 3 boyutlu şehir modellerinde binaların dağılımının heterojen olmasıdır. Bu nedenle bazı alanlarda diğerlerinden çok daha fazla şehir nesnesi vardır ve düzenli tile yaklaşımı ile hiyerarşik alt bölmelere ayrılmamıştır. Şehirdeki nesnelerin yoğunluğu dikkate alınmamıştır ve heterojen boyutta tilesetler oluşturulmuştur. Verilerde hiyerarşi olmaması görselleştirme hattının filtreleme ve haritalama adımlarını optimize etmek için sahne grafiklerinden yararlanmayı engellemektedir.

Benner ve arkadaşlarının 2013 yılında yaptıkları çalışmada (Benner ve dig., 2013), 3D şehir modelleri için mevcut CityGML standardının ayrıntı düzeyi (LOD) kavramı tartışılmış ve alternatif bir kavram geliştirilmiştir. Mevcut CityGML LOD kavramında tanımlanan eksiklikler ve bu eksiklikleri gidermeye yönelik alternatif bir kavram olan GLOD (Genelleştirilmiş LOD) ve SLOD (Spesifik LOD) ayrımı önerilmiştir. Bu ayrım, modelleme sürecinde daha etkili ve verimli bir yaklaşım sunmaktadır.

Von Schwerin ve arkadaşları tarafından yapılan çalışmada (Von Schwerin ve dig., 2013), antik mimari ve peyzajların analizine odaklanan bir 3D WebGIS aracı geliştirilmiştir. Bu araç, mekansal olarak referanslanmış verilerle entegre edilen 3D modelleri kullanarak araştırmacıların antik yapıları ve çevresini daha detaylı incelemelerini sağlamaktadır. Geliştirilen 3D WebGIS aracı, araştırmacılara 3D modeller üzerinde mekansal veriyle bağlantılı sorgulamalar yapma imkanı sunarak, antik mimari ve peyzajların gerçek zamanlı olarak sanal gerçeklik ortamında analiz edilmesine olanak tanır. Açık kaynak yazılım seçenekleri dikkate alınarak gelecekteki 3D teknolojilerinin sürdürülebilirliği sağlanmıştır. Ayrıca, bu araç, araştırmacıların 3D modelleri ve mekansal verileri entegre bir şekilde kullanarak daha detaylı ve kapsamlı analizler yapmalarına olanak tanımıştır.

Gaillard ve arkadaşları tarafından yapılan çalışmada (Gaillard ve dig., 2015), 3 boyutlu kent ve manzara modellerini web üzerinde görselleştirmek için bir çerçeve geliştirilmiştir. CityGML dosyaları JSON formatına dönüştürülmüş ve veri setleri sabit boyutlu normal bölmelere ayrıştırılmıştır. Aşamalı görselleştirme süreci, "three.js" kütüphanesi kullanılarak gerçekleştirilmiştir. Önerilen oluşturma stratejisi sadece normal döşemelerle çalışmayı hedefler ve bitişik döşemelerin zorluklarına odaklanır. Bu çalışma, çok sayıda bitişik döşemenin mevcut olabileceği durumlarda hiyerarşik veri yapılarının kullanımının önemine işaret etmektedir.

Chaturvedi ve arkadaşları yaptıkları çalışmada (Chaturvedi ve dig., 2015), çok büyük anlamsal 3 boyutlu şehir modellerinin işlenmesi, görselleştirilmesi ve analizi için web tabanlı bir 3 boyutlu istemci geliştirmişlerdir. Geliştirilen uygulamanın tarayıcılar arası, platformlar arası ve eklentiden bağımsız olmasını sağlayan HTML5 VE WebGL kullanılmıştır. Görselleştirme amacıyla ise WebGL tabanlı Cesium sanal küresinde çalışılmıştır. Ancak çalışmada 3D standartlarının hiçbiri uygulanmamış ve düzenli dikdörtgen bölümleme gerçekleştirilmiştir.

Chen ve arkadaşları tarafından 2016 yılında yayınlanan çalışmada (Chen ve dig., 2016), karmaşık 3 boyutlu modellerin çevrimiçi iletimi ve gerçek zamanlı olarak oluşturulması sırasında veri iletiminin ve ağ yeniden yapılandırmasının verimliliğini artırmak için yeni bir aşamalı ağ yapısı önerilmiştir. Aşamalı görselleştirmenin ilk aşamasında, temel veri ve köşe kümeleme sadeleştirmesi tarafından oluşturulan temel dizin, temel işleme için istemciye iletilir. Ardından, bakış açısı simülasyon nesnesine yaklaştıkça artımlı veriler ve daha yüksek seviyelerde karşılık gelen indexler iletilir. Çok ölçekli artırımlı veri organizasyonu, karşılık gelen ağ ayrıntıları düzeyini ayrı ayrı ileterek ve yeniden yapılandırarak Web 3D simülasyon sisteminin performansına ve verimliliğine fayda sağladığı gözlemlenmiştir.

Biljecki ve arkadaşları çalışmalarında (Biljecki ve dig., 2016), 3D bina modelleri için belirli uygulamalara uygun bir LOD modelleme yaklaşımını önermektedirler. Önerilen yaklaşım, bina modellerini bağlam-bilinçli ve farklı özelliklere sahip LOD seviyeleriyle sunarak, belirli uygulamalara daha iyi uyum sağlamayı amaçlamaktadır. Araştırma, beş temel değerlendirme kriteri ortaya koymaktadır. Bu kriterler, modellerinin kullanım amacına göre özelleştirilmiş detay seviyeleri sunarak, ilgili uygulamalar için daha fazla verimlilik sağlamaktadır.

Kim ve arkadaşlarının yaptığı çalışmada (Kim ve dig., 2017), Kore tipi mekânsal bilgi açık platformu olan V-world uygulamasından bahsedilmiştir. Uygulama, ülkenin 2

boyutlu ve 3 boyutlu harita ve idari bilgilerinin kolayca kullanılabilmesi için çeşitli hizmetler sunmaktadır. Çoklu model istekleri ve çizim çağrılarının neden olduğu 3 boyutlu harita hizmetinin performans düşüşünü azaltmak için birden çok bina modelini tek bir tileda birleştiren bir 3 boyutlu tile modeli uygulayarak model dosyası için talep ve çizim çağrılarının sayısı azaltılmaya çalışılmıştır. Ek olarak, modelin geri alınma süresini kısaltarak model dosyasının yüklenmesi için gereken süreyi azaltmak için QuadTree algoritması uygulanmıştır.

Chen ve arkadaşlarının 2018 yılında yayınladığı çalışmada (Chen vedig., 2018), İnşaat Endüstrisi Temel Sınıfları (IFC) veri formatındaki Bina Bilgi modellerini (BIM), açık kaynaklı bir 3D WebGIS platformunda mevcut jeo-uzamsal verilerle entegre etmek için yeni bir uygulama geliştirmişlerdir. Geliştirilen uygulamanın GLTF'den b3dm'e dönüştürülme aşamasında, uygulamanın kolaylığı için aynı IFC tipindeki bileşenleri bir b3dm dosyasına hiyerarşi olmaksızın sardığı belirtilmiştir. İkinci dezavantajı ise işleme performansını iyileştirmek için kritik olan Hiyerarşik Ayrıntı Düzeyini (HLOD) dikkate almamasıdır.

Song ve Li'nin 2018 yılındaki konferans bildirisinde (Song ve Li, 2018), terminal bellek boyutuna ve ekran alanı hatasına dayalı olarak bir 3D Tile dinamik yükleme ve programlama stratejisi önerilmiştir. Bu strateji, görsel etkileşim sürecinde veri yükünü gerçek zamanlı olarak izler. Yüklenen veriler önceden ayarlanmış eşiği aştığında, bazı yüksek çözünürlüklü bölmeler kaldırılır ve yerine biraz daha düşük çözünürlüklü bölmeler kullanılır. Bu stratejinin benimsenmesinin, yüklenen veri miktarının kararlığını ve görsel etkileşim sırasında daha iyi görsel efektleri sağlayabileceğini ve tarayıcının her zaman normal şekilde çalışabileceğini kanıtlamıştır.

Murshed ve arkadaşlarının yaptığı çalışmada (Murshed ve dig., 2018), açık kaynaklı Cesium sanal küresini temel alan, hazır 3 boyutlu veri kümelerini ve sanal küre yazılımındaki gelişmeleri kullanarak akıllı şehir uygulamalarının entegre edilmesinin potansiyelini ortaya koymaktadır. Çalışmalarında elde ettikleri sonuçlarda, 3D Tiles'da dönüştürme aracı hala geliştirme aşamasında ve henüz kamuya açık olmadığı, dönüştürme için orijinal verilerin Cesium ekibiyle paylaşılması gerektiği gözlemlenmiştir.

Mete ve arkadaşları çalışmalarında (Mete ve dig., 2018), WebGL'yi destekleyen açık kaynaklı bir javascript paketi olan CesiumJS'nin web üzerinde 3 boyutlu veri görselleştirmesi için kullanımı tartışılmaktadır. WebGL ve Cesium gibi Javascript tabanlı haritalama kütüphaneleri sayesinde üç boyutlu coğrafi verilerin web üzerinde

görselleştirilmesinin önemi vurgulanmaktadır. Bu çalışma, 3 boyutlu ve 4 boyutlu verileri de içeren çeşitli veri formatlarını görselleştirmek için Cesium'un kullanımını göstermekte ve onu web tabanlı coğrafi uygulamalar için değerli bir araç haline getirmektedir.

She ve arkadaşları tarafından yapılan çalışmada (She ve dig., 2019), model karmaşıklığını azaltmak ve işleme verimliliğini artırmak için model görünümünü koruyan yeni bir basitleştirme yöntemi önerilmiştir. Doku süreksizlikleri, her köşenin doku koordinatları ile komşu üçgenler arasındaki eşleme ilişkisini kaydeden yeni bir veri modelini geliştirerek ele almışlardır. Bina modelinin yüzey ağı daha sonra topoloji ve görünüm tarafından yönlendirilen bölgelerle bölünür. Son olarak, hem geometrik hem de doku hatalarını dikkate alan gelişmiş bir hata ölçüsü türetmek için ağ bölümlendirme bilgisi kullanmışlardır ve doku koordinatları her basitleştirme işleminden sonra ayarlanır.

Büyükdemircioğlu ve Kocaman'ın 2020 yılındaki çalışmaları (Buyukdemircioğlu & Kocaman, 2020), şehir modellerinde nesne çeşitliliğinin ve detay seviyesinin artmasına rağmen, otomatik yeniden yapılandırma, dinamik güncelleme ve modellerin yüksek performanslı web tabanlı görselleştirmesi gibi konuların hala zorlu olduğunu ortaya koymaktadır. Bu çalışma, mevcut yapıları gelecekteki şehir modelleriyle entegre eden bir çerçeve sunarak, yeniden yapılandırma, görselleştirme ve performans ayarlama gibi zorluklara değinmektedir. Bu, gelecekteki kentsel planlama ve modelleme çalışmaları için önemli bir katkı sağlamaktadır.

Hu ve arkadaşlarının yaptıkları çalışmada (Hu ve dig., 2020), akıllı şehir yönetiminde önemli bir rol oynayan yer altı boru ağlarının ayrıntılı 3 boyutlu görüntüsünün oluşturulması ve bu boru ağlarının yönetiminin zorlukları ele alınmıştır. Çalışmada, çok sayıda boru noktasının ve bölümünün yönetilmesinin zorluğu sebebiyle, 3 boyutlu boru ağı modelleme ve organizasyon yönetimi üzerine derinlemesine bir araştırma gerçekleştirilmiştir. Oluşturulan model gevşek dörtlü veri organizasyonu aracılığıyla 3 boyutlu boru ağı için heterojen bir veri yapısı oluşturulmuştur. Önerilen veri yapısının, Cesium tarafından benimsenen 3 boyutlu tilelar için uygun olduğu ve boru ağı modelleri için gerekli görsel etki ve verimliliği sağladığı gözlemlenmiştir.

Xu ve arkadaşlarının yayınladıkları çalışmada (Xu ve dig., 2020), Bina Bilgi Modellemesi (BIM) ve Coğrafi Bilgi Sistemi (GIS) arasındaki veri entegrasyonu ve bilgi etkileşimi için bir uygulama planı sunmuşlardır. BIM modeli, İnşaat Endüstrisi Temel Sınıfları'ndan (IFC) biri olarak kabul edilen veri formatından 3D Tiles formatına

dönüştürülmüştür. Dönüştürme sırasında koordinat dönüştürme, veri eşleme, uzamsal dizin ve Ayrıntılar Düzeyi (LOD) hiyerarşik bölümü tamamlanır ancak gerçekleştirilen çözüm otomatik bir işlem akışı sağlamamakta, farklı yazılım bileşenleri ve farklı işlem adımları kullanılmaktadır. Gerçekleştirilen çözüm web tabanlı değildir.

Tung ve arkadaşları çalışmalarında (Tung ve dig., 2020), etrafında çeşitli yüzeyler bulunan bir alanın şeklini hassas bir şekilde tahmin etmek amacıyla bir 3D modelleme sistemi geliştirilmiştir. Bu sistem, farklı yönlere bakılarak çekilen görüntüler yardımıyla uzayın şeklini tahmin eder ve bu görüntülerden elde edilen kısmi modelleri bir araya getirerek genel bir model oluşturur. Çoklu görüntüler kullanarak uzayın şeklini doğru bir şekilde tahmin eden bu sistem, yakalanan görüntülere dayalı olarak kısmi ve genel modellerin oluşturulması ile bunu başarır.

Jaillot'ın geliştirdiği uygulamada (Jaillot, 2020), web üzerinde 3 boyutlu şehir modellerini sunmak için standart formatların genel bir kavramsal modelini tasarlar, şehirlerin zamansal boyutunu bu genel kavramsal modellere resmileştirir ve entegre eder. Ayrıca 3D Tiles standardındaki kavramsal modeli mantıksal ve teknik spesifikasyon düzeylerinde belirtir. 3D Tiles standardını genişleterek 3DCM'ler, py3dtiles açık kaynaklı yazılım bileşeni kullanılarak döşenmiştir. Bu bileşen, CityGML formatındaki 3DCM'leri 3D tilelarda dönüştürmek için kullanılan bir python kütüphanesidir. py3dtiles'ın dezavantajı ise web tabanlı bir bileşen olmamasıdır.

Vu ve arkadaşlarının 2020 yılında yaptıkları çalışmada (Vun ve dig., 2020), verilerin dağılımına dayalı olarak uygun mekansal bölümleme tekniğini seçmek için derin öğrenme kullanımı araştırılmıştır. Mevcut yöntemler, verilerin dağılımına dayalı olarak otomatik olarak bölümleme tekniği seçimini gerçekleştirememektedir. Bu nedenle, bu çalışmada derin öğrenme modelleri kullanılarak, veri kümesi özetlerine dayanarak en iyi bölümleme tekniğini tahmin eden bir model önerilmektedir. Önerilen derin öğrenme modeli, verilerin dağılımına dayalı olarak en uygun bölümleme tekniğini seçmek için tasarlanmıştır. Modelin performansı, deneysel değerlendirmelerle test edilmiş ve modelin doğruluk açısından mevcut temel yöntemleri geride bıraktığı gösterilmiştir. Bu bulgular, derin öğrenmenin büyük veriler için mekansal bölümleme tekniklerinin seçimi konusunda önemli bir potansiyele sahip olduğunu ortaya koymaktadır.

Zhan ve arkadaşları tarafından yapılan çalışmada (Zhan ve dig., 2021), web tabanlı 3D model görüntüleyicide bina bilgi modellerini sorunsuz görselleştirmek için 3D Tiles'a dayalı oldukça verimli bir yöntem önermişlerdir. Bu yöntem karoların

boyutunu açıklayan R-Tree algoritmasına dayalı 3D modeller için bir tile yöntemi kullanarak, bina bilgi modellerinin orijinalini bozmadan bölmek için kullanılmıştır. Ardından bina bilgi modelini katmanlamak için kullanılan bir ayrıntı düzeyi yöntemi olan "Mask Filter" tanıtılmıştır. Önerilen yöntemin geleneksel yöntemlerle karşılaştırıldığında yüksek verimlilik elde ettiği gözlemlenmiştir. Bununla birlikte, araştırmanın belirli yönleri daha fazla optimize gerektirdiği belirlenmiş, bilgisayar üzerinde sadece deneysel bir platformda düşünülmüş ve mobil cihazlar için bu görselleştirme dikkate alınmamıştır.

Usta'nın yaptığı doktora tez çalışmasında (Usta, 2021), 3D Tiles spesifikasyonuna uygun olarak 3D bölümleme yapan web tabanlı açık kaynak kodlu yazılım bileşeni geliştirilmiştir. Bölümleme için R-Tree ve Adaptive QuadTree hiyerarşik veri yapıları kullanılmış ve üretilen 3D bölümler Cesium.js açık kaynak kodlu javascript kütüphanesi kullanılarak görselleştirilmiştir. Adaptive QuadTree ve R-Tree veri yapıları 3D bölümleme açısından irdelenmiş, oluşturulma süreleri, veri güncelleme süreleri ve konumsal sorgu performansları karşılaştırılmıştır.

Xiang ve arkadaşları yaptıkları çalışmada (Xiang ve dig., 2021), kentsel 3D sahnelerin görselleştirilmesinde karşılaşılan bilgisayar performansı sınırlamalarını ve veri organizasyon sorunlarını çözmeyi hedeflemişlerdir. Özellikle, 3D modellerin rastgele bir klasörde depolanmasının veri işleme verimliliğini düşürdüğü, depolama fazlalığına ve anlamsal parçalanmaya yol açtığı vurgulanmaktadır. Bu sorunları çözmek için, uyarlanabilir QuadTree ve sahne grafiği tabanlı bir 3D sahne yönetim yöntemi (AQT-SG) önerilmiştir. Deneysel sonuçlar, önerilen yöntemin global ve yerel detayları daha iyi koruyabildiğini, farklı ölçekler arasında geçiş yaparken kararlılığı sağladığını ve genel görsel render etkisini iyileştirdiğini göstermektedir. Ayrıca, bellek fazlalığı karşılaştırma deneyinde bu yöntemin üstünlüğü kanıtlanmıştır. Ancak, zaman kısıtlamaları nedeniyle, önerilen tasarımda bazı sorunlar hala mevcuttur. Örneğin, bu tasarım yalnızca farklı ölçeklerde 3D şehir sahnelerinin indeksleme problemini ele almaktadır. Belirli bir ölçekteki hiyerarşik LOD'un dinamik planlanması konusunda derinlemesine araştırma yapılmamıştır.

Yang ve arkadaşlarının 2021 yılında yaptıkları çalışmada (Yang ve dig., 2021), 3D GIS teknolojisinde veri işleme ve görselleştirme süreçlerinin optimize edilmesi üzerinde durulmaktadır. Geliştirilen algoritmalar, veri işleme ve görselleştirmeyi iyileştirmekte ve daha iyi görsel etkiler sağlamak amacıyla optimize edilmektedir. Geliştirilmiş alan odaklı işleme yöntemi ve nesne yönelimli optimizasyon algoritması,

3D coğrafi bilginin daha etkin işlenmesini sağlamaktadır. Bu çalışma, coğrafi bilgi sistemlerinde 3D görselleştirme ve veri işleme süreçlerinin geliştirilmesine önemli katkılar sunmaktadır.

Khayyal ve arkadaşları tarafından yapılan çalışmada (Khayyal ve dig., 2022), ücretsiz kaynaklar kullanılarak GIS verilerine dayalı 3D şehir modellemesi incelenmektedir. Araştırma, 3D şehir modellerinin oluşturulması ve mekansal analizine odaklanmakta olup, farklı veri kaynaklarının doğruluğunu ve uygulama potansiyelini değerlendirmektedir. GIS teknolojisi ve ücretsiz veriler kullanılarak prosedürel modelleme teknikleri ile 3D şehir modeli oluşturulmuştur ve modellere dayalı mekansal analizler gerçekleştirilmiştir. Bu çalışma, 3D şehir modellemesi alanında önemli bir katkı sunmaktadır.

Hillmann ve arkadaşlarının yaptığı çalışmada (Hillmann ve dig., 2022), Octree bölümleme kullanılarak Cesium 3D Tiles biçiminde otomatik olarak bir ayrıntı hiyerarşisi düzeyi oluşturulmuştur. Bu yöntemin avantajı, her bir detay seviyesi aşaması için oluşturulan bölmelerin, işleme sırasında farklı detay seviyeleri arasındaki görsel farkları en aza indirecek şekilde tutarlı bir şekilde hesaplanmasını sağlayan bir yöntemdir.

Murtiyoso ve arkadaşlarının yayınladıkları çalışmada (Murtiyoso ve dig., 2024), mevcut 3D teknolojilerinin ormancılıkta nasıl kullanıldığına dair geniş bir inceleme sunulmaktadır. Bu bağlamda, sanal ormanların gereksinimleri, görselleştirme ve uygulamaları özetlenmiştir. Ormancılıkta 3D teknolojilerinin trendleri, zorlukları ve potansiyel etkileşimleri tartışılmaktadır. Dijital 3D verilerin ormancılıkta giderek artan bir şekilde kullanıldığı ve büyük bir potansiyel gösterdiği vurgulanmaktadır. Ayrıca, sanal ormanlarda yeni yöntemlerin daha fazla dikkat çekmesi gerektiği belirtilmektedir.

Yang ve arkadaşları yaptıkları çalışmada (Yang ve dig., 2024), büyük ölçekli üç boyutlu şehir modellerinde verilerin organizasyon verimliliğini artırmayı ve LOD oluşturmanın neden olduğu bina parçalanma sorununu çözmeyi amaçlamışlardır. Binaların mekansal dağılımını dikkate alan bir veri organizasyon yöntemi önermişlerdir. Bu yöntem, sahnedeki binaları makro, mezo ve mikro ölçeklerde sınıflandırmakta ve bu sınıflamayı R-Tree kullanarak kaydetmektedir. Daha sonra, şehir modelinin veri dizinini oluşturmak için uyarlanabilir bir QuadTree kullanılmaktadır. R-Tree, ön işleme aşamasında statik olarak oluşturulurken, uyarlanabilir QuadTree sahne oluşturma sırasında dinamik olarak oluşturulmaktadır. Önerilen yöntemin, kullanıcının ilgi alanının bütünlüğünü sağlamakla kalmayıp, aynı zamanda 3D sahne inşasının

verimliliğini de artırdığını göstermektedir. Bu nedenle, bu yöntemin büyük ölçekli kentsel sahnelerde veri organizasyonuna uygulanmasının evrensel bir uygulanabilirliği olduğu sonucuna varılmıştır.

2.2. Bölümleme Algoritmaları Kaynak Araştırması

Samet tarafından 2006 yılında yayınlanan kitapta (Samet, 2006), QuadTree tabanlı çok çözünürlüklü yöntemlerin mekansal ve zamansal alanların dinamik tahmini için nasıl kullanılabileceği inceleniyor. QuadTree, özellikle resim işleme ve coğrafi bilgi sistemleri gibi alanlarda çoklu çözünürlüklü verileri işlemek için popüler bir veri yapısıdır. Makalede, QuadTree'nin dinamik tahmindeki uygulamalarını ve avantajlarını ele alarak çok çeşitli alanlarda kullanılabilirliğini vurgulanıyor.

Haverkort ve arkadaşları 2016 yılındaki konferans bildirisinde (Haverkort ve dig., 2016), QuadTree'nin, noktalar, çizgiler ve görüntüler gibi geometrik nesneler için kullanılan hiyerarşik bir veri yapısı olduğuna değinmişlerdir. Bu hiyerarşik veri yapısı, belirli bir durdurma kuralına kadar alanı tekrar tekrar dört alt bölüme ayırır. QuadTree, geometrik nesneler için bir veri yapıları sınıfını tanımlamakta olup, bir bölgeyi daha fazla alt bölgeye bölmenin gerekli olup olmadığını belirleyen bir durdurma kuralı kullanarak alanı hiyerarşik olarak bölmektedir. Ayrıca, geometrik nesneler için QuadTree veri yapıları ve lineer QuadTree varyantı tanıtılmıştır.

Tong yaptığı çalışmada (Tong, 2002), 3D coğrafi bilgi sistemlerinde etkili ve kompakt bir raster veri yapısı olan Octree incelenmiştir. Octree, 3D nesneleri hiyerarşik olarak temsil eden ve bellek tasarrufu sağlayan sıkıştırma tekniklerine yardımcı olan bir 8'li ağaç yapısı kullanmaktadır. Bu çalışmada, Octree tabanlı harita birleştirme için geçerli stratejiler gösterilmektedir. Bellek verimli 3D ortam haritalarını birleştirmek için uygulanabilir bir yöntem sunulmaktadır. Çalışma, Octree'nin karmaşık coğrafi mekansal bilgiyi temsil etmek için ideal bir veri yapısı olduğunu ve aynı özelliğe sahip sıralanmış octantları sıkıştırmanın etkili bir yolunu açıklamaktadır. Sayma sıralama algoritmasını kullanarak yapılan sıkıştırma tekniği, hızlı sıralamadan daha hızlıdır. Octree kodu için bellek saklama miktarı, diğer yöntemlere kıyasla oldukça düşüktür.

Zhou ve arkadaşları çalışmalarında (Zhou ve dig., 2012), bir grafik işlem biriminde (GPU) tam bir Octree veri yapısı oluşturma sistem ve yöntemi sunulmaktadır. Ancak, yöntem büyük ölçekli uygulamalar için uygun değildir. GPU üzerinde tam bir Octree veri yapısı oluşturulması, nokta bulutunun düğüm, köşe, kenar ve yüz dizilerini oluşturmak için kullanılmıştır.

Nguyễn ve arkadaşlarının 2022 yılındaki çalışmalarında (Nguyễn ve dig., 2022), çok seviyeli sınıflandırma ve sınıflandırma vektörlerinin eğitimi için kullanılan Kd-Tree yapısının, görüntü modelleri üzerinde diğer yöntemlere kıyasla yüksek performans sergilediği ortaya koyulmaktadır. Çalışmada önerilen yöntem, çeşitli görüntü veri setleri üzerinde etkili bir şekilde çalışarak, farklı alanlarda görüntü sınıflandırmasında üstün bir performans göstermektedir. Aynı veri seti üzerinde diğer yöntemlere göre daha yüksek performans sergileyen bu yöntem, Kd-Tree yapısını kullanarak çok katmanlı sınıflandırma için görüntü sınıflandırması gerçekleştirmektedir. Önerilen algoritmalar ve eğitim sınıflandırıcıları, çeşitli veri setleri üzerindeki deneysel sonuçlar göz önüne alındığında, Kd-Tree yapısına dayalı görüntü sınıflandırma yöntemi başarılı bir şekilde uygulanmıştır. Bu bulgular, Kd-Tree tabanlı yaklaşımın, görüntü sınıflandırmasında etkin ve verimli bir çözüm sunduğunu doğrulamaktadır.

Lu ve arkadaşlarının 2023 yılındaki konferans bildirisinde (Lu ve dig, 2023), nokta bulutu verileri Kd-Tree ile organize edilip yönetilmiş ve yapılan deneyler, Kd-Tree'nin verileri etkin bir şekilde yönetebildiğini ve ilgili yarıçap arama ve en yakın komşu arama işlemlerini yüksek verimlilikle gerçekleştirebildiğini kanıtlamıştır. İlgili yarıçap arama ve en yakın komşu arama işlemlerinde yüksek verimlilik sağlamaktadır. 3D lazer tarama teknolojisi, nokta bulutu verilerinin düzenlenmesi için kullanılmıştır. Bu çalışma, Kd-Tree ilgili yarıçap arama ve en yakın komşu arama işlemlerinde yüksek verimliliğini ve nokta bulutu verilerinin etkili yönetimini vurgulamaktadır.

Yang yaptığı çalışmada (Yang, 2007), R-Tree, Coğrafi Bilgi Sistemlerinde (GIS) verileri etkili bir şekilde düzenlemek için kullanılan dinamik bir mekansal indeksleme yapısı olduğunu açıklamışlardır. Bu yapı, bir nesnenin dikdörtgeni tamamen başka bir nesnenin içine alındığında ortaya çıkan durumlar için yeni bir varyasyon önerir. R-Tree'nin ekleme, bölme ve ayarlama algoritmaları, mekansal nesnelerin kapsayan dikdörtgenleri ile oluşturulan dinamik bir indeksleme yapısı olan R-Tree'nin ayrıntılı bir şekilde açıklanmıştır. Ekleme, bölme ve ayarlama algoritmaları detaylı bir şekilde açıklanmış ve test edilmiştir.

Yan tarafından yapılan çalışmaya göre (Yan, 2010), R-Tree algoritması, 3D GIS'de yaygın olarak kullanılan bir mekansal indeksleme yöntemi olduğunu açıklamışlardır. Bu yöntem, kardeş düğümler arasındaki örtüşmeyi azaltarak sorgu hızını artırmayı amaçlar. Bunun yanı sıra, R-Tree algoritmasının, 3D kapsama ve örtüşme hacimleri gibi kriterleri de dikkate aldığı belirtilmiştir.

2.3. 3D City Model İçin Kullanılan Mevcut Yazılım Bileşenleri

Web üzerinde 3DCM'lere tile yapısına kavuşturmak ve görselleştirmek için yazılım bileşenlerine baktığımızda açık kaynaklı yazılım bileşenleri olarak web tabanlı bir çözüm yoktur (Usta, 2021). Bu bölümde mevcut yazlım bileşenleri incelenip, farklılıkları ve eksiklikleri ortaya çıkarılarak, neden bu programlar dışında bir yazılım bileşenine ihtiyaç duyulduğu açıklanacaktır.

2.3.1. Obj23dtiles

3D model dosyasının obj formatından 3D Tiles formatına dönüştürülme süreci, genellikle bir araç veya yazılım kullanılarak gerçekleştirilir. Obj formatı, yaygın olarak kullanılan bir 3D model dosya biçimidir. Dönüştürme işlemi, obj dosyasını okuyabilen ve onu 3D Tiles formatına aktarabilen bir araç veya yazılım kullanmayı içerir. Bu tür dönüştürmeyi destekleyen araçlar arasında CesiumIon, FME ve Babylon.js gibi örnekler bulunmaktadır.

obj23dtiles.js adlı bir açık kaynak Node.js modülü, 3D verileri 3D Tiles'a dönüştürmek için kullanılabilir. Bu modül, Cesium tarafından geliştirilen başka bir açık kaynaklı Node.js modülü olan "obj2gltf" temel alınarak oluşturulmuştur. Bu Node.js modülü, bir komut satırı aracı ve bir modül olarak obj model dosyalarını obj2gltf tabanlı olarak 3D Tiles'a dönüştürür. Ancak, "obj23dtiles" modülünün bir dezavantajı, verilerden bir hiyerarşi oluşturmaması ve bir tile yöntemi uygulamamasıdır. Yani, yalnızca obj formatındaki tüm 3 boyutlu şehir modellerini b3dm formatına dönüştürür.

Obj dosyasının 3D Tiles'a dönüştürülmesi sonrasında, bu veriler WebGL tabanlı bir sanal dünya ve harita motoru olan CesiumJS veya 3D içerik için bulut tabanlı bir platform olan CesiumIon gibi 3D Tiles formatını destekleyen uygulamalarda kullanılarak yönetimi ve dağıtımı gerçekleştirilir (Usta, 2021). Bu uygulamalar, 3D verilerin interaktif olarak görüntülenmesine ve etkileşimli kullanıcı deneyimlerinin oluşturulmasına olanak tanır.

2.3.2. Citygml-to-3dtiles

Citygml-to-3dtiles, Citygml verilerini 3D Tiles formatına dönüştürmek amacıyla kullanılabilen açık kaynaklı bir yazılım aracıdır. Bu araç, Cesium tarafından geliştirilmiş olup 3D coğrafi uygulamalarda kullanılmak üzere Citygml dosyalarından 3D Tiles veri kümeleri oluşturmak için tasarlanmıştır (Cesium).

Citygml-to-3dtiles, Citygml verilerini okumak ve GLTF modellerine dönüştürmek için Citygml4j kütüphanesini kullanır. Daha sonra, GLTF modellerini 3D Tiles formatına dönüştürmek için GLTF pipeline aracını entegre eder. Ortaya çıkan tile

seti, CesiumJS veya diğer 3D Tiles görüntüleyicileri gibi araçlar kullanılarak görselleştirilebilir ve çevrimiçi olarak yayınlanabilir.

Citygml-to-3dtiles'in çıktısı, her ayrıntı düzeyi için tile setleri içeren bir dizin ve tileset.json formatındaki 3D Tiles veri kümesini içerir (Usta, 2021). Bu araç, çeşitli Citygml profillerini destekler, örneğin, binalar, arazi kullanımı, ulaşım altyapısı ve su kütleleri. Ayrıca, özelleştirilebilir ve diğer Citygml profillerini desteklemek üzere uyarlanabilir. Büyük veri kümelerini işleyebilmek adına 3D Tiles'ı döşeyip yayınlayabilir (Yücel, 2009).

Genel olarak, Citygml-to-3DTiles, Citygml verilerini kolayca görselleştirilebilen ve web üzerinde yayınlanabilen bir formata dönüştürmek için kullanışlı bir araçtır. Şehir planlama, mimari ve jeo-uzamsal analiz gibi birçok uygulama için faydalı olabilir, çünkü Citygml verilerinden 3D Tiles veri kümeleri oluşturma sürecini basitleştirir.

2.3.3. Py3dtiles

Py3dtiles, web üzerinde 3D jeo-uzamsal verileri depolamak ve yayınlamak için popüler bir format olan 3D Tiles verilerini oluşturmak ve işlemek için tasarlanmış bir Python kitaplığıdır (Usta, 2021). Kullanıcılara Python kodu kullanarak 3D Tiles setlerini oluşturma ve düzenleme imkanı tanır, bu da geliştiriciler ve araştırmacılar için 3D jeo-uzamsal verilerle çalışmayı kolaylaştırır. Py3dtiles, çeşitli işlemler için araçlar sunarak 3D Tiles verileriyle çalışmayı destekler:

2.3.4. Virtual City Publisher

Virtual City Publisher, kullanıcılara şehirlerin ve diğer coğrafi alanların 3D sanal modellerini oluşturma ve yayınlama imkanı sunan bir yazılım aracıdır. Bu araç, binalar, yollar, arazi, bitki örtüsü ve diğer detaylarla tamamlanmış ayrıntılı 3D şehir ve manzara modelleri oluşturma konusunda kullanıcılara olanak tanır. Oluşturulan sanal şehirler, şehir planlama, mimari görselleştirme, oyun ve simülasyon gibi çeşitli amaçlar için kullanılabilmektedir (Klimke, 2019).

2.3.5.FeatureManipulation Engine (FME)

FME veri entegrasyonu platformudur ve kullanıcılara veri iş akışlarını bağlamaları, dönüştürmeleri ve otomatikleştirmeleri için olanak tanır (Usta, 2021). FME, görsel bir kullanıcı arayüzü üzerinden veri iş akışlarını programlamadan veya komut dosyası oluşturmadan tasarlama ve yapılandırma imkanı sunar.

2.3.6. CesiumIon

CesiumIon, 3D verileri bölümlemek, optimize etmek ve barındırmak için çevrimiçi bir platform olarak öne çıkar. Modern bir bulut mimarisi üzerine inşa edilen

uygulama, büyük veri kümelerini CesiumJS gibi web uygulamalarına verimli bir şekilde aktarılabilen herhangi bir cihazda paylaşılabilir. CesiumIon'un sunduğu varlık türleri arasında fotogrametri veya Lidar'dan türetilmiş ağ bilgileri, BIM, CAD veya diğer 3D modeller, nokta bulutları, 3D binalar, uydu veya drone görüntüleri ve arazi gibi çeşitli 3D jeo-uzamsal veri türleri bulunmaktadır (Cesium).

Yazılım bileşenlerinin yetenekleri Tablo 1'de özetlenmiştir.

Tablo 1. Yazılım bileşenlerinin özeti (Usta, 2021)

Yazılım İsmi	Masaüstü / Web Tabanlı	Kaynak Türü	Hiyerarşik Tile
Obj23dtiles	Masaüstü	Açık Kaynak	Yok
Citygmlto3dtiles	Masaüstü	Açık Kaynak	Yok
Py3dtiles	Masaüstü	Açık Kaynak	Var
Virtual City Publisher	Web Tabanlı	Lisanslı	Var
FME	Masaüstü	Lisanslı	Var
CesiumIon	Web Tabanlı	Lisanslı	Var

3. MATERYAL VE METOT

Tez çalışmasında kullanılan arazi hakkında detaylı bilgiler ve kullanılan bölümleme algoritmaları olan QuadTree, Octree, R-Tree, Kd-Tree ve önerilen Optimize QuadTree yöntemlerinin açıklaması bu bölümde ele alınmıştır.

3.1.Materyal

3.1.1. Obj23dtiles kütüphanesi

Obj23dtiles, obj formatındaki 3D modelleri 3D Tiles formatına dönüştürme işlevselliği sunar. Bu, büyük 3D veri kümelerini daha küçük tilelara bölerek etkili bir şekilde işlemenize ve iletmemenize olanak tanır. Web üzerinde veri transferini ve görselleştirmeyi optimize etmek için önemlidir.

Obj23dtiles, Cesium ekosistemiyle uyumlu olarak çalıştığı ve açık kaynak bir Node.js modülü olduğu için tez çalışmasında bu kütüphanenin kullanılması tercih edilmiştir.

3.1.2. Visual Studio Code

Visual Studio Code Microsoft tarafından Windows, Linux ve MacOS için geliştirilen bir kaynak kodu düzenleyicisidir. Çok çeşitli programlama dilleri ve çerçeveleri için sözdizimi vurgulama, kod tamamlama, hata ayıklama, sürüm kontrol entegrasyonu ve daha fazlası dahil olmak üzere birçok özellik ve uzantı sağlar. Ayrıca topluluk tarafından oluşturulan birçok eklentiyi ve uzantıyı destekleyerek geliştiricilerin işlevselliğini özelleştirmesine ve genişletilmesine olanak tanır. Tez çalışmasında kullanım kolaylığı, hızı ve çok yönlülüğü nedeniyle Visual Studio Code tercih edilmiştir.

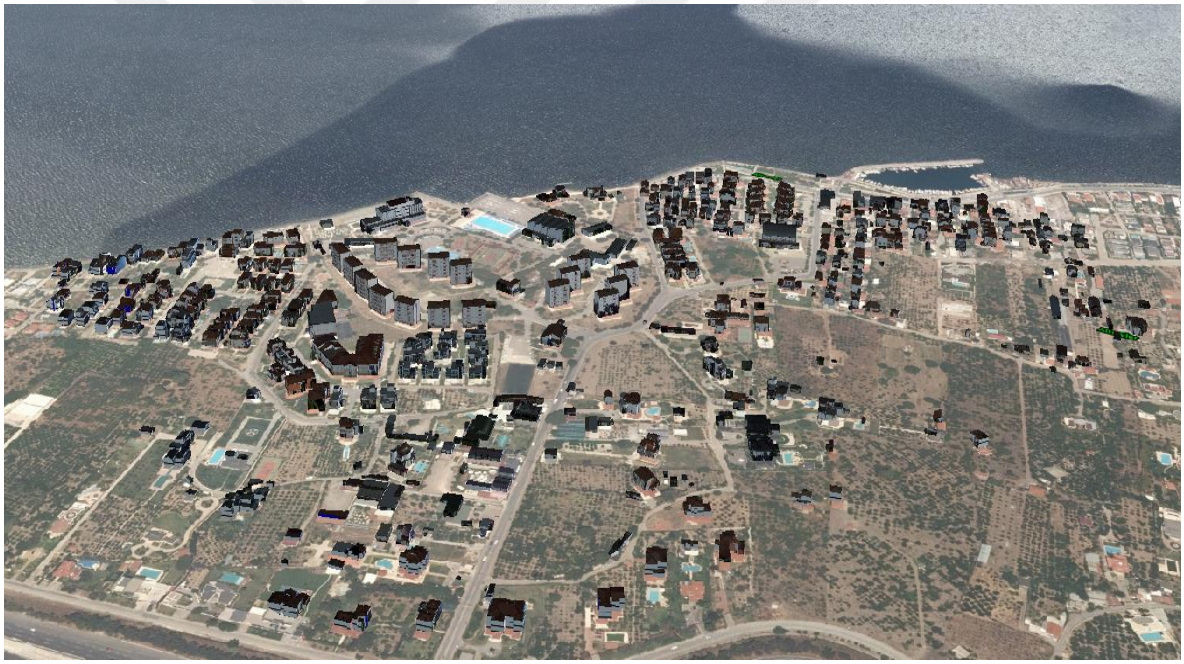
3.1.3. CesiumIon

CesiumIon, büyük miktarda jeo-uzamsal veri, uydu görüntüleri, arazi verileri ve 3D modelleri gibi çeşitli veri tiplerini etkili bir şekilde yönetme ve analiz etme yetenekleri sunar. CesiumIon, kullanıcı dostu bir arayüze sahiptir ve veri yükleme, işleme ve görselleştirme gibi işlemleri kolayca gerçekleştirmenize olanak tanır. Böylece, karmaşık veri işleme süreçlerini basitleştirir (Cesium). Farklı veri türlerini destekler, bu da kullanıcıların uydu görüntüleri, arazi verileri, 3D modeller, nokta bulutları ve diğer jeo-uzamsal verileri platforma yüklemelerini sağlar. Bu tür avantajlarından dolayı tez çalışmasının sonuçlarının görselleştirilmesi için CesiumIon tercih edilmiştir.

3.1.4. Çalışmada kullanılacak arazi verileri

3D Tiles probleminde farklı bölümleme algoritmalarının başarısını ölçmek için 3 farklı bölge üzerinde çalışmalar yapılmıştır. Bu araziler küçük, orta ve büyük olarak farklı boyutlardadır. Böylelikle yapılan çalışmanın etkisi daha sağlıklı gözlenmiş olur. Çalışmada kullanılacak araziler büyüklüklerine göre sırasıyla İzmir Narlıdere ilçesi, Hakkari Biçer Mahallesi ve Ankara Çankaya ilçesidir.

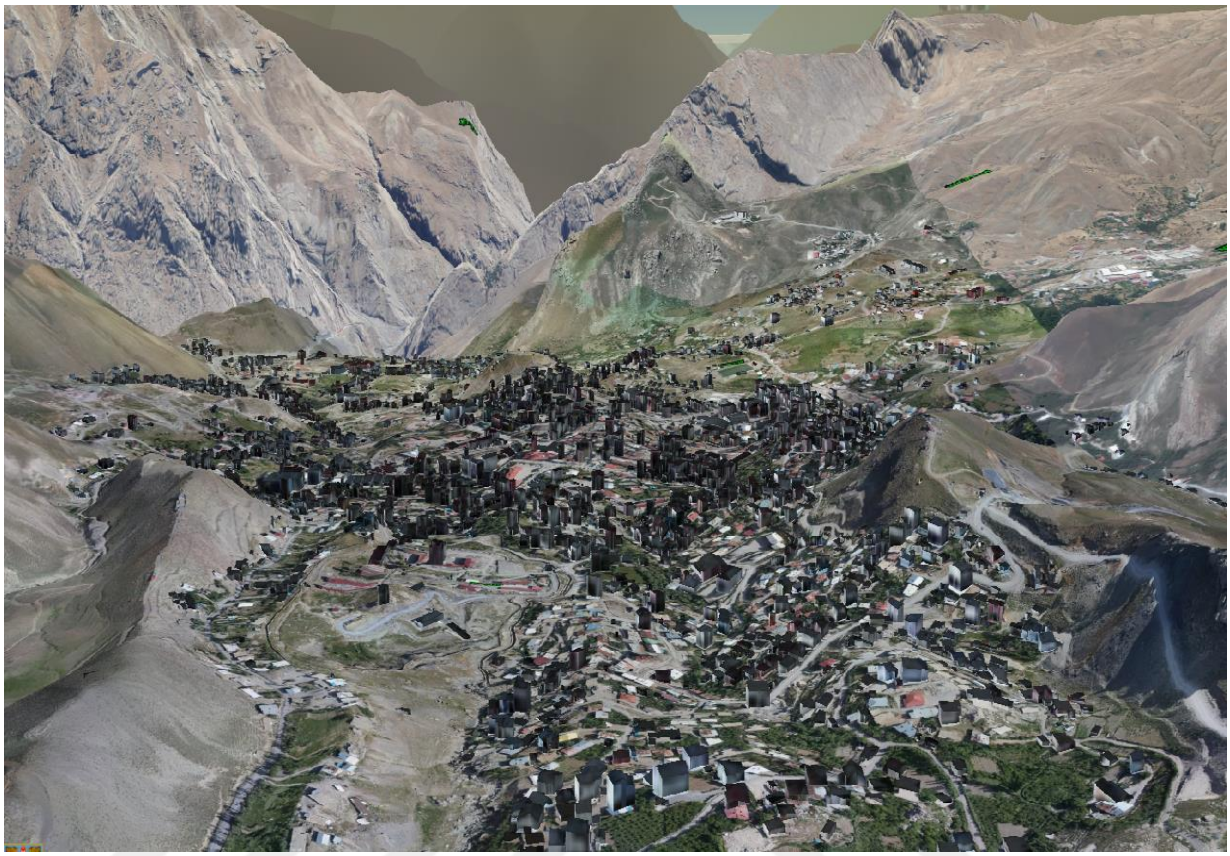
İzmir Narlıdere ilçesine ait veriler 1098 binadan oluşmaktadır. Bu veriler kıyaslama için 4 parçaya ayrılarak 21 bina için, 100 bina için, 500 bina ve 1098 bina için performans kıyaslaması yapılmaktadır. Narlıdere ilçesi verisi toplamda 1098 binadan oluşan, 289MB veri boyutuna sahip, içerisinde koordinatlar için map.txt dosyası barındıran, toplamda 3.294 adet dosyadan oluşan ham obj formatında görüntülerin bulunduğu veri setidir. Şekil 6'de Narlıdere ilçesinin 3D görüntüsü görülmektedir.



Şekil 6. İzmir Narlıdere ilçesi 3D görünümü

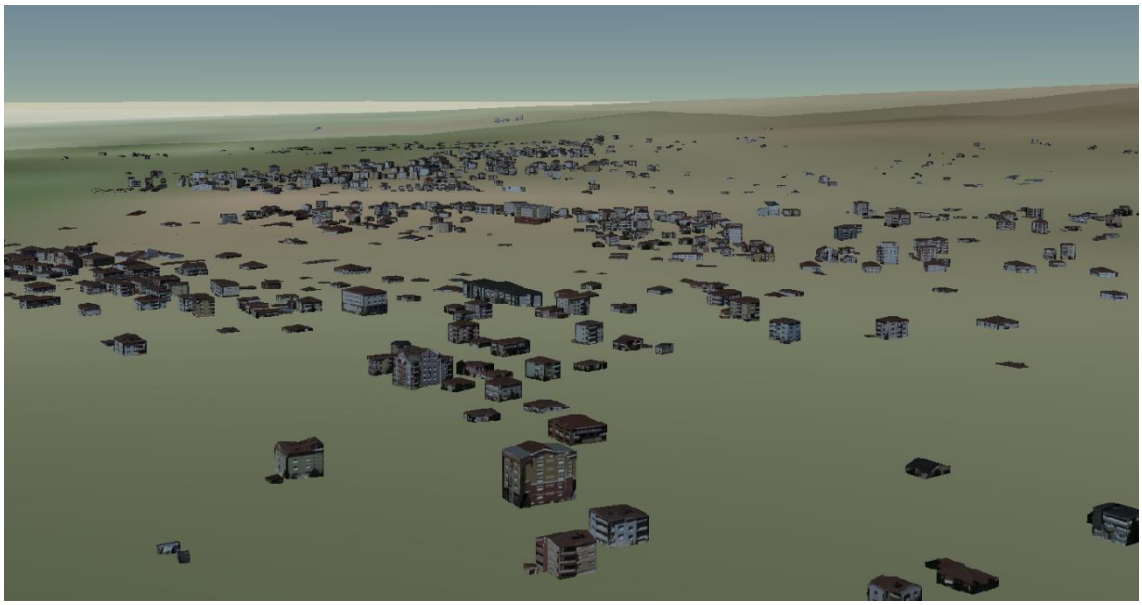
Orta boyuttaki veri bölgemiz olan Hakkari merkez Biçer Mahallesinde toplamda 8207 bina bulunmaktadır. Bu veriler 4 parçaya ayrılarak 21 bina için, 1000 bina için, 4000 bina ve 8207 bina için performans kıyaslaması yapılmaktadır. Böylece yapılan çalışmanın etkisi farklı boyutlarda girdi verilesi için elde edilmiş olur. Şekil 7'de Hakkari Biçer Mahallesinin 3D görüntüsü mevcuttur. Biçer Mahallesi verisi toplamda 8207 binadan oluşan, 1.45GB veri boyutuna sahip, içerisinde koordinatlar için coi

dosyası barındıran, 30.773 adet dosyadan oluşan ham obj formatında görüntülerin bulunduğu veri setidir.



Şekil 7. Hakkari Biçer Mahallesi 3D görünümü

Son çalışma alanı olarak Ankara Çankaya ilçesine ait veriler kullanılmıştır. Toplamda 70.560 binadan oluşmakta olan veri seri kıyaslama için 4 parçaya ayrılarak 21 bina için, 10000 bina için, 50000 bina ve 70.560 bina için performans kıyaslaması yapılmaktadır. Ankara Çankaya verisi toplamda 70.560 binadan oluşan, 7.37GB veri boyutuna sahip, içerisinde koordinatlar için map.txt dosyası barındıran, 211.680 adet dosyadan oluşan ham obj formatında görüntülerin bulunduğu veri setidir. Şekil 8'de Ankara Çankaya ilçesinin 3D görüntüsü görülmektedir.



Şekil 8. Ankara Çankaya ilçesi 3D görünümü

3.2.Metot

3.2.1. QuadTree algoritması

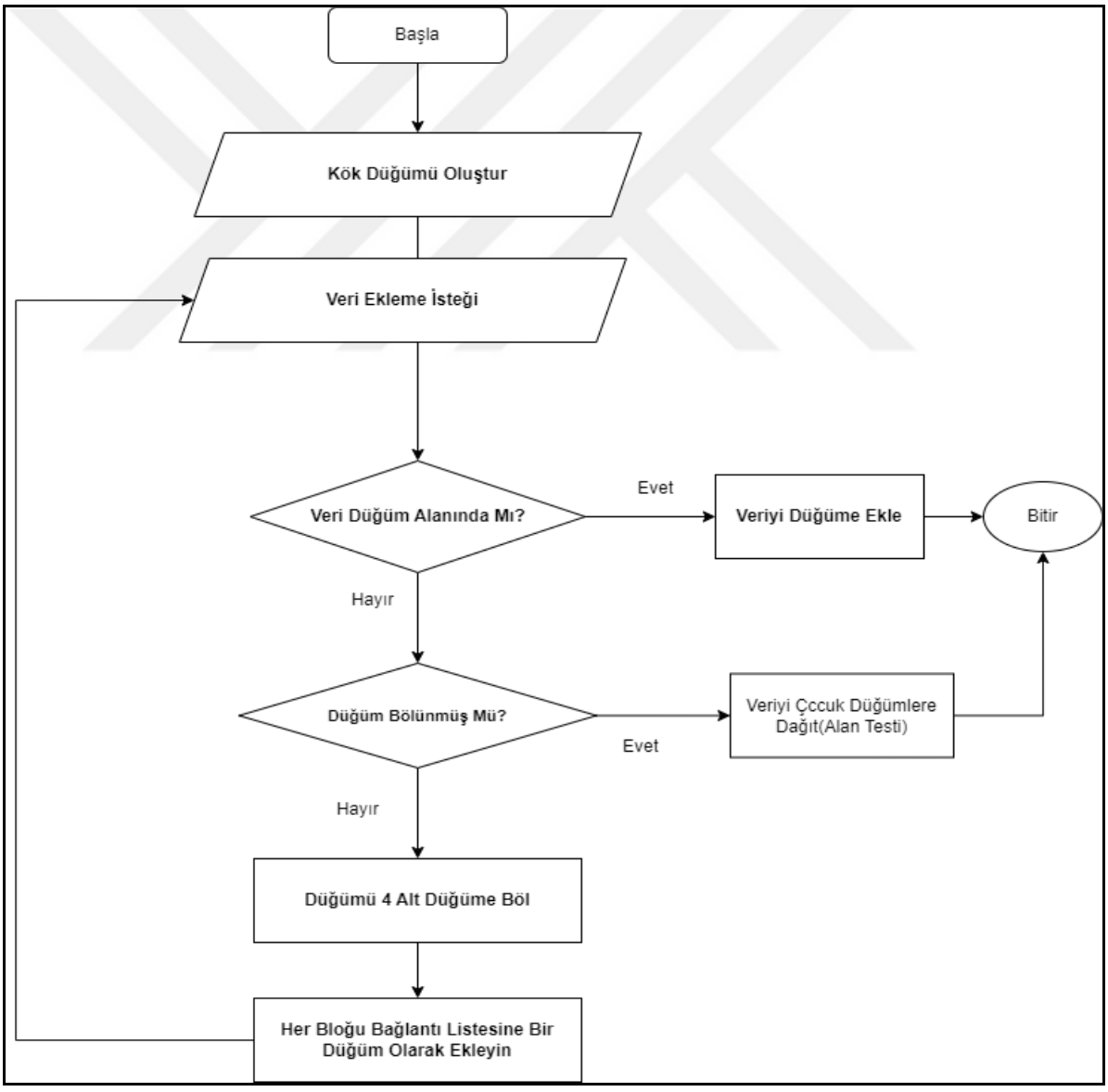
QuadTree, bir iki boyutlu alanı etkili bir şekilde bölen ve veri noktalarını indekslemek ve aramak için kullanılan bir algoritmadır. QuadTree'nin temel özelliklerinden biri, 2 boyutlu uzayı hiyerarşik olarak dört parçaya bölen bir ağaç yapısı olmasıdır. Bu yapı, her düğümün dört alt düğüme sahip olduğu bir ağaç yapısı oluşturur. Bu özellik, alan sorguları için hızlı bir şekilde cevap verebilme yeteneği sağlar (Haverkort ve dig., 2016).

QuadTree, birçok uygulama alanında kullanılmaktadır. Özellikle, coğrafi bilgi sistemleri, bilgisayar grafikleri, görüntü işleme ve sıkıştırma gibi alanlarda önemli bir rol oynamaktadır. Örneğin, coğrafi bilgi sistemlerinde, QuadTree'nin kullanımı, verilerin harita üzerinde etkili bir şekilde temsil edilmesini sağlar (Samet, 2006).

QuadTree algoritması, genellikle döngüler kullanılarak uygulanır. Bu döngüler, ağaç yapısını oluşturmak ve sorguları gerçekleştirmek için kullanılır. QuadTree'nin etkili bir şekilde uygulanması, veri yapısının her düğümündeki verilerin doğru şekilde işlenmesini gerektirir (Samet, 2006). Bir QuadTree'nin performansı, genellikle derinliği, veri yoğunluğu ve sorgu tipine bağlıdır. Daha derin bir QuadTree, daha fazla ayrıntı sağlayabilir, ancak sorgu süreleri artabilir. Bu nedenle, QuadTree'nin tasarımında dikkatli bir dengeleme gereklidir.

Şekil 9'da QuadTree için akış diyagramı verilmiştir. QuadTree'nin ilk adımı, bir başlangıç noktası oluşturmaktır. Bu nokta genellikle bir kök düğüm olarak adlandırılır ve genellikle alana ve verilere ilişkin bilgiler içerir. Veri, QuadTree'ye eklenmek

istendiğinde, öncelikle hangi düğüme ekleneceğine karar verilmelidir. Eğer veri, düğümün alanı içindeyse, veri düğüme eklenir. Eğer düğüm bölünmüşse, veri, hangi çocuk düğümlerine ekleneceğine göre uygun olan düğüme eklenir. Bir düğüm belirli bir kurala göre bölünmüş olabilir. Bu bölünme genellikle düğümün alanının bir kısmını dört eşit parçaya böler. Bu şekilde, her bir çocuk düğüm, orijinal alanın dörtte birini temsil eder. Böylece, daha küçük alanlar üzerinde daha etkili bir işlem yapılabilir. Bir düğüm belirli bir kurala göre bölünmüş olabilir. Bu bölünme genellikle düğümün alanının bir kısmını dört eşit parçaya böler. Bu şekilde, her bir çocuk düğüm, orijinal alanın dörtte birini temsil eder. Böylece, daha küçük alanlar üzerinde daha etkili bir işlem yapılabilir.



Şekil 9. QuadTree akış diyagramı

3.2.2. Octree algoritması

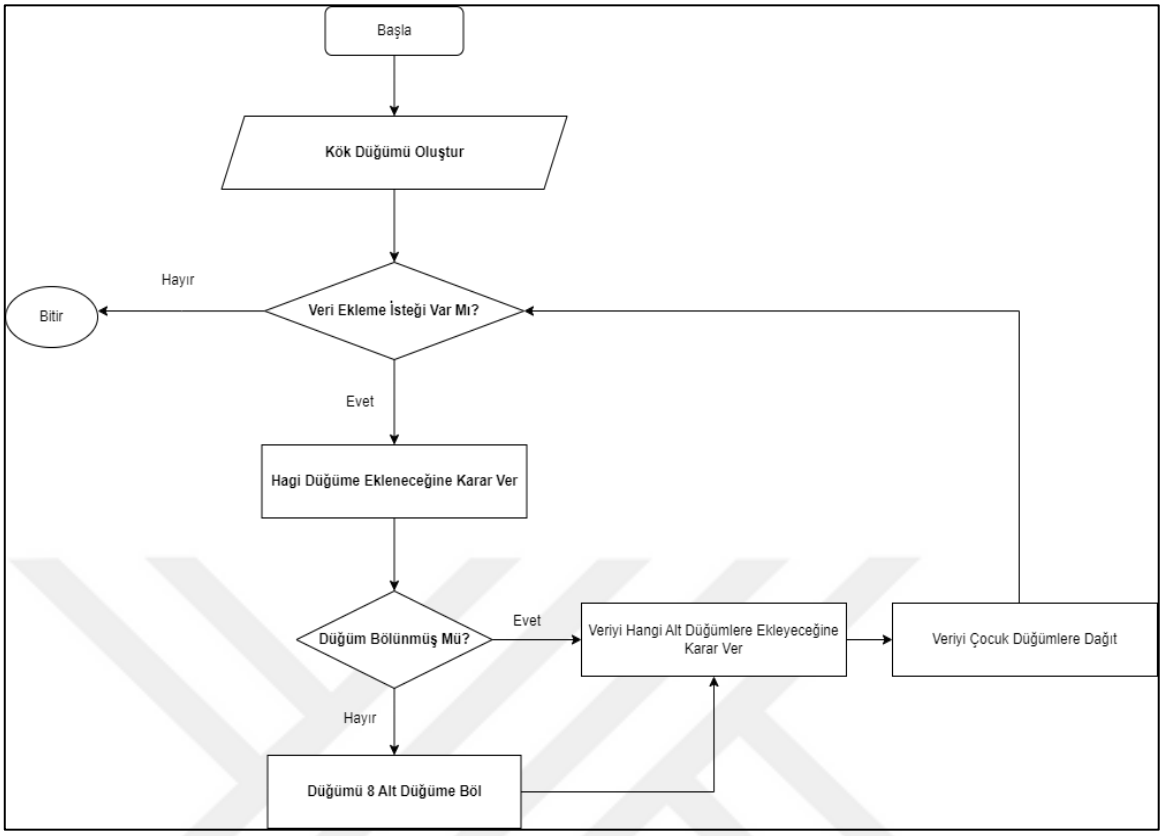
Octree, üç boyutlu uzayı hiyerarşik olarak sekiz parçaya bölen bir veri yapısı ve sorgu algoritmasıdır. Bu yapı, 3D uzayda verilerin etkili bir şekilde organize edilmesini ve sorgulanmasını sağlar. Octree'nin temel özelliği, 3D uzayı sekiz parçaya bölen bir ağaç yapısı oluşturmasıdır. Her düğüm, sekiz alt düğüme sahip olup, her bir alt düğüm, 3D uzayın bir bölgesini temsil eder. Bu hiyerarşik yapı, verilerin yüksek seviyede organize edilmesini ve alan sorguları için hızlı bir şekilde cevap verebilme yeteneği sağlar (Tong, 2002).

Octree, birçok uygulama alanında kullanılmaktadır. Özellikle, bilgisayar grafikleri, robotik, hacim işleme, medikal görüntüleme ve görüntü işleme gibi alanlarda önemli bir rol oynamaktadır. Örneğin, medikal görüntüleme alanında, Octree'nin kullanımı, 3D verilerin hızlı bir şekilde işlenmesini, analiz edilmesini ve görüntülerin ayrıştırılmasını sağlar (Zhou ve dig., 2012).

Octree algoritması, genellikle döngüler kullanılarak uygulanır. Bu döngüler, ağaç yapısını oluşturmak, veri eklemek veya silmek ve sorguları gerçekleştirmek için kullanılır. Octree'nin etkili bir şekilde uygulanması, veri yapısının her düğümündeki verilerin doğru şekilde işlenmesini gerektirir.

Bir Octree'nin performansı, genellikle derinliği, veri yoğunluğu ve sorgu tipine bağlıdır. Daha derin bir Octree, daha fazla ayrıntı sağlayabilir, ancak sorgu süreleri artabilir. Bu nedenle, Octree'nin tasarımında dikkatli bir dengeleme gereklidir.

Şekil 10'da Octree için akış diyagramı verilmiştir. İlk adım, Octree'nin kök düğümünü oluşturmaktır. Bu kök düğüm genellikle bir üç boyutlu kutu (bounding box) tarafından temsil edilir. Veri, Octree'ye eklenmek istendiğinde, öncelikle hangi düğüme ekleneceğine karar verilmelidir. Eğer veri, düğümün alanı içindeyse, veri düğüme eklenir. Eğer düğüm bölünmüşse, veri, hangi çocuk düğümlerine ekleneceğine göre uygun olan düğüme eklenir. Bir düğüm belirli bir kurala göre bölünmüş olabilir. Bu bölünme genellikle düğümün alanının bir kısmını sekiz eşit parçaya böler. Bu şekilde, her bir çocuk düğüm, orijinal alanın sekizde birini temsil eder. Böylece, daha küçük hacimler üzerinde daha etkili bir işlem yapılabilir. Eğer bir düğüm bölünmüşse ve yeni bir veri eklenirse, bu veri çocuk düğümlere dağıtılmalıdır. Dağıtım, eklenen verinin hangi çocuk düğümlere gideceğini belirlemek için bir kurala dayanabilir. Örneğin, veri kendi hacminin sekizde birinden daha fazla bir kısmını kaplıyorsa, veri o düğüme eklenir.



Şekil 10. Octree akış diyagramı

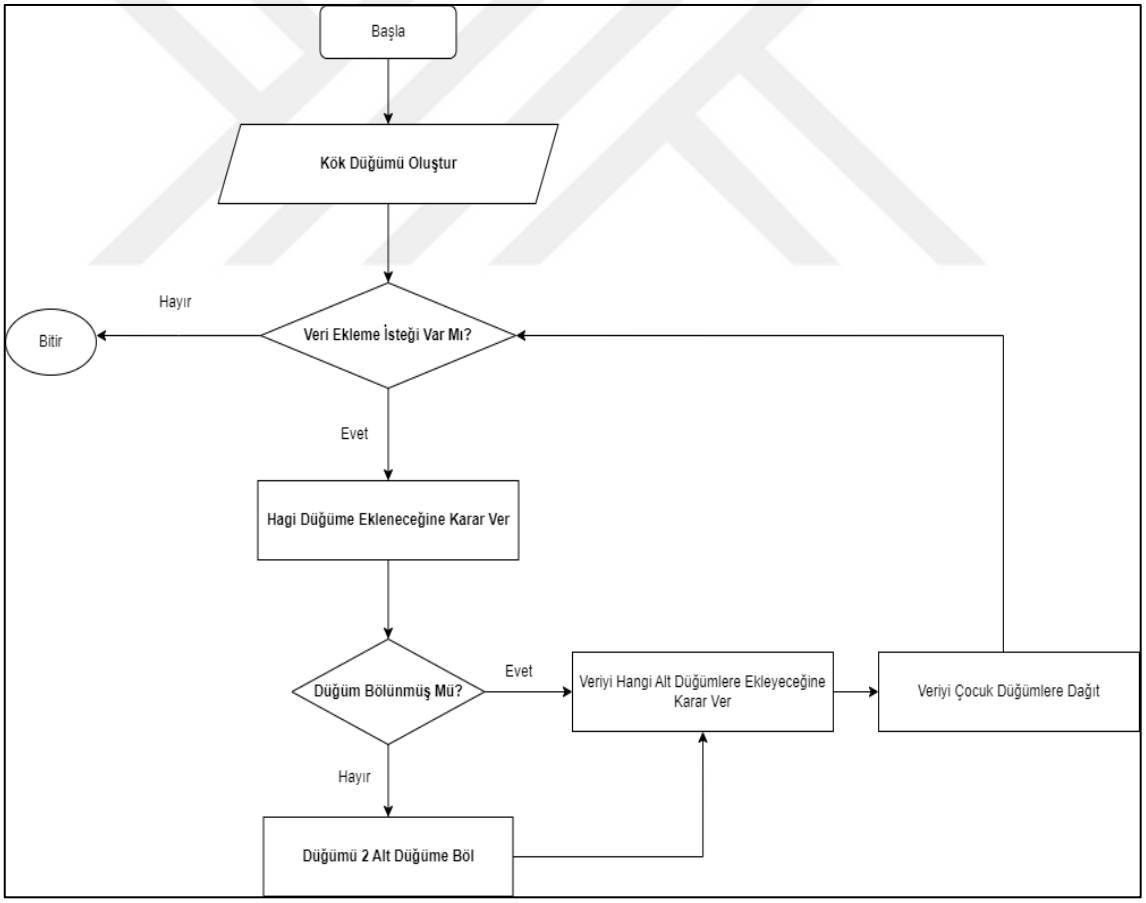
3.2.3. Kd-Tree algoritması

Kd-Tree, çok boyutlu veri kümelerini hızlı bir şekilde organize etmek ve sorgulamak için kullanılan bir veri yapısı ve sorgu algoritmasıdır. Kd-Tree'nin temel özelliği, veri kümesini çok boyutlu düzlemde rekürsif olarak bölmesidir. Her bölünme, bir boyut boyunca veri kümesini iki parçaya böler. Bu işlem, veriyi daha küçük ve daha yönetilebilir parçalara böler, böylece sorguları hızlandırır (Nguyễn ve dig., 2022).

Kd-Tree, birçok uygulama alanında kullanılmaktadır. Özellikle, veri tabanı yönetimi, makine öğrenmesi, bilgisayarlı görü işleme ve robotik gibi alanlarda önemli bir rol oynamaktadır. Örneğin, makine öğrenmesi algoritmalarında, Kd-Tree, veri noktalarını hızlı bir şekilde sınıflandırmak veya aramak için kullanılır (Lu ve dig., 2023).

Bir Kd-Tree'nin performansı, genellikle boyut sayısı, veri yoğunluğu ve sorgu tipine bağlıdır. Daha fazla boyut, ağacın derinliğini artırabilir ve sorgu sürelerini etkileyebilir. Bu nedenle, Kd-Tree'nin tasarımında ve uygulanmasında dikkatli bir dengeleme gereklidir.

Şekil 11’de Kd-Tree için akış diyagramı verilmiştir. İlk adım, Kd-Tree'nin kök düğümünü oluşturmaktır. Bu kök düğüm genellikle veri kümesinin merkezi noktasına göre oluşturulur. Veri, Kd-Tree'ye eklenmek istendiğinde, öncelikle hangi düğüme ekleneceğine karar verilmelidir. Kök düğümde başlayarak, her bir düğümün bölünme düzlemi boyunca veriyi karşılaştırarak hangi yönde ilerleneceğine karar verilir. Bu şekilde, veri hangi düğüme ekleneceği belirlenir ve düğüm oluşturulur. Her düğüm, bölünme düzlemine göre iki alt düğüme bölünür. Bu bölünme düzlemi, düğümde bulunan verilerin özelliklerine göre belirlenir. Örneğin, bir 3-boyutlu Kd-Tree'de, her düğüm bir düzlem tarafından iki parçaya ayrılır. Eğer bir düğüm bölünmüşse ve yeni bir veri eklenirse, bu veri çocuk düğümlere dağıtılır. Dağıtım, eklenen verinin hangi çocuk düğümlere gideceğini belirlemek için bir kurala dayanır. Bu kural, bölünme düzlemi boyunca veriyi karşılaştırarak belirlenir.



Şekil 11. Kd-Tree akış diyagramı

3.2.4. R-Tree algoritması

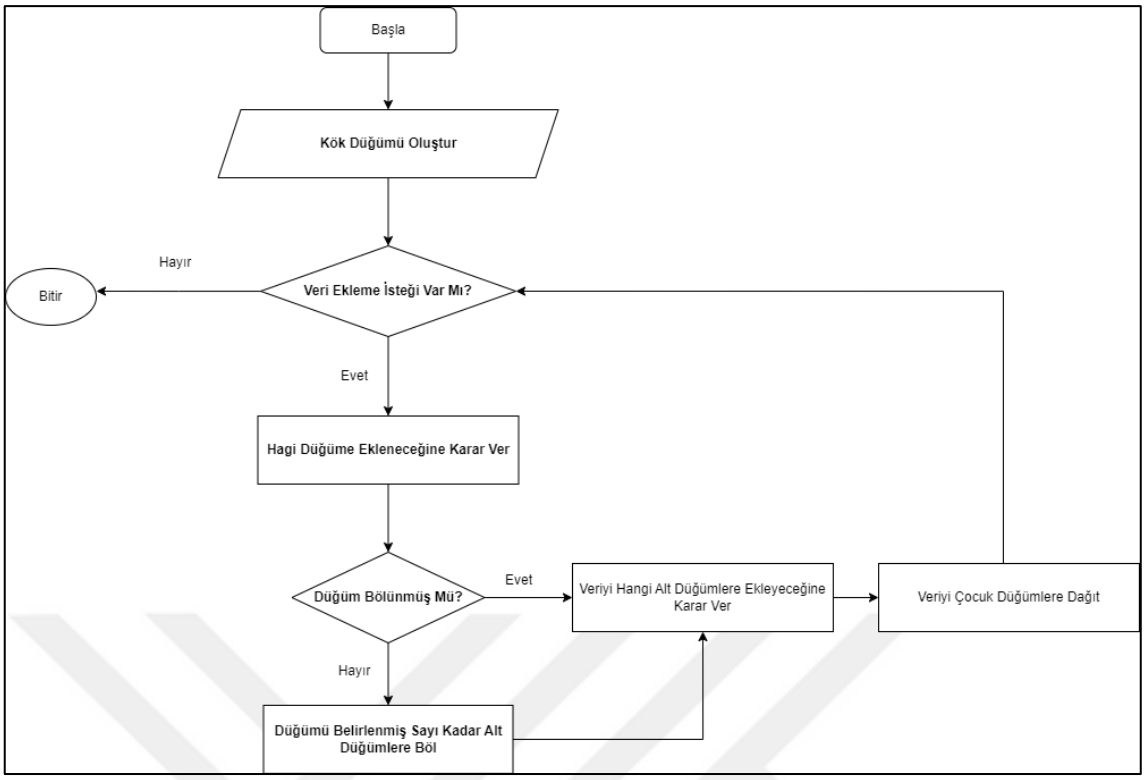
R-Tree, çok boyutlu veri kümelerini organize etmek ve sorgulamak için kullanılan bir veri yapısı ve sorgu algoritmasıdır. R-Tree'nin temel özelliği, veri kümesini bir dizi dikdörtgen kutu içine yerleştirmesidir. Her dikdörtgen, bir boyutlu veya çok boyutlu bir uzayda bir alanı temsil eder. Bu yapı, veri kümesini hiyerarşik olarak organize eder ve alan tabanlı sorguları hızlandırır (Yan, 2010).

R-Tree, birçok uygulama alanında kullanılmaktadır. Özellikle, coğrafi bilgi sistemleri, veri tabanı yönetimi, nesne tanıma ve sorgu işleme gibi alanlarda önemli bir rol oynamaktadır. Örneğin, coğrafi bilgi sistemlerinde, R-Tree, nesnelerin konumunu ve alanlarını hızlı bir şekilde aramak için kullanılır (Yang, 2007).

R-Tree algoritması, genellikle döngüler kullanılarak uygulanır. Her adımda, veri kümesi bir veya daha fazla dikdörtgen içine bölünür ve her bir bölme için aynı işlem tekrarlanır. R-Tree'nin etkili bir şekilde uygulanması, verilerin düzgün bir şekilde dağıtılmasını ve her düğümün optimum boyutlara sahip olmasını gerektirir.

R-Tree algoritması, çok boyutlu veri kümelerinin etkili bir şekilde işlenmesi ve sorgulanması için önemli bir araçtır. Yaygın uygulama alanları ve esnek yapısı, R-Tree'nin bilimsel ve endüstriyel uygulamalarda yaygın olarak kullanılmasını sağlar.

Şekil 12'da R-Tree için akış diyagramı verilmiştir. İlk adım, R-Tree'nin kök düğümünü oluşturmaktır. Bu kök düğüm genellikle boş bir düğümdür ve veri eklenene kadar içerisinde hiçbir şey bulunmaz. Veri, R-Tree'ye eklenmek istendiğinde, öncelikle hangi düğüme ekleneceğine karar verilmelidir. Kök düğümden başlayarak, verinin ekleneceği uygun bir düğüm bulunur. Bu düğümün kapasitesi aşılmamışsa, veri doğrudan bu düğüme eklenir. Kapasite aşılmışsa, uygun bir alt düğüme bölünerek veri eklenir. Bir düğüm bölünmesi gerektiğinde, bu işlem R-Tree'nin mantığına göre gerçekleştirilir. R-Tree, genellikle bir düğümün hacmi veya alanı içindeki verilerin dağılımı gibi kriterlere dayanarak bölünme işlemini gerçekleştirir. Böylece, daha küçük ve daha homojen alt düğümler elde edilir. Eğer bir düğüm bölünmüşse ve yeni bir veri eklenirse, bu veri uygun alt düğümlere dağıtılır. Dağıtım işlemi, verinin eklenmesi için en uygun alt düğümün belirlenmesine dayanır. Bu, genellikle verinin düğümler arasında minimal bir alan veya hacim artışı sağlayacak şekilde yapılan bir seçimdir.



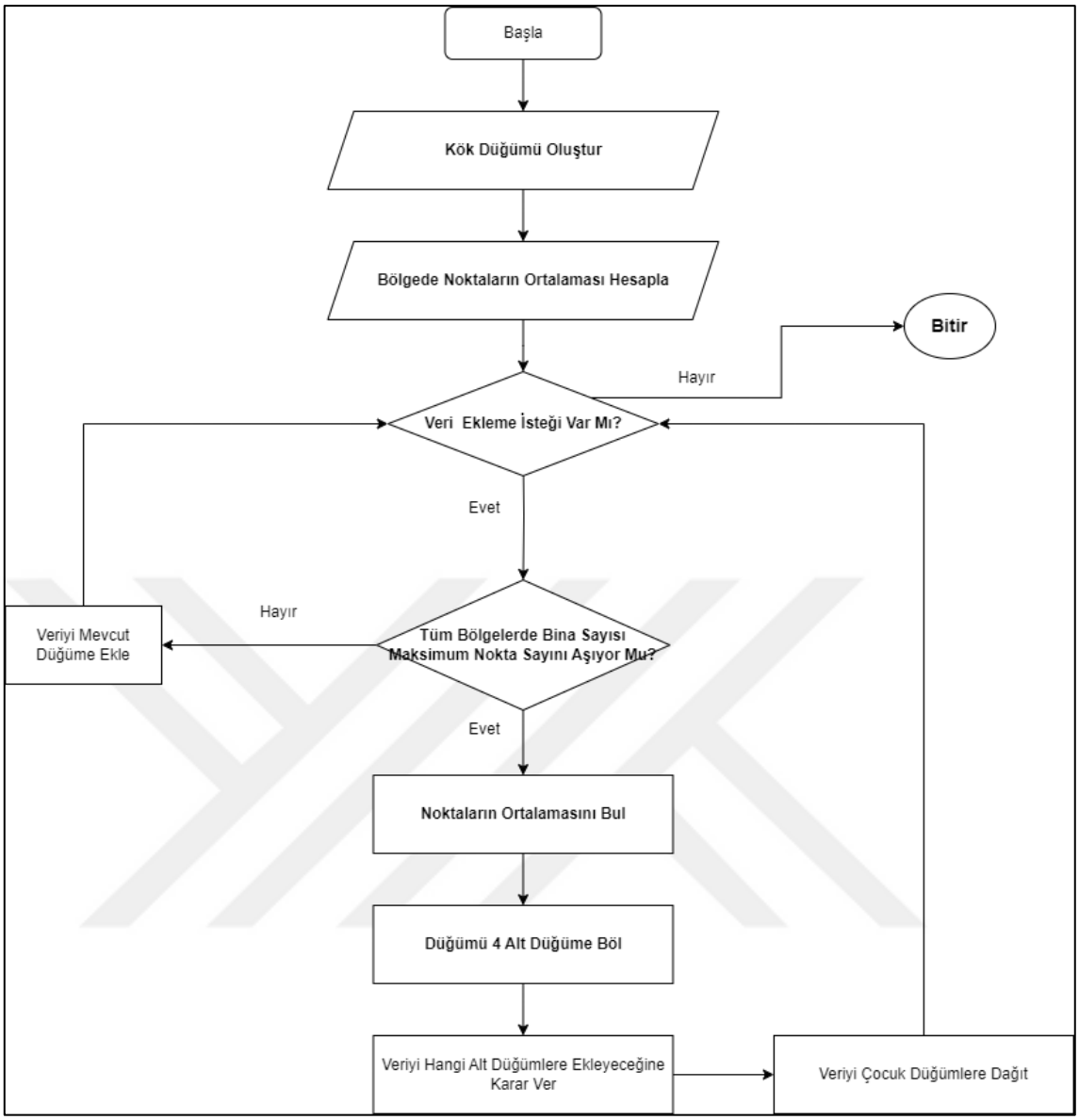
Şekil 12. R-Tree akış diyagramı

3.2.5. Optimize QuadTree algoritması

Bu çalışmada QuadTree algoritmasında bazı değişiklikler yaparak modifiye edilmiş QuadTree algoritması önerilmiştir. QuadTree algoritmasından farklı olarak, uzayı noktaların ortalamasına göre bölerek oluşturulan bir ağaç yapısı oluşturan optimize bir yöntem önerilmiştir. Bu algoritma, uzayı hiyerarşik bir şekilde temsil etmek için kullanılır ve noktaların benzer özelliklere sahip olduğu bölgeleri bir araya getirir.

Şekil 13'da Optimize QuadTree algoritması için akış diyagramı verilmiştir. İlk adım, kök düğümünü oluşturmaktır. Uzay içindeki noktaların ortalamasına göre homojenlik düzeyine göre bölümleme başlar. Başlangıçta, tüm noktalar bir araya getirilir ve ortalaması alınır. Eğer oluşan her bir bölgenin içerisinde kalan bina sayısı maksimum nokta sayısını aşmıyor ise bölme işlemi sona erer. Aksi takdirde, uzay daha küçük parçalara bölünür.

Bölme işleminde uzay dört eşit parçaya bölünür (sol üst, sağ üst, sol alt, sağ alt). Her bir alt bölge içindeki noktaların ortalaması alınır. Her bir alt bölge, maksimum bölme kontrolüne tabi tutulur. Her bir alt bölge, Optimize QuadTree algoritması uygulanarak daha fazla detaylandırılabilir. Bu işlem, belirli bir bina sayısına veya istenen ayrıntı seviyesine ulaşılana kadar devam eder.

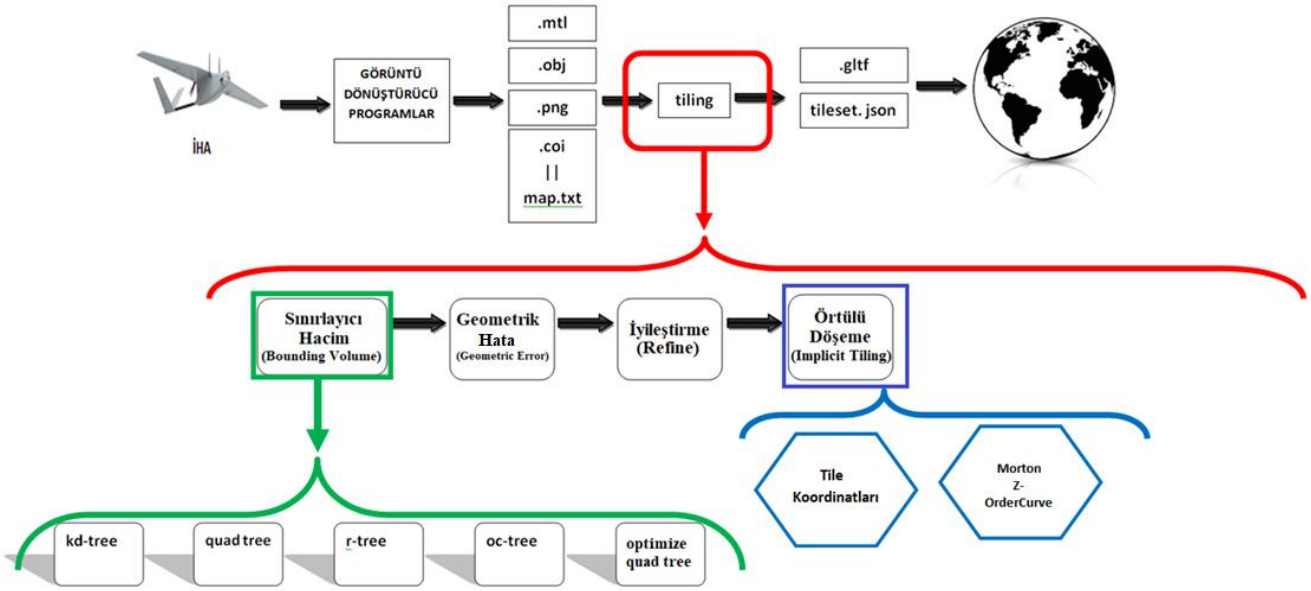


Şekil 13. Optimize QuadTree akış diyagramı

Optimize QuadTree algoritması, noktaların ortalamalarına göre bölme prensibini kullanarak uzayı hiyerarşik bir şekilde temsil etmek için güçlü bir araç olduğu öngörülmektedir. Bu algoritma, homojen bölgelerde veri analizi ve sorgulama için etkili bir şekilde kullanılabilir. Böylece verilerin yoğun olduğu bölgelerde daha çok bölümleme sağlanarak daha homojen bir yapı elde edilebilecektir.

4. 3D TILES PROBLEMİNİN ÇÖZÜMÜ

3D verilerinin elde edilme süreci, Şekil 14'te gösterildiği üzere, şu şekilde gerçekleşmektedir. İlk olarak, insansız hava aracı (İHA) ekipleri araziye giderek belirlenen bölgelerin uçuşlarını gerçekleştirir. Elde edilen görüntüleri, görüntü dönüştürücü programlar aracılığıyla işlenerek her bir 3D model için .mtl, .obj, .png, .coi veya map.txt formatında dört dosya çıktısı üretilir. Bu işlem sonrasında, elde edilen dosyalar tiling işlemine tabi tutulur ve çıktı olarak .gltf dosyaları ile tileset.json dosyası oluşturulur.

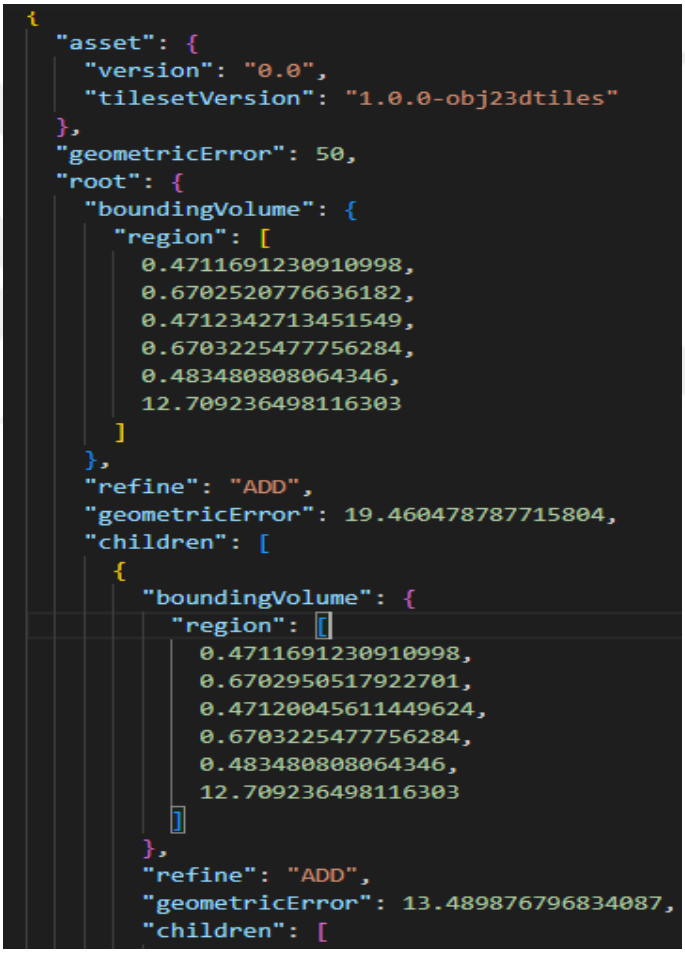


Şekil 14. 3D Tiles Süreci

Gltf dosyaları, 3D model ve sahne verilerini içeren bir dosya formatıdır. Tileset dosyaları ise, 3D Tiles formatında birden çok gltf dosyasını ve ilgili verileri içeren yapılandırmalardır. Bu yapılar, 3 boyutlu coğrafi verilerin etkili bir şekilde depolanması, yönetilmesi ve görüntülenmesi için kullanılır. Şekil 15'de oluşturulan örnek gltf dosyaları görülmektedir.

3D modeli gösterim için kullanılacak olan uygulama (Örneğin; CesiumIon) tile kümesi oluşturmak için gltf dosyalarını tileset.json'a okuyarak gösterimini gerçekleştirir. Şekil 16'da oluşturulmuş bir tileset.json dosyasının içeriği gösterilmektedir.

Şekil 15. Oluşan GLTF örnekleri



Şekil 16. Tileset.json içeriği

Literatürdeki kaynaklar ile oluşturulmuş bir 3D Tiles verisinin tileset.json dosyası incelendiğinde, herhangi bir hiyerarşik detay seviyesinin olmadığı ,tüm 3D modellerin tek bir çocuk altında oluşturulduğu gözlemlenmektedir (Usta, 2021). Bu durumda bir tile yapısının olmamasından dolayı verilerin gösterimi sırasında sistem tüm

modelleri aynı anda sunmaya çalışacak ancak bu gösterim biçimi performans düşüklüğüne, modellerin geç yüklenmesine ve sistemin zorlanmasına sebep olmaktadır. Bu durumu önlemek için Şekil 10'da gösterilen tiling işlemi uygulanmıştır. Sonuç olarak elde edilen tileset.json dosyasına, map.txt dosyasından alınan veri adedi ve koordinatlarına uygun olarak bölümleme algoritması uygulanmış ve böylece bir ağaç yapısı oluşturulmuştur.

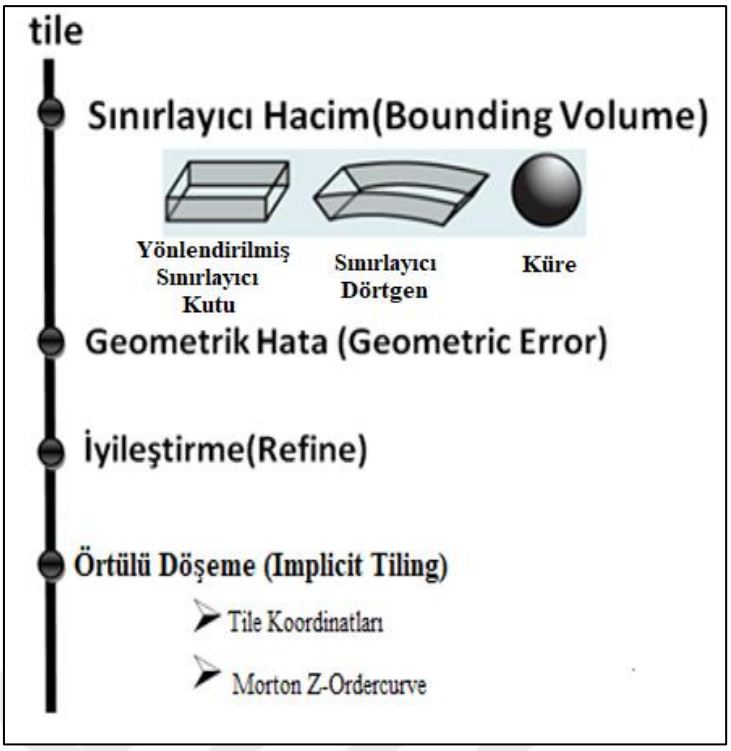
Bölümleme algoritması olarak, Şekil 10'da gösterilen bölümlendirme uygulanarak. Kd-Tree, QuadTree, R-Tree, Octree ve optimize bir bölümleme algoritması geliştirilerek beş algoritmanın performans kıyaslaması yapılmış ve en optimal olanı seçilmiştir. Çıkan ağaç yapısına uygun olarak tileset.json dosyası yeniden düzenlendiği takdirde 3D Tiles yapısına ulaşılmış olunacaktır. Böylece, daha performanslı ve sistemi zorlamayan bir gösterim akışı elde edilmiştir.

4.1. Tiling Aşamaları

Tiling aşaması, 3D Tiles formatındaki verilerin belirli bir bölgeyi kapsayan daha küçük parçalara bölünmesi işlemidir. Bu, büyük ve karmaşık bir coğrafi alana ait 3D verilerin daha etkili bir şekilde yönetilmesini sağlar ve web tabanlı uygulamalarda daha hızlı yükleme ve görüntüleme sağlar. Tiling aşaması sonucunda 3D modellerin görüntülenebilmesi için GLTF dosyaları ve tileset.json dosyası elde edilir.

Tile json, web üzerinde etkileşimli haritaları, özellikle 3D Tiles'ı temsil etmek için kullanılan açık bir standarttır. Json (JavaScript Object Notation) veri biçimini temel alarak, hafif ve okunması kolay bir metin biçiminde veri alışverişi yapmayı sağlar.Bu standart, haritacıların yakınlaştırma düzeyleri, tile URL şablonları ve diğer meta veriler gibi özellikleri basit ve tutarlı bir şekilde tanımlamalarını sağlar (Cozzi, 2015). Tile json, harita sağlayıcılarının haritalarını web tarayıcıları veya mobil uygulamalar gibi istemci uygulamaları tarafından kolayca tüketilebilen bir standart biçimde yayınlamalarına olanak tanır.

Tile json dosyaları, çeşitli harita kitaplıkları ve uygulamalarıyla entegre edilebilir. Aynı zamanda, web harita uygulamaları için harita tilelarını oluşturma ve sunma sürecini standardize ederek basitleştirir (Lilley, 2021). Şekil 17'de tile içeriğinin temsili bir görüntüsü görülmektedir.



Şekil 17. Bir tileset.json nesnesinin öğeleri(Cozzi & Lilley, 2022)

4.1.1. Sınırlayıcı hacim (Bounding Volume)

Sınırlayıcı hacim (SH), 3D bir sahnede bir nesnenin veya bir grup nesnenin kapladığı alanı temsil etmek için kullanılan küreler, kutular veya silindirler gibi basit geometrik şekillerdir. Genellikle bilgisayar grafiklerinde, video oyunu geliştirmede ve fizik simülasyonlarında kullanılır. Temel amacı çarpışma algılama, ışın izleme ve 3D nesneleri içeren diğer işlemlerin karmaşıklığını azaltmaktır (Cozzi, 2022). Karmaşık 3D modellerin kendisinde hesaplamalar yapmak yerine, bu işlemleri üzerinde çalışması daha hızlı ve daha verimli olan ve çok daha basit sınırlayıcı hacimlerde gerçekleştirilebilir.

Sınırlayıcı hacim, ayrıntılı 3D model yerine SH'e dayalı görünürlük testleri gerçekleştirerek işlemeyi optimize etmek için de kullanılabilir. Bu, daha hızlı işleme ve gelişmiş performans sağlar. Her biri kendi güçlü ve zayıf yönleri olan farklı türde sınırlayıcı hacimler vardır. SH türü seçimi temsil edilen nesnelerin karmaşıklığı ve sistemin performans gereksinimler dahil olmak üzere uygulamanın özel gereksinimlerine bağlıdır.

En yaygın SH'ler den bazıları şunlardır:

4.1.1.1. Sınırlayıcı dörtgen (Region)

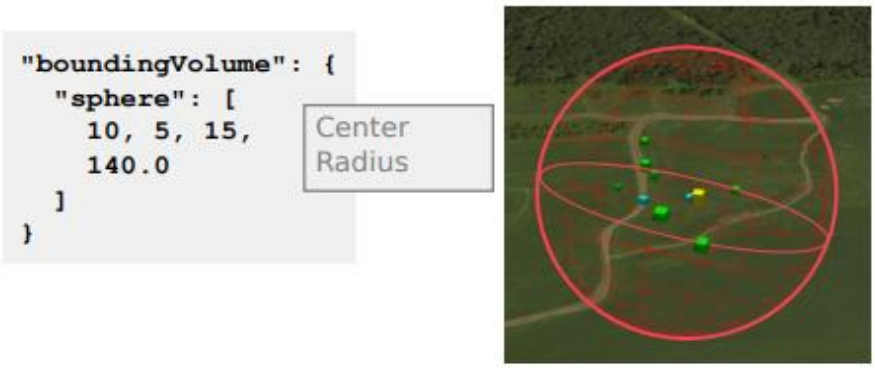
Basit bir sınırlayıcı hacimdir. Bir karonun veya karo grubunun içeriğini içene alan, uzayda dikdörtgen bir bölge tanımlar. Kolay ve verimli kavşak testlerine izin verir.

Bir sınırlayıcı dörtgen bölgesi, Şekil 18'de içeriği de bulunmakta olan sınırlayıcı hacmin minumum ve maksimum köşelerini tanımlayan bir koordinat dizisi kullanarak belirtilir. Sınırlayıcı küre uzlamsal bir kapsam sağlar ve işleme ve akış amaçları için verimli ayırma ve görünürlük belirlemesine olanak tanır (Lilley, 2021).

Şekil 18. Sınırlayıcı dörtgen tipi sınırlayıcı hacim(Lilley, 2021)

4.1.1.2. Küre (Sphere)

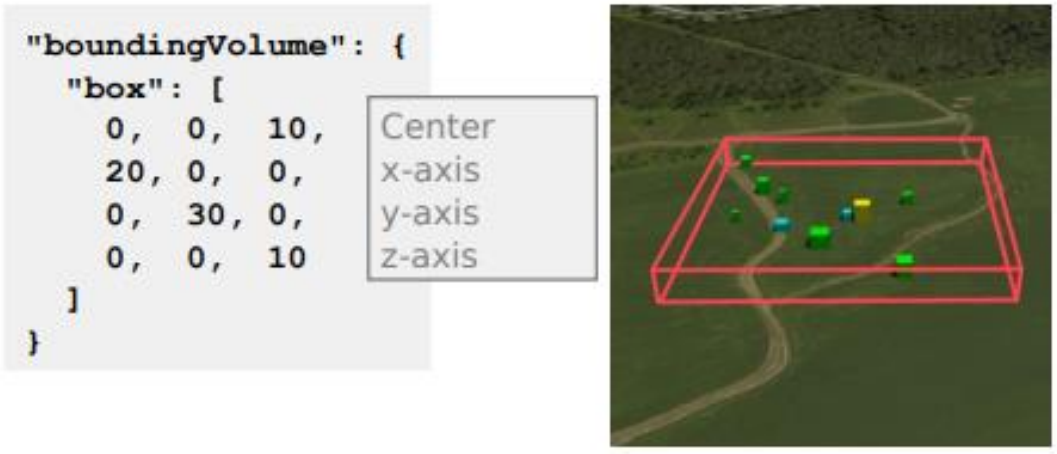
Genellikle hızlı çarpışma testi için kullanılan basit, yuvarlak şekilli bir hacimdir. Merkez konumu ve yarıçapı ile tanımlanır. Şekil 19'da küre tipi sınırlayıcı hacim görülmektedir.



Şekil 19.Küre tipi sınırlayıcı hacim (Lilley, 2021)

4.1.1.3. Yönlendirilmiş sınırlayıcı kutu (Oriented Bounding Box)

Herhangi bir yöne yönlendirilebilen kutu şeklindeki bir hacimdir. Bir koordinat sisteminin eksenleriyle hizalanmayan nesneleri temsil etmek için kullanışlıdır. Kutunun merkez konumu ve üç adet 3D vektör ile tanımlanır (Lilley, 2021). Şekil 20'de yönlendirilmiş sınırlayıcı kutuya ait sınırlayıcı hacim görülmektedir.



Şekil 20. Yönlendirilmiş sınırlayıcı kutu tipi sınırlayıcı hacim (Lilley, 2021)

Belirtilen SH'lar arasından, sınırlayıcı küre kullanmanın, tasarımında önemli avantajları bulunduğu gözlemlenmiştir. Coğrafi bölgeyi doğru bir şekilde temsil eden sınırlayıcı küre, 3D verilerin belirli bir coğrafi alana ait olduğunu belirtir. Bu, verilerin hangi coğrafi bölgeye ait olduğunu netleştirir ve veri yönetimini kolaylaştırır. Ayrıca, sınırlayıcı küre kullanmak hafıza ve kaynak kullanımını azaltabilir ve daha iyi performans sağlayabilir. Özellikle büyük ve karmaşık coğrafi alanlar için ölçeklenebilirlik ve esneklik sağlar. Bu avantajlardan dolayı tez çalışmasında sınırlayıcı küre kullanımı tercih edilmiştir.

4.1.2.Geometrik hata (Geometric error)

Geometrik hata, aynı zamanda geometrik sapma veya geometrik bozulma olarak bilinir; 3D bir nesnenin gerçek şekli ile bilgisayar tarafından oluşturulan bir gösterim arasındaki farkın ölçüsüdür (Cozzi, 2022).

3D grafiklerde, sayısal hesaplamaların kesinliğindeki sınırlamalar, basitleştirilmiş geometrik ilkelerin kullanımı ve düşük çözünürlüklü ağ gösterimlerinin kullanımı gibi çeşitli faktörlerden kaynaklanan geometrik hatalar ortaya çıkabilir (Cozzi ve Lilley, 2022). Bu hatalar, nesnenin gerçek şekli ile bilgisayar tarafından oluşturulan gösterim arasındaki yaklaşıklığı ölçen bir mesafe olarak ifade edilebilir.

Geometrik hata, bilgisayar destekli tasarım, sanal gerçeklik ve bilgisayar oyunları gibi birçok uygulama alanında önemli bir faktördür. Bu uygulamalarda, 3D modellerin doğruluğu, görsel deneyimin kalitesi ve sistem performansı üzerinde önemli bir etkiye sahip olabilir. Daha hassas sayısal hesaplamalar, yüksek çözünürlüklü ağ temsilleri ve karmaşık algoritmalar gibi yöntemler kullanılarak geometrik hatanın

azaltılması amaçlanır, böylece nesnenin gerçek şekline daha yakın bir gösterim elde edilir.

4.1.2.1. 3D karolardaki her karo için geometrik hata nasıl hesaplanır?

Geometrik hata, 3D harita üzerinde belirli bir hata eşiğine göre detay seviyelerini kontrol etmek için kullanılan matematiksel hesaplamaları içerir (Lights, 2021). Her karo için geometrik hatayı hesaplamak için gerekli Denklem 4.1'de formüle edilerek gösterilmiştir.

$$Geometrik\ Hata = \frac{\text{SSE} * \ 0{,}5629165124598852 * \text{Uzaklık}}{936} \tag{4.1}$$

Denklem 4.1'de SSE ile ifade edilen değer ekran alanı hatasını (Screen Space Error - SSE) ifade eder. Bilgisayar grafikleri ve 3D modelleme alanında kullanılan bir tekniktir. Bu teknik, büyük ve karmaşık 3D modellerin optimize edilmesini sağlamak için 3D nesnenin öngörülen boyutu ile piksel arasındaki hataya dayanır. SSE, kamera ile 3D nesne arasındaki mesafeyi, nesnenin ekrandaki boyutunu ve istenen maksimum hata eşiğini dikkate alarak hesaplanır. Hesaplamada kullanılan temel kavram, nesnenin görünen boyutu ile ekranda kapladığı piksel sayısı arasındaki ilişkidir (Lilley, 2021). Kamera nesneye yaklaştıkça, SSE azalır. Bu durumda, ayrıntı düzeyi artar, yani daha fazla detay görüntülenir (Cozzi ve Lilley, 2022). Kamera nesneden uzaklaştıkça, SSE artar. Bu durumda, ayrıntı düzeyi azalır, yani daha az detaylı bir görüntü kullanılır. Bu dinamik ayarlama, kullanıcının bakış açısına ve nesnenin konumuna bağlı olarak optimize edilmiş bir görsel deneyim sağlar.

Eğer kamerayı varsayılan ayarlarla kullanıyorsanız, tarayıcınızın penceresi ekranı tamamen kaplıyorsa ve ekran çözünürlüğünüz 1920x1080 ise, belirli bir değeri temsil etmek üzere kullanılan katsayı yaklaşık olarak 0,5629165124598852'dir. Bu değer, tarayıcı penceresinin durumu, tuval boyutu ve kamera konumu gibi faktörlere bağlı olarak değişebilir, bu nedenle belirtilen değer sadece bir varsayımdır ve uygulama koşullarına göre değişebilir (Lilley, 2021). Burada söz konusu olan değeri 0,5629165124598852 olarak kabul ediyoruz, ancak bu değerin farklı durumlar için uygun şekilde ayarlanması gerekebilir.

Denklem 4.1'de uzaklık değeri, mevcut durumda kameranın dünya koordinat konumundan döşemenin merkez konumuna kadar olan mesafedir, birimi metredir.

Denklem 4.1'de bulunan 936 değeri, Cesium'un tarayıcıda çalışmakta olduğu ekranın piksel yüksekliğini ifade eder. Değeri kendiniz ayarlamazsanız değer ekranın piksel yüksekliği olur. Cesium'unuz tam ekranı kaplıyorsa ve monitörünüzün

çözünürlüğü 1920×1080 ise o zaman bu değer sizin tarayıcınız tam ekran olduğunda genellikle 936 piksel olur (Lights, 2021).

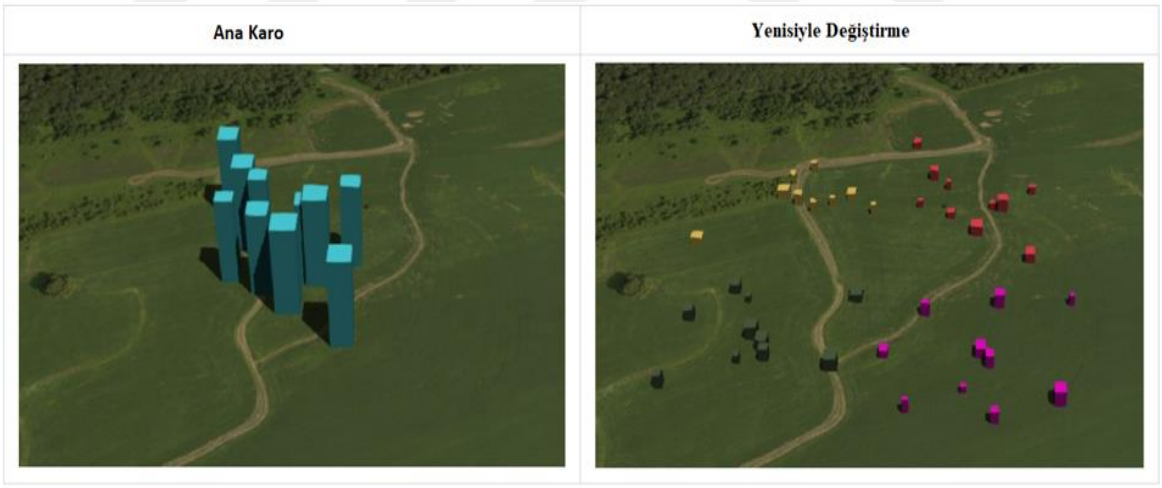
4.1.3. İyileştirme (Refinement)

Genellikle, karoların kalitesini ve görsel görünümünü iyileştirme sürecini ifade eder. 3D karoların tasarımı ve uygulaması bağlamında iyileştirme, uyumlu bir genel tasarım oluşturmak için karoların dikkatlice seçilmesini ve düzenlenmesini veya tek tek karo parçaları arasında daha kusursuz bir geçiş elde etmek için karoların yerleşiminde ince ayarlamalar yapılmasını içerir (Lilley, 2021).

İyileştirme süreci, görünüşü optimize etmek, detayları artırmak ve genel estetik kaliteyi yükseltmek amacıyla uygulanan çeşitli teknikleri kapsayabilir. Bu süreç, özellikle 3D karoların tasarımında ve uygulanmasında, genel estetik dengeyi ve kaliteyi artırmaya odaklanarak karoların seçimi ve düzenlenmesi aşamalarını içerir (Lilley, 2021).

4.1.3.1. Yenisiyle değiştirme (Replecement)

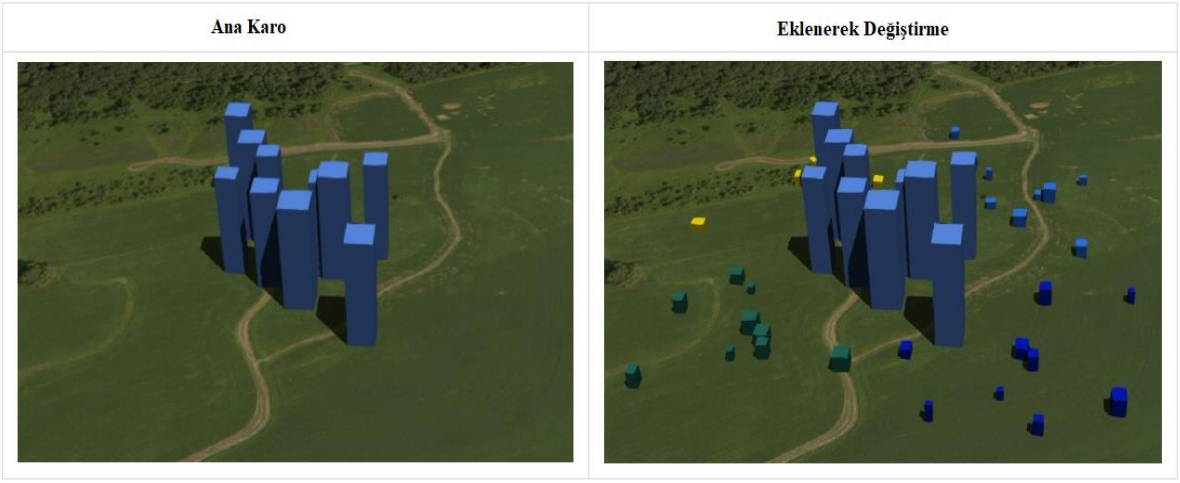
İyileştirildiğinde alt öğelerini kendisinin yerine koyar, yani üst kutucuk artık işlenmez. Yenisiyle değiştirme kullanılarak oluşturulmuş örnek bir karo görünütüsü Şekil 21'de görülmektedir.



Şekil 21. Yenisiyle değiştirme kullanılarak oluşturulmuş karo (Cozzi ve Lilley, 2022)

4.1.3.2. Eklenerek değiştirme (Additive)

İyileştirildiğinde kendisini ve alt öğelerini aynı anda işler. Alt kutucuklar ana kutucuğa ek olarak işlenir. Ekleyerek değiştirme kullanılarak oluşturulmuş örnek bir karo görünütüsü Şekil 22'de görülmektedir.



Şekil 22. Eklenerek değiştirme kullanılarak oluşturulmuş karo (Cozzi & Lilley, 2022)

Bir tileset, yalnızca yenisiyle değiştirme iyileştirmesini, yalnızca eklenerek değiştirme iyileştirmesini veya iki iyileştirmenin de herhangi bir kombinasyonunu kullanabilir. Bir karo setinin kök karosu için bir ayrıntılandırma türü gereklidir; diğer tüm karolar için isteğe bağlıdır. Eklenmediğinde, bir kutucuk üst öğesinin ayrıntılandırma türünü devralır (Lilley, 2021).

Eklenerek değiştirme, verilere zarar vermeden ek veri katmanları oluşturmaktadır. Bu esnekliğin geliştirme sürecini kolaylaştırıp, performansı artırdığı gözlemlenmiştir. Bu tür avantajlarından dolayı tez çalışmasında iyileştirme türü olarak eklenerek değiştirme tercih edilmiştir.

4.1.4. Örtülü döşeme (Implicit Tiling)

Oluşturan sınırlayıcı hacimlerin içerisine uygun bir şekilde 3D modellerin yerleştirilmesi aşamasına örtülü döşeme ismi verilmektedir.

Bilgisayar grafikleri ve uzamsal veri yönetimi dünyasında, büyük ölçekli veri kümelerinin verimli temsili ve işlenmesi sürekli bir zorluktur. Örtülü döşeme, veri düzenleme ve depolamaya çok yönlü bir yaklaşım sunarak bu zorluğun üstesinden gelmek için güçlü bir teknik olarak ortaya çıkmıştır (Cozzi ve Lilley, 2022). Karmaşık 3D modellerin ağlar üzerinden iletildiği ve işlendiği 3D model akışına uygulanabilir. Modeli geometrik özelliklere veya doku çözünürlüğüne göre bölümlere ayırarak, modellerin kademeli olarak yüklenmesini ve akışını sağlayarak sorunsuz görüntüleme deneyimlerini kolaylaştırır.

Örtülü döşemenin uyarlanabilir doğası verilerin uygun bir ayrıntı düzeyinde temsil edilmesini sağlayarak depolama gereksinimlerini en aza indirir ve aktarım sırasında bant genişliği kullanımını azaltır. Verilerin isteğe bağlı olarak yüklenmesini ve

işlenmesini sağlayarak genel performansı artırır. Kamera mesafesi, kullanıcı etkileşimi veya hesaplama kaynakları gibi faktörlere dayalı olarak ayrıntı düzeyini dinamik olarak ayarlama yeteneği sağlar. Bu dinamik ayrıntı düzeyi, kaynakların optimum şekilde tahsis edilmesini sağlayarak gerçek zamanlı işleme ve etkileşimli deneyimlerle sonuçlanır (Cozzi ve Lilley, 2022).

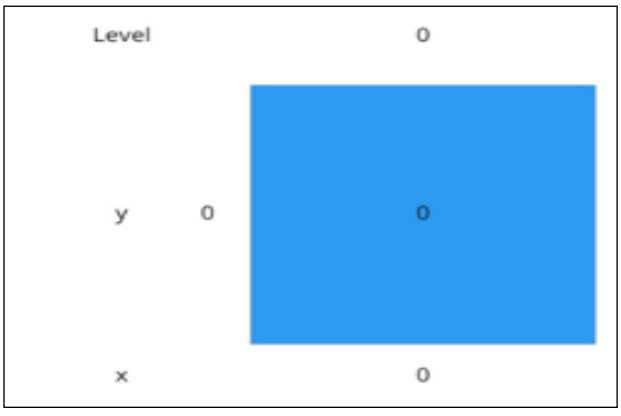
Şekil 10'da gösterilen tile koordinatlarının bulunması ve Morton Z-ordercurve doldurma eğrisi işlem adımları bu aşama için gerçekleştirilir.

4.1.4.1.Tile koordinatları

Tile koordinatları, sınırlayıcı hacimlerle belirlenen tileları tanımlamak ve yerleştirmek için kullanılan sistemi ifade eder. Oluşturulan GLTF dosyaları ile tileset.json dosyasının birbiriyle sağlıklı bir şekilde anlaşabilmesi için tile konumunu ve ayrıntı düzeyini benzersiz bir şekilde belirtmenin yolunu sağlar.

Örtülü döşemedeki tile koordinat sistemi, hiyerarşinin her seviyesinin farklı bir ayrıntı düzeyini temsil ettiği hiyerarşik bir yapıya dayanır. Tile koordinatları, konumlarına ve ayrıntı düzeylerine göre belirli tileların verimli bir şekilde gezinmesine ve alınmasına izin verecek şekilde düzenlenir (Cozzi ve Lilley, 2022).

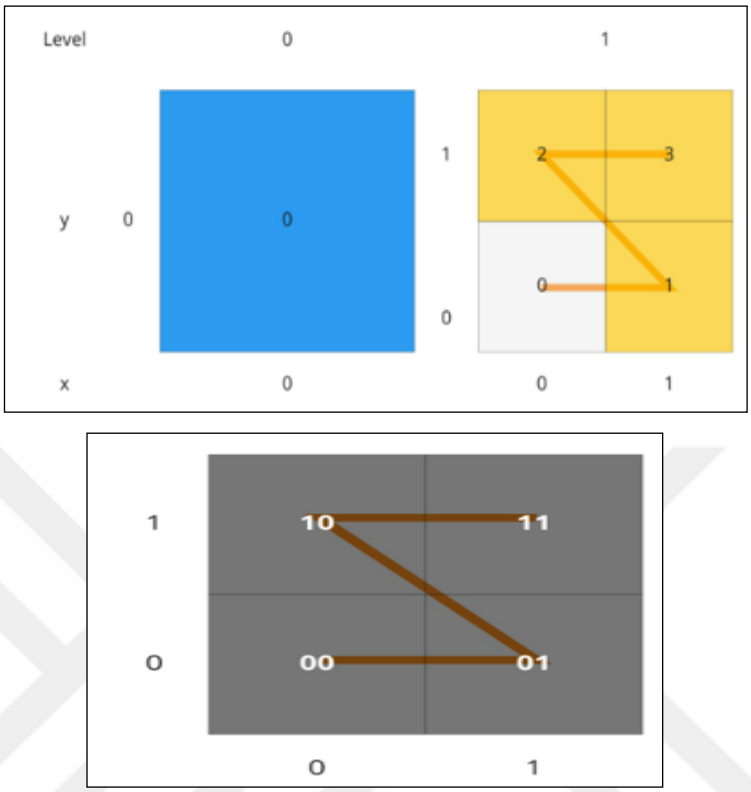
Bu yapının nasıl çalıştığını bir örnekle inceleyelim. Dört ağaç tabanlı örtük döşeme yapısıyla temsil edilen bir arazi veri kümesini düşünelim. En yüksek ayrıntı seviyesi (Level 0), tüm arazi veri kümesi tek bir tile ile temsil edilir (Cozzi ve Lilley, 2022). Şekil 23'de görüldüğü üzere bu seviye 0 seviyesidir ve tile koordinatı (0,0) olur ve bunun kök tile olduğunu gösterir.



Şekil 23. Kök tile görünümü(Cozzi & Lilley, 2022)

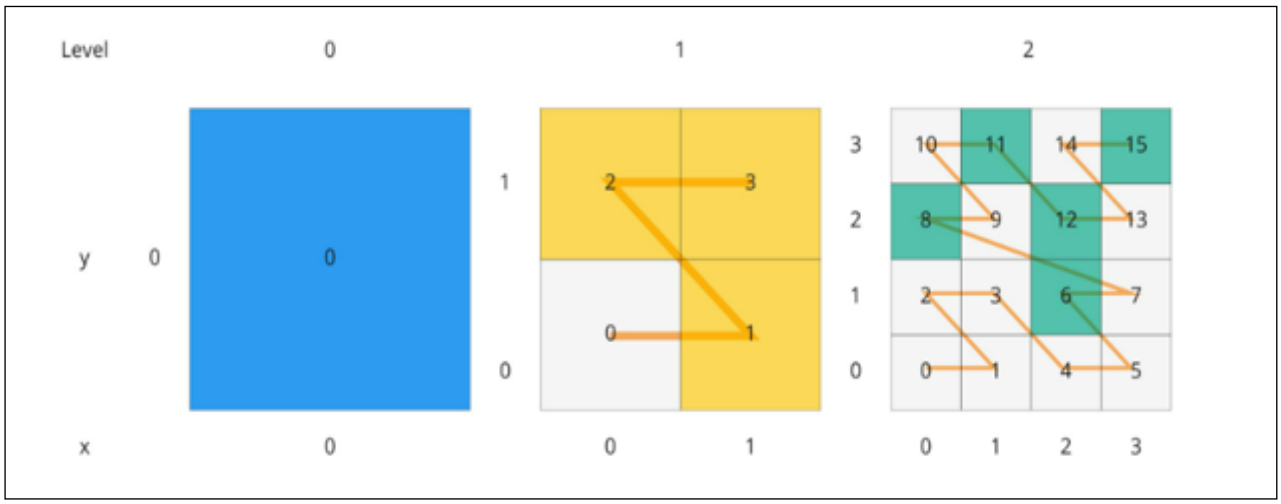
Bir sonraki ayrıntı düzeyine (Level 1) geçtiğimizde, arazi veri kümesi 2x2'lik bir ızgara oluşturan dört parçaya bölünür (Cozzi ve Lilley, 2022). Şekil 24'de görüldüğü

üzere 1. düzey tilelar için tile koordinatları (0,0),(0,1),(1,0) ve (1,1) olacaktır ve bu değerler ızgara içindeki göreli konumlarını gösterir.

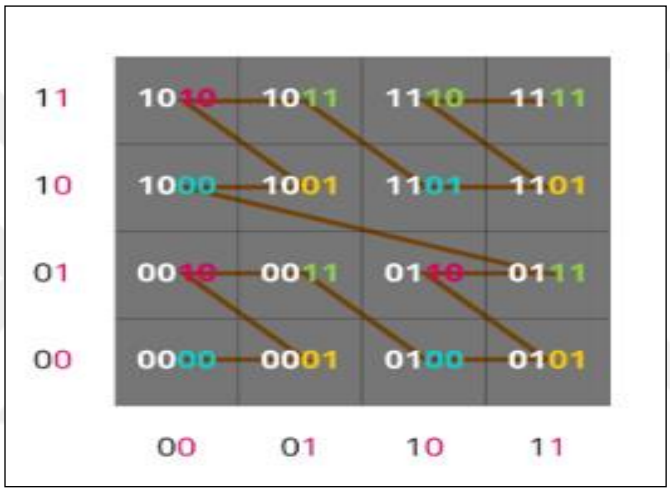


Şekil 24. 1. seviye bölümleme sonucu tile görüntüsü ve indeksleme (Cozzi ve Lilley, 2022)

Bu hiyerarşik alt bölümleme ile devam edildiğinde, 2. seviyede, her 1. seviye karo ayrıca dört karoya bölünerek 2. seviye karolarda oluşan 4x4'lük bir ızgara elde edilir. Şekil 25'de 2. seviye bölümleme sonucu tile görüntüsü ve indekslemesi görülmektedir. Bu 2. seviye döşemler için tile koordinatları Şekil 26'da görüldüğü üzere (0,0,0,0), (0,0,0,1), (0,1,0,0)… gibi dört indeks kullanılarak belirtilecektir (Cozzi ve Lilley, 2022).



Şekil 25. 2. seviye bölümleme sonucu tile görüntüsü ve indeksleme (Cozzi ve Lilley, 2022)



Şekil 26. 2. seviye bölümleme sonucu (Cozzi & Lilley, 2022)

Tile yapısı sonraki her ayrıntı düzeyinde daha küçük tilelara bölündükçe tile koordinatları yenilemeli olarak tanımlanmaya devam eder.

Tile koordinatları, yalnızca tile yapısı içindeki tile'ın konumunu temsil etmez, aynı zamanda tilelar arasındaki hiyerarşik ilişki hakkında bilgi sağlar. Hiyerarşik yapı, tile koordinatlarına göre belirli tileların verimli bir şekilde geçişine, gezinmesine ve alınmasına olanak tanır (Cozzi ve Lilley, 2022). Uygulamalar tile koordinatlarını kullanarak örtük döşeme yapısı içindeki farklı ayrıntı düzeylerindeki tileları kolayca tanımlayabilir ve bunlara erişebilir.Bu, izleyicinin gereksinimlerine ve etkileşimine dayalı olarak uygun ayrıntı düzeyinin sunulmasını sağlayarak verilerin verimli bir şekilde oluşturulmasını, akışını ve işlenmesini sağlar.

Tile koordinatları için belirli koordinat sistemi ve kurallarının tile şemasına ve uygulamaya bağlı olarak değişebileceğini bilmek önemlidir.Burada verilen örnek, arazi veri kümlerinde yaygın olarak kullanılan ve veri kümemiz için uygulanan dört ağaç

benzeri bir yapıya dayanmaktadır, ancak diğer tile şemaları, farklı koordinat sistemlerini veya hiyerarşileri kullanabilir.

4.1.4.2. Morton Z-OrderCurve

Morton Z-ordercurve, Morton sırası veya serpiştirme sırası olarak da bilinir, çok boyutlu verileri tek boyutlu bir doğrusal dizide eşleyen bir boşluk doldurma eğrisidir.1966'da bilgisayar bilimcisi GuyG.Morton tarafından icat edildi ve verimli mekansal indeksleme ve veri alma için bilgisayar grafikleri, bilgisayar görüşü ve coğrafi bilgi sistemlerinde yaygın olarak kullanılır. Morton Z-ordercurve, verimli uzamsal sorgulamalar ve veri alımı sağlamak için dörtlü ağaçlar ve sekizli ağaçlar gibi uzamsal indeksleme yapılarında yaygın olarak kullanılır (Cozzi ve Lilley, 2022).

Şekil 24 ve Şekil 26'da bulunan Z-sıralama yapısı tez kapsamındaki veri kümesi için de uygulanmıştır. Tilejson dosyası oluşturulurken Morton Z-ordercurve doldurma eğrisi uygulanmıştır. Böyleye oluşturulan ağaç yapısında veri kaybı yaşanması önlenmiştir.

Morton Z-ordercurve, veri kümesindeki her noktanın x ve y'e koordinatlarının ikili temsillerini serpiştirerek çalışır. Bu durum noktalar arasındaki uzamsal ilişkiyi temsil eden doğrusal bir bit dizisiyle sonuçlanır (Lee ve dig., 2007). Ortaya çıkan dizi, orijinal iki boyutlu uzaydaki komşu noktaların tek boyutlu dizide muhtemelen birbirine yakın olacak şekilde sıralanır, bu da verimli veri alımını kolaylaştırır.

4.2. Grafik Kitaplığı İletim Formatı (GLTF)

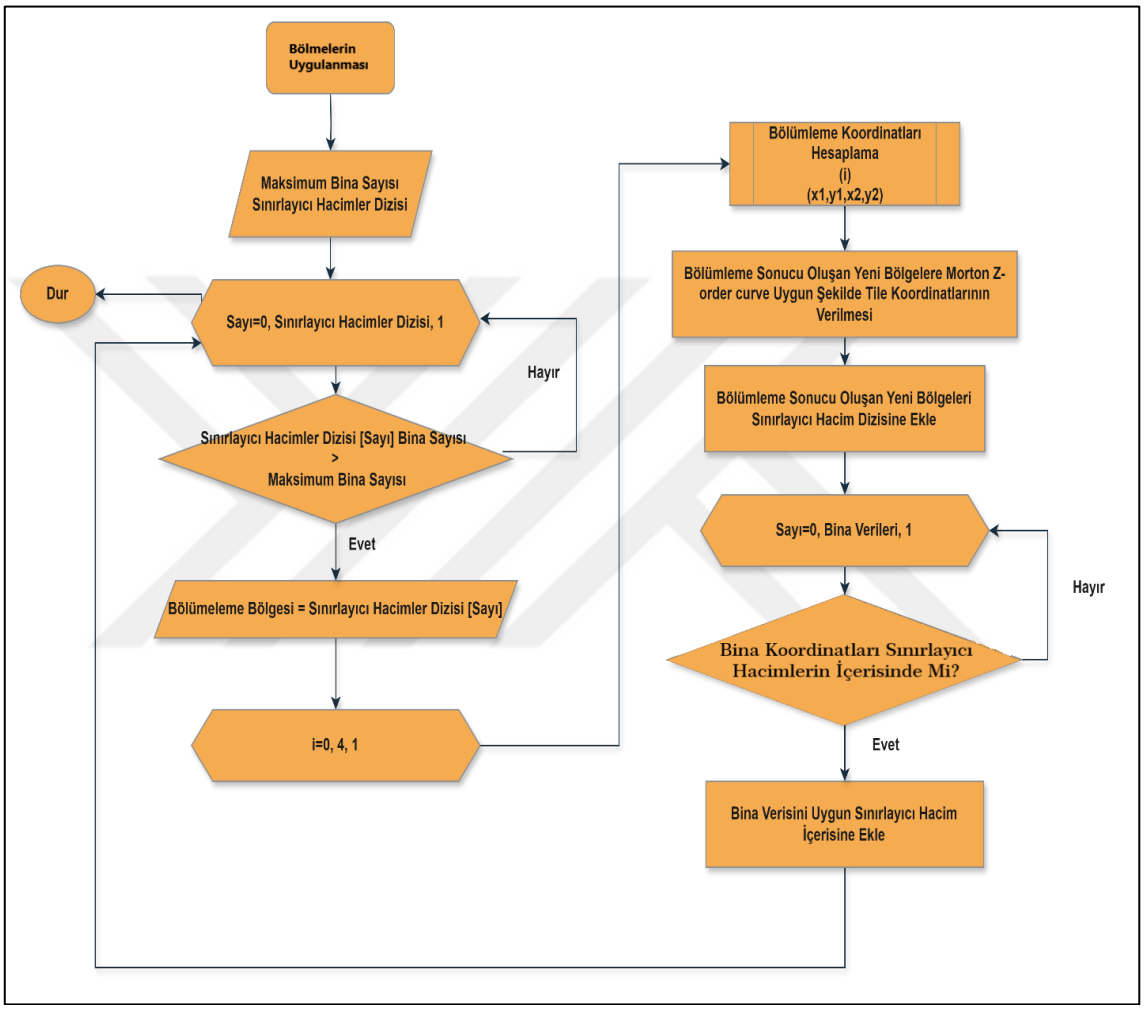
Şekil 14'de belirtilen sürecin çıktı ürünlerinden biri olan GLTF formatlı dosya 3D modeli gösterim için kullanılacak olan Cesium tarafından okunmaktadır.

GLTF, 3D modelleri ve sahneleri tanımlayan bir açık standart dosya formatıdır. JSON tabanlı bir format kullanır, bu da dosyaların kullanıcı tarafından okunabilir ve düzenlenebilir olmasını sağlar. 3D grafik uygulamaları, web tarayıcıları ve çeşitli platformlarda kullanılmak üzere tasarlanmıştır (Schillinget ve dig., 2016). GLTF dosyaları, 3D modeller, malzemeler, kameralar ve sahneler gibi öğeleri içerebilir. Ayrıca, veri sıkıştırması yaparak büyük 3D modelleri optimize eder. WebGL ile uyumlu olacak şekilde tasarlanmıştır ve özellikle web tarayıcılarında kullanım için optimize edilmiştir. GLTF, renk, yansıma, yarı saydamlık gibi malzeme özelliklerini, görsel detayları ifade eder ve animasyonları destekler. Modüler bir tasarımı vardır ve uzantıları destekler, bu da yeni özelliklerin eklenmesini kolaylaştırır (Usta, 2021). 3D grafiklerin tarayıcılarda veya uygulamalarda kullanılması, paylaşılması ve depolanması için yaygın olarak kullanılan bir formattır.

4.3. Önerilen Algoritmaların Gerçekleştirimi

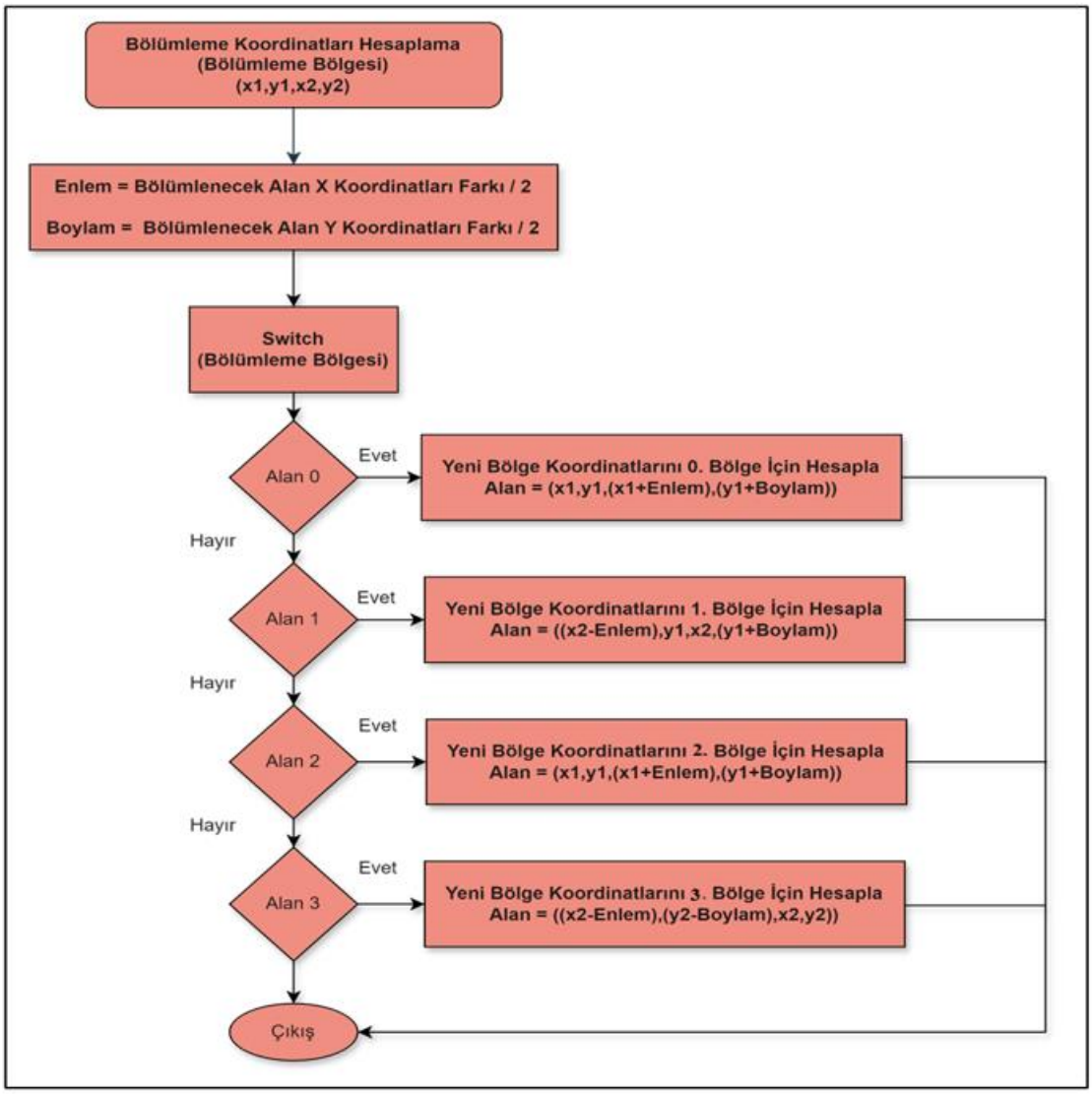
4.3.1. QuadTree algoritması

QuadTree algoritması bölümlemelerin uygulandığı ve bölümleme koordinatlarının bulunduğu iki temel fonksiyondan oluşmaktadır. Şekil 27' de bölümlemelerin uygulandığı fonksiyonun akış diyagramı bulunmaktadır.



Şekil 27. Bölümlemelerin uygulandığı fonksiyonun akış diyagramı

Bölümlemelerin uygulandığı fonksiyon, bir bölgeyi dört alt bölgeye böler ve her alt bölgeye ait verileri düzenler. İşlemler arasında, bölgenin belirli bir alt bölgeye bölünüp bölünmediğini kontrol etme, uygun verileri ekleyerek ve çıkararak düzenleme ve son olarak bölgeyi dört alt bölgeye ayırdıktan sonra belirli koşullar altında ilgili verileri taşıma bulunmaktadır. Bu iterasyon her bir bölgenin içerisine düşen obje sayısı maksimum bina sayısından küçük oluncaya kadar devam eder. Şekil 28'de bölümleme fonksiyonunun bir alt programı da olan bölümleme koordinatlarının bulunduğu fonksiyonun akış diyagramı görülmektedir.

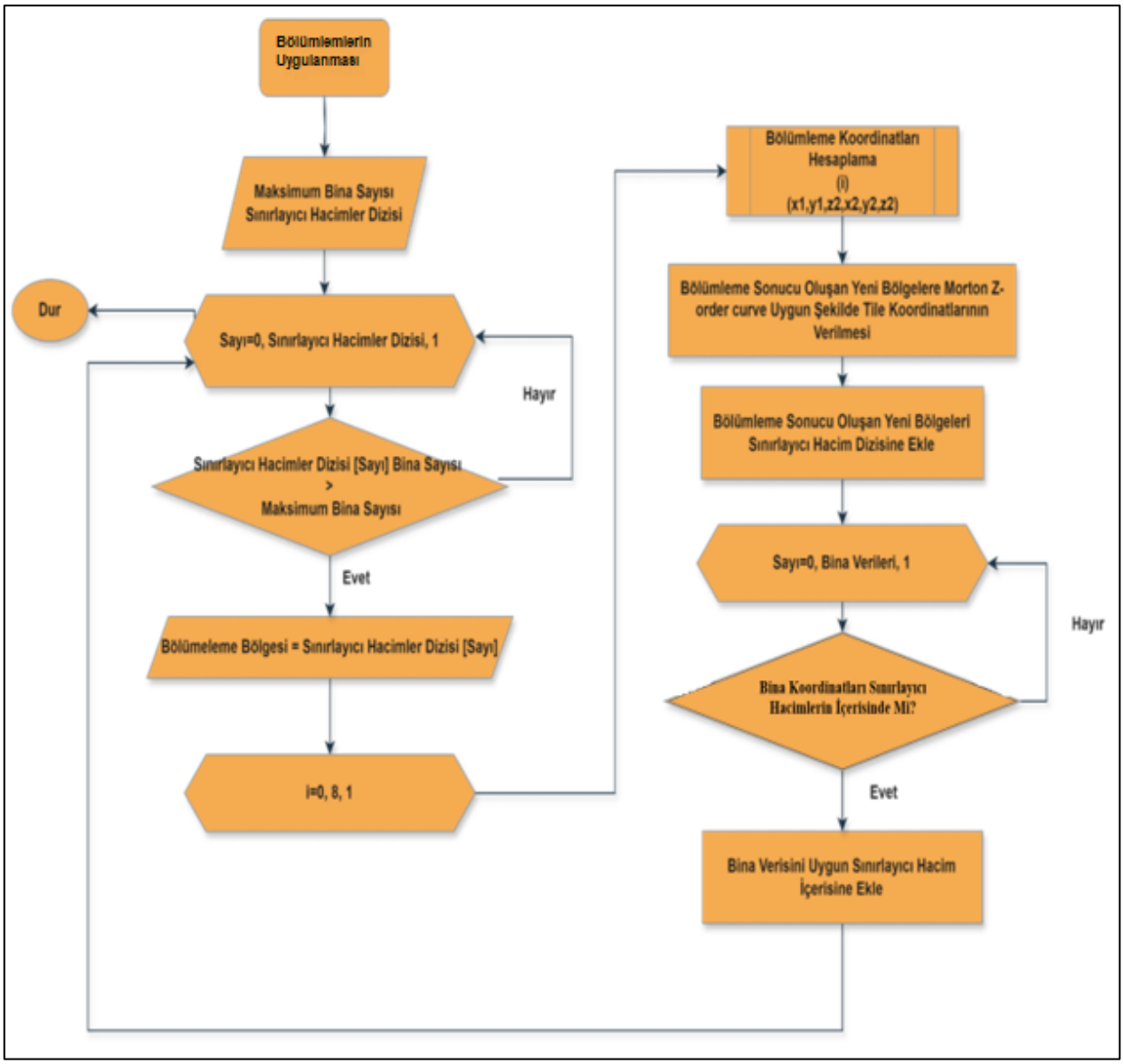


Şekil 28. Bölümleme koordinatlarının hesaplandığı fonksiyonun akış diyagramı

Bölümleme koordinatlarının bulunduğu fonksiyon ise bir bölgeyi dört alt bölgeye bölerek, her bir alt bölgeyi temsil eden yeni karoların koordinatlarını bulur. Bu işlem, mevcut bölgenin içerisine düşen objelerin x ve y değerlerinin ortalaması bulunarak her alt bölgeye ait koordinatlar belirlenir. Fonksiyon, bu dört alt bölge için yeni karo nesneleri oluşturur ve bunları döndürür. Eğer geçerli bir durum yoksa boş değer döndürülür.

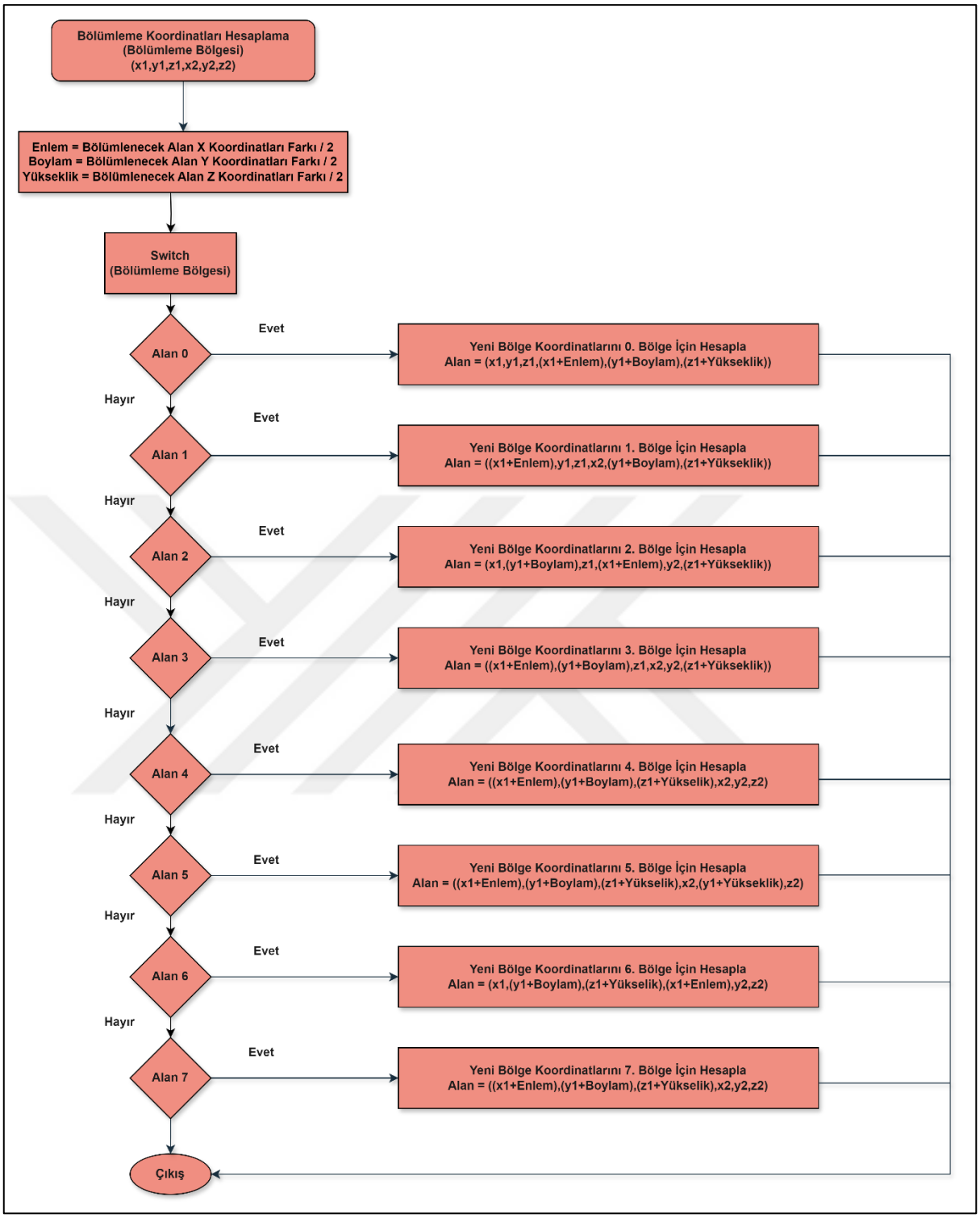
4.3.2. Octree algoritması

QuadTree algoritması bölümlemelerin uygulandığı ve bölümleme koordinatlarının bulunduğu iki temel fonksiyondan oluşmaktadır. Şekil 29'da bölümlemelerin uygulandığı fonksiyonun akış diyagramı bulunmaktadır.



Şekil 29. Octree bölümlemelerin uygulandığı fonksiyonun akış diyagramı

Bölümlemelerin uygulandığı fonksiyon, bir bölgeyi sekiz alt bölgeye böler ve her alt bölgeye ait verileri düzenler. İşlemler arasında, bölgenin belirli bir alt bölgeye bölünüp bölünmediğini kontrol etme, uygun verileri ekleyerek ve çıkararak düzenleme ve son olarak bölgeyi sekiz alt bölgeye ayırdıktan sonra belirli koşullar altında ilgili verileri taşıma bulunmaktadır. Bu iterasyon her bir bölgenin içerisine düşen obje sayısı maksimum bina sayısından küçük oluncaya kadar devam eder. Bölümleme koordinatlarının bulunduğu fonksiyonun akış şeması Şekil 30' da görülmektedir.

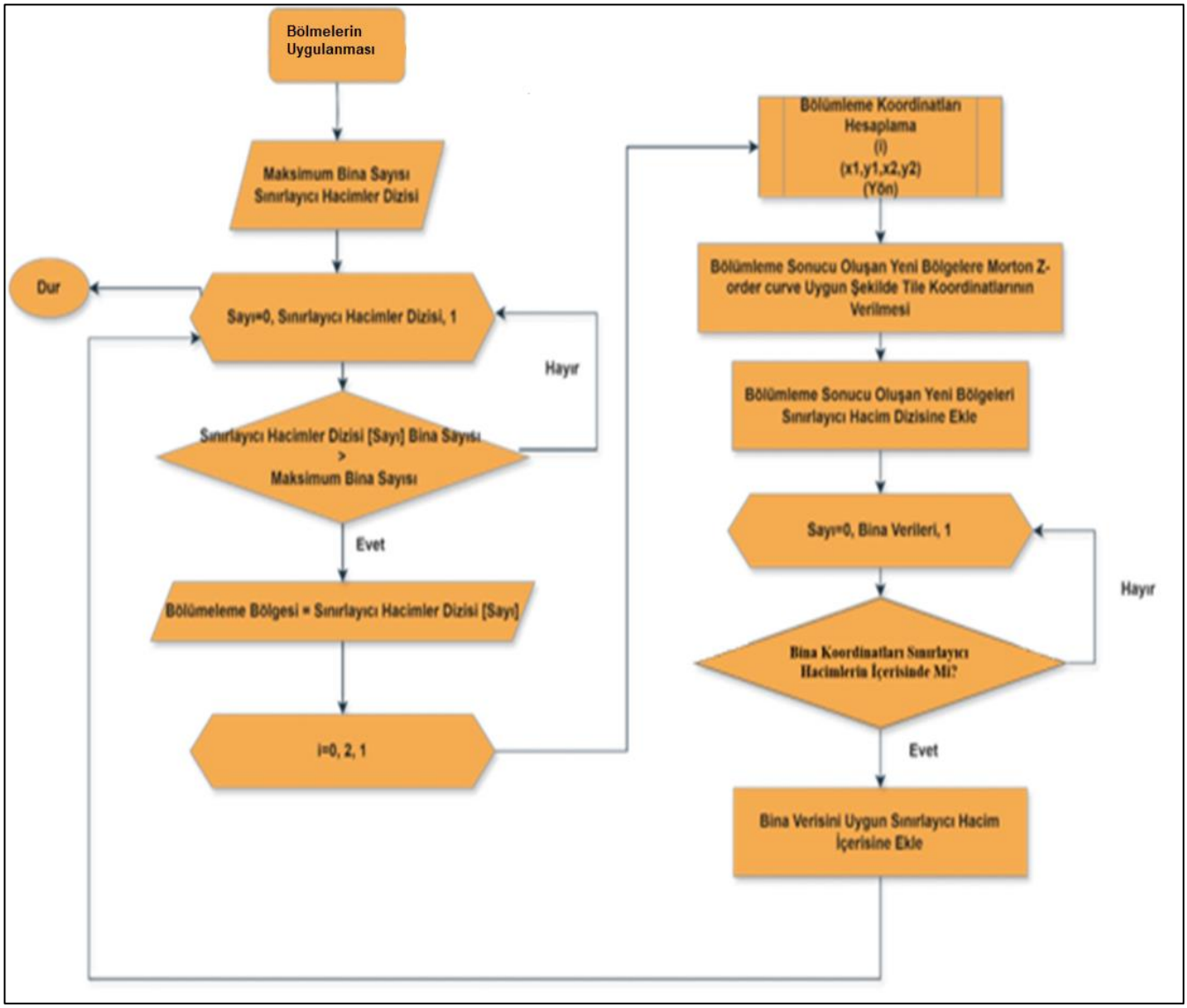


Şekil 30. Octree için bölümleme koordinatlarının hesaplandığı fonksiyonun akış diyagramı

Bölümleme koordinatlarının bulunduğu fonksiyon, bir bölgeyi sekiz alt bölgeye bölerek her bir alt bölgeyi temsil eden yeni bölgenin koordinatlarını bulur. Bu işlem, mevcut bölgenin x, y ve z değerlerinin orta noktaları bulunarak her alt bölgeye ait koordinatlar hesaplanır. Fonksiyon, bu sekiz alt bölge için yeni karo nesneleri oluşturur ve bunları döndürür. Eğer geçerli bir durum yoksa boş değer döndürülür.

4.3.3. Kd-Tree algoritması

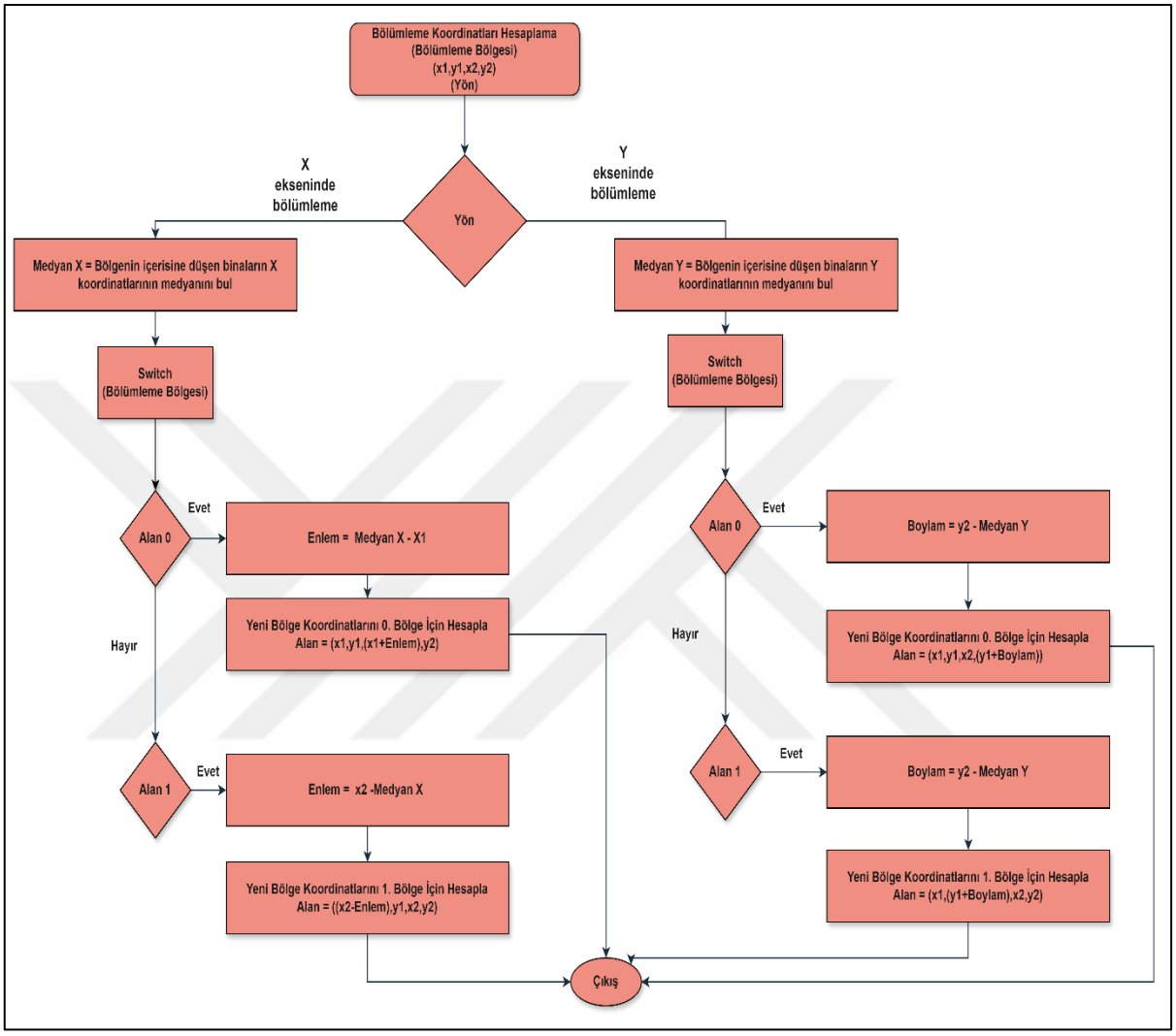
Kd-Tree algoritması, bölümlemelerin uygulandığı ve bölümleme koordinatlarının bulunduğu iki temel fonksiyondan oluşmaktadır. Şekil 31'de bölümleme fonksiyonun işlem adımları görülmektedir.



Şekil 31. Kd-Tree algoritmasına göre bölümlemelerin uygulandığı fonksiyonun akış diyagramı

Bu fonksiyon, bir bölgeyi iki alt bölgeye böler ve her alt bölgeye ait verileri düzenler. İşlemler arasında, bölgenin belirli bir alt bölgeye bölünüp bölünmediğini kontrol etme, uygun verileri ekleyerek ve çıkararak düzenleme ve son olarak bölgeyi iki alt bölgeye ayırdıktan sonra belirli koşullar altında ilgili verileri taşıma bulunmaktadır. Bu iterasyon her bir bölgenin içerisine düşen obje sayısı maksimum bina sayısından küçük oluncaya kadar devam eder.

Bölümleme koordinatlarının hesaplandığı fonksiyon, bölümleme yapılacak bölgeyi 2 alt bölgeye bölerek her bir alt bölgeyi temsil eden yeni bölgenin koordinatlarını bulur. Fonksiyon akış diyagramı Şekil 32'de görülmektedir.



Şekil 32. Kd-Tree için bölümleme koordinatlarının hesaplandığı fonksiyonun akış diyagramı

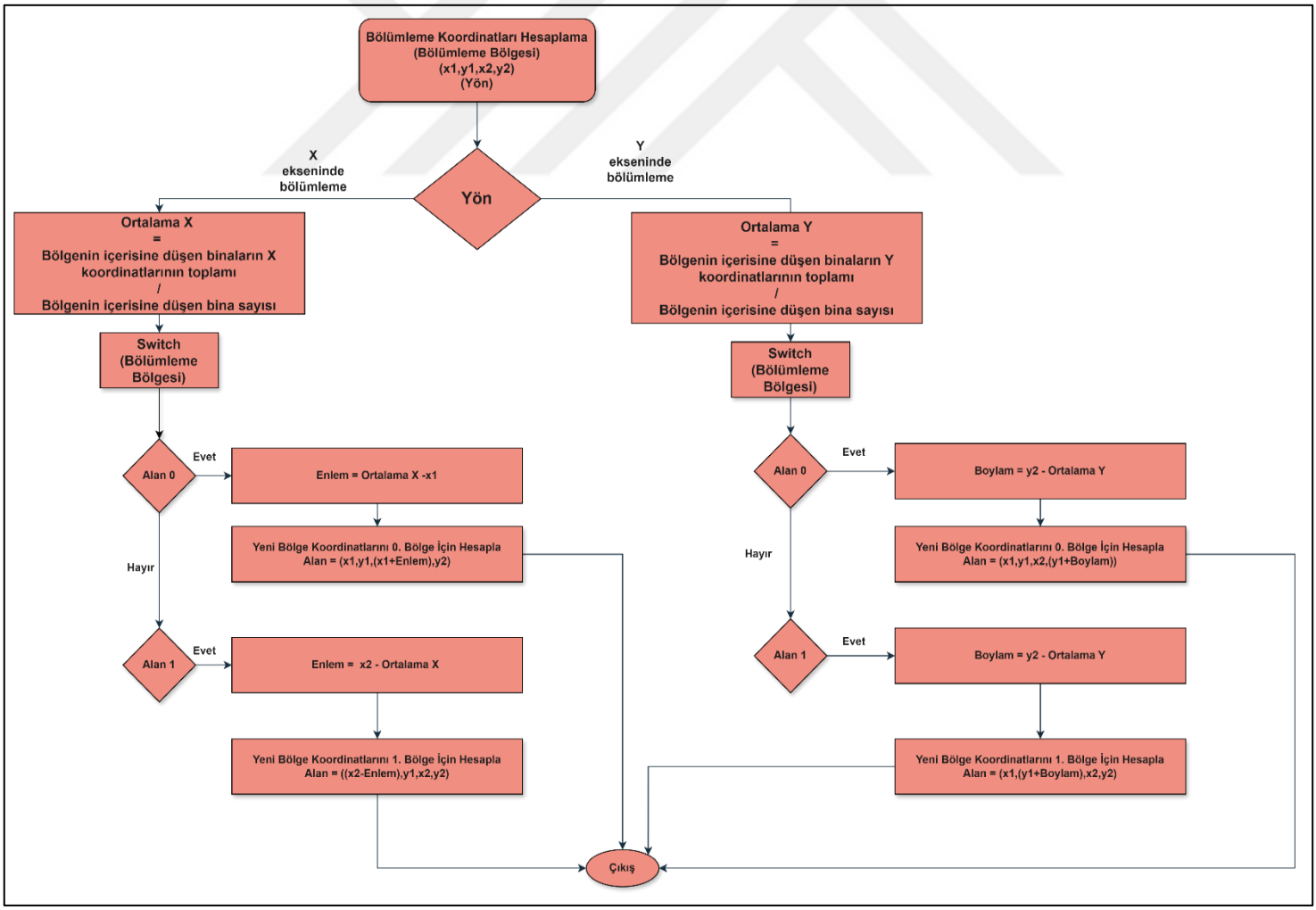
Kd-Tree algoritmasında bölümleme yapılacak noktaların bulunması, mevcut bölgenin içerisine düşen objelerin x ve y değerlerinin medyan değeri hesaplanarak yapılır. Bölme işlemi her iterasyonda yön değiştirerek uygulanmaktadır. Örneğin ilk iterasyonda x ekseni yönünde bölümleme gerçekleştirilecekse bölümleme yapılacak bölge içerisine düşen objelerin x koordinatlarının medyanı hesaplanarak bölümlenecek koordinatlar hesaplanır ve x ekseninde bölümleme gerçekleştirilerek 2 alt bölgeye ayrılır. Bir sonraki işlem adımında oluşan 2 bölge için ayrı ayrı olarak y ekseni yönünde bölümleme yapılacak şekilde bölgenin içerisine düşen objelerin y koordinatlarının

meydanları hesaplanır ve y ekseninde bölümleme gerçekleştirilir. Bu döngü elde edilen her bir bölgenin obje sayısı maksimum bina sayısını aşmayana kadar devam ettirilir.

4.3.4. R-Tree algoritması

R-Tree algoritmasının çalışma prensibi Kd-Tree algoritmasına benzemektedir. Bölümlemelerin uygulandığı fonksiyonun akış diyagramı Şekil 31'deki Kd-Tree ile aynıdır. Bölge iki alt bölgeye bölünür ve her alt bölgeye ait veriler düzenlenir. İşlemler arasında, bölgenin belirli bir alt bölgeye bölünüp bölünmediğini kontrol etme, uygun verileri ekleyerek ve çıkararak düzenleme ve son olarak bölgeyi iki alt bölgeye ayırdıktan sonra belirli koşullar altında ilgili verileri taşıma bulunmaktadır. Bu iterasyon her bir bölgenin içerisine düşen obje sayısı maksimum bina sayısından küçük oluncaya kadar devam eder.

R-Tree algoritmasının Kd-Tree algoritmasından ayıran fark bölümleme koordinatlarının bulunduğu fonksiyonda gerçekleşmektedir. Şekil 33'de bu fonksiyonun akış diyagramı görülmektedir.



Şekil 33. R-Tree için bölümleme koordinatlarının hesaplandığı fonksiyonun akış diyagramı

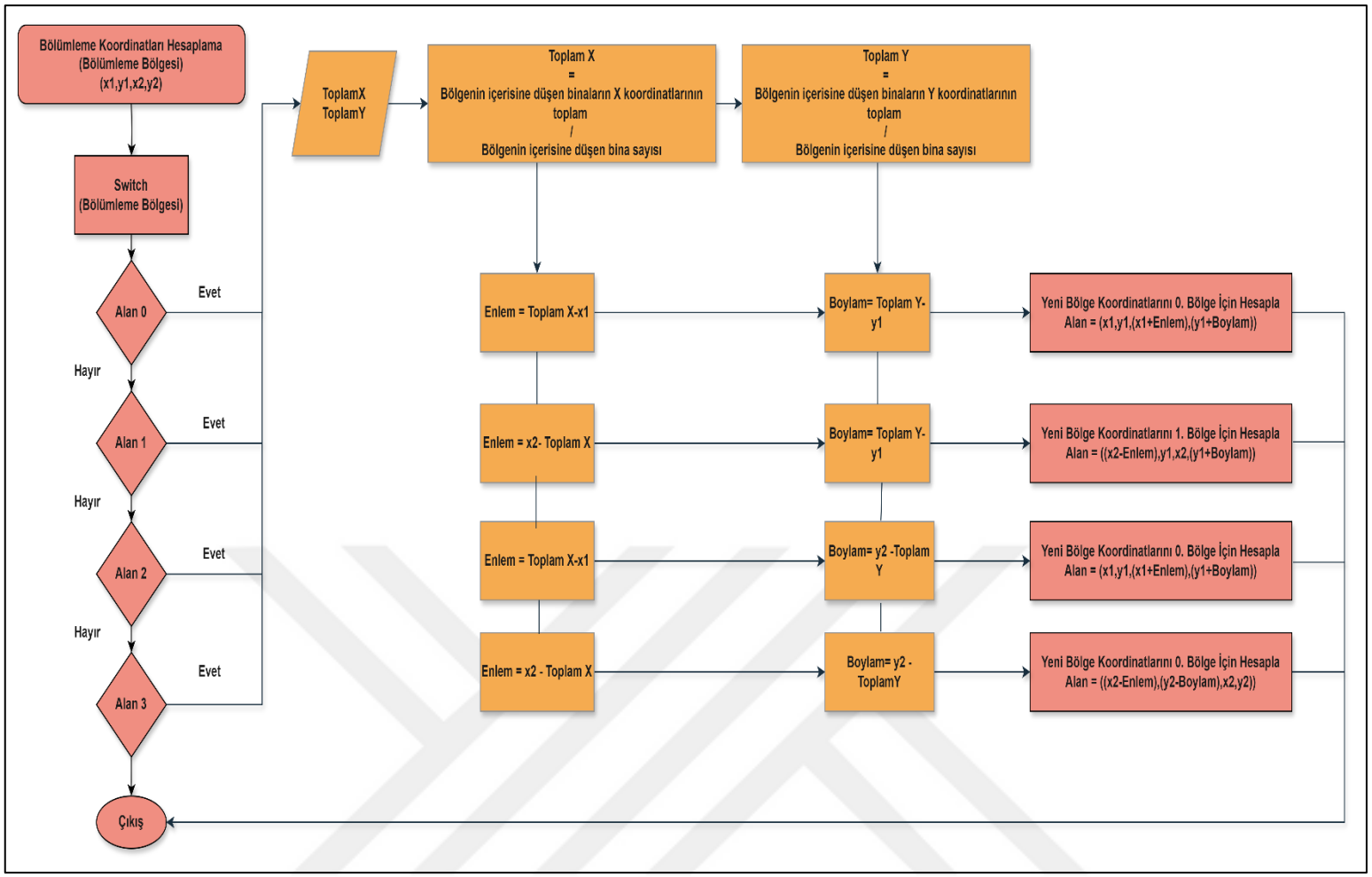
R-Tree algoritmasında bölümleme yapılacak noktalar, mevcut bölgenin içerisine düşen objelerin x ve y değerlerinin ortalaması alınarak hesaplanır. Bölme işlemi Kd-Tree algoritmasında olduğu gibi her iteresyonda yön değiştirerek uygulanmaktadır. Bu döngü elde edilen her bir bölgenin obje sayısı maksimum bina sayısını aşmayana kadar devam ettirilir.

4.3.5. Önerilen Optimize QuadTree algoritması

Mevcut olan bölümleme algoritmalarının yanı sıra daha optimal bir bölümleme yapacağını düşündüğümüz Optimize QuadTree, temelinde bölümleme algoritmalarıyla aynı prensibe göre çalışmaktadır.Tek fark olarak bölümleme esnasında her bir sınırlayıcı hacimin içerisine düşen nokta sayılarının ortalamasını alarak bölümleme yapılacak koordinatı bulmaktadır.

Önerilen algoritmamızda da bölümlemelerin uygulandığı ve bölümleme koordinatlarının bulunduğu iki temel fonksiyondan oluşmaktadır. Temel Bölümlemelerin uygulandığı fonksiyon, bir bölgeyi dört alt bölgeye bölerek her bir alt bölgeyi temsil eden yeni karoların koordinatlarını bulur. QuadTree algoritmasının bölümleme uygulaması fonksiyonuyla aynı çalışmaktadır. Şekil 27’de olduğu gibi gerçekleşen işlem adımları arasında bölgenin belirli bir alt bölgeye bölünüp bölünmediğini kontrol etme, uygun verileri ekleyerek ve çıkararak düzenleme ve son olarak bölgeyi dört alt bölgeye ayırdıktan sonra belirli koşullar altında ilgili verileri taşıma bulunmaktadır. Bu iterasyon her bir bölgenin içerisine düşen obje sayısı maksimum bina sayısından küçük oluncaya kadar devam eder.

Optimize QuadTree algoritmasını, QuadTree algoritmasından ayıran ve daha optimal sonuçlar elde edilmesini sağlayan fark bölümleme yapılacak noktaların bulunması fonksiyonudur. Şekil 34’de bölümleme koordinatlarının hesaplandığı fonksiyonun akış diyagramı bulunmaktadır.



Şekil 34. Bölümleme koordinatlarının hesaplandığı fonksiyonun akış diyagramı

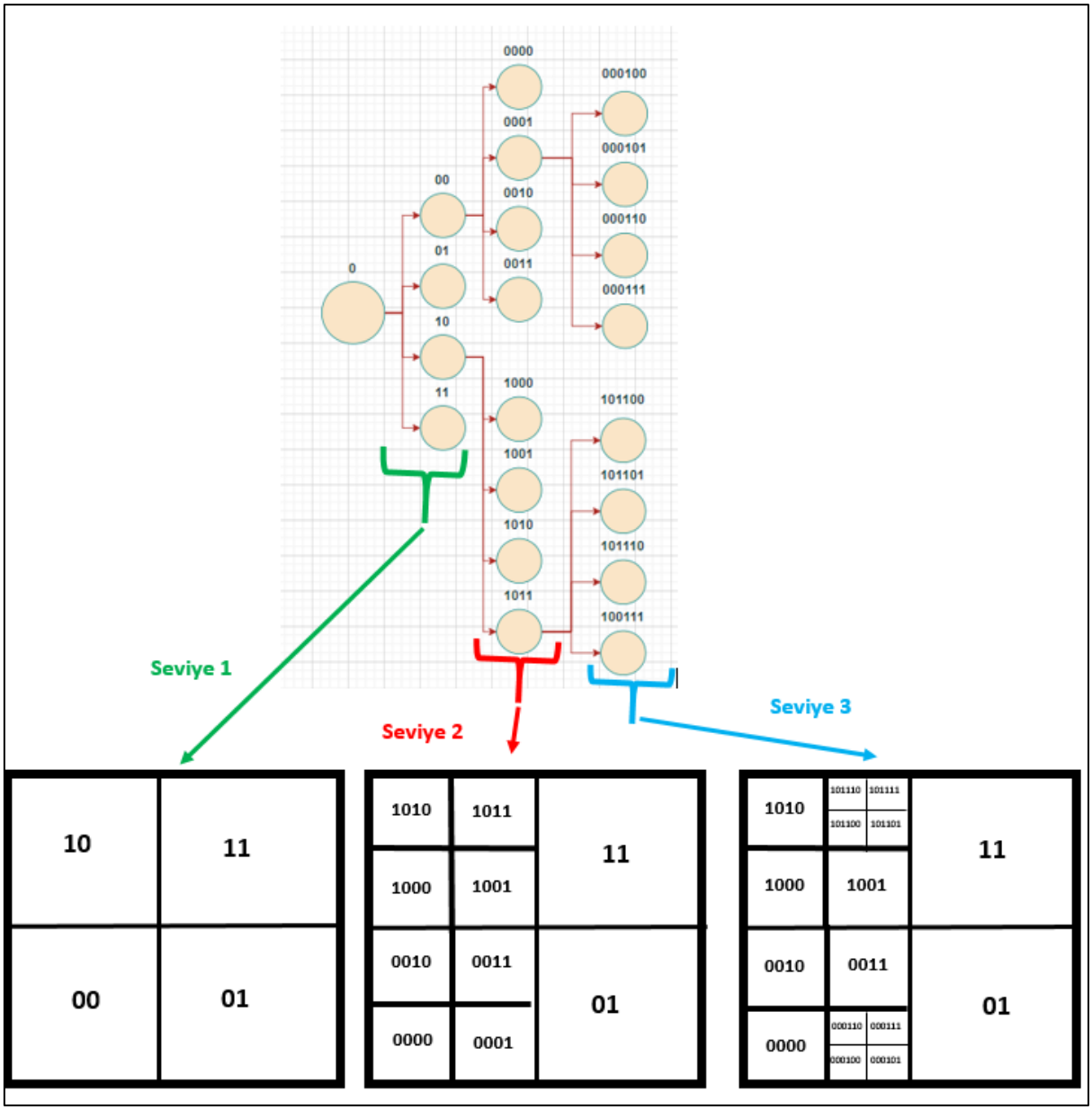
Mevcut bölgenin içerisine düşen objelerin x ve y değerlerinin ortalaması bulunarak hesaplanır. Bölgenin içerisine düşen objelere göre bölümleme yapılacak noktaların belirlenmesi, veri yoğunluğunun olduğu bölgede karo sayısının artıracağı ve daha doğru sonuçlar elde edileceği varsayılmaktadır. Bu döngü elde edilen her bir bölgenin obje sayısı maksimum bina sayısını aşmayana kadar devam ettirilir.

Optimize QuadTree algoritması, büyük ölçekli kentsel 3D modellerin daha verimli ve etkili şekilde bölünmesine olanak tanır. Geleneksel QuadTree algoritmasına kıyasla, veri yoğunluğu olan bölgelerde daha detaylı bölümleme yaparak, görselleştirme ve veri yönetimi performansını artırır. Algoritmanın temel yeniliği, bölümleme yapılacak koordinatların belirlenmesinde kullanılan ortalama nokta sayısı yaklaşımıdır. Bu yenilikçi yaklaşım, yoğun veri kümeleri için daha hassas bölümleme ve daha yüksek performans sunarak, mimarlık, şehir planlama, jeo-uzamsal analiz, artırılmış gerçeklik ve birçok başka alanda yapılan çalışmalara önemli katkılar sunar.

5. ARAŞTIRMA SONUÇLARI VE TARTIŞMA

5.1. QuadTree Algoritmasına Göre Elde Edilen Sonuçlar

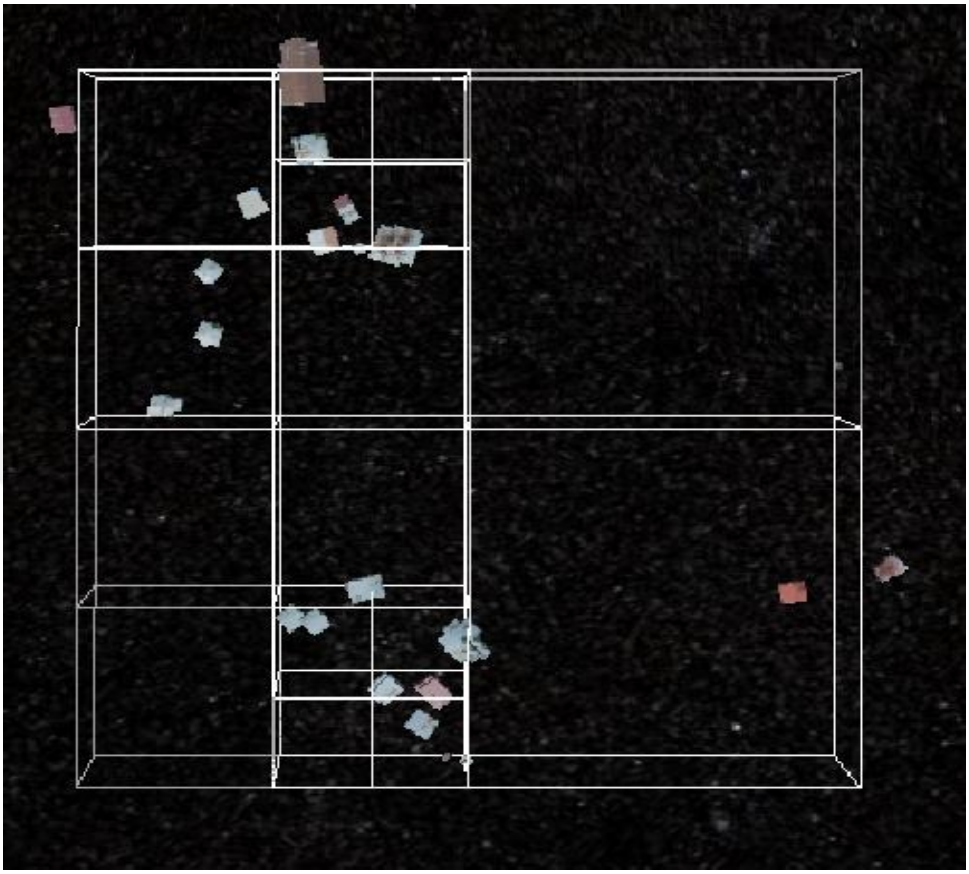
Hakkari Biçer Mahallesinden seçilen 21 bina verisi QuadTree algoritmasına göre bölümlenmiş ve oluşan yapının tile kooordinatları oluşturulmuştur. Morton indekleri hesaplanarak elde edilen 3D Tiles ağaç yapısı ve yerleştirimi Şekil 35'de görülmektedir.



Şekil 35. QuadTree algoritmasına göre çalışma alanının bölümleme sonucu elde edilen yapısı

21 bina için sınırlayıcı hacim içerisinde maksimum bina sayısı 5 olarak belirtilmiştir. Bu kısıtlamalara göre 3 seviyeli bir ağaç yapısı elde edilmiş olup elde edilen sınırlayıcı hacimlerin tile koordinartları belirlenmiştir. Tile koordinarlarının veriliş sıralaması ise Morton Z-ordercurve doldurma eğrisi temel alınarak oluşturulmuştur. Böylece oluşan GLTF dosyaları ile tilese.json dosyası sorunsuz bir

şekilde haberleşme gerçekleştirebilmiştir. Şekil 36'da 21 bina için QuadTree algoritması uygulanarak oluşturulmuş ve tilenmış verinin 3 boyutlu olarak uzayda bina verileride yerleştirelerek oluşturulmuş görüntüsü görülmektedir.

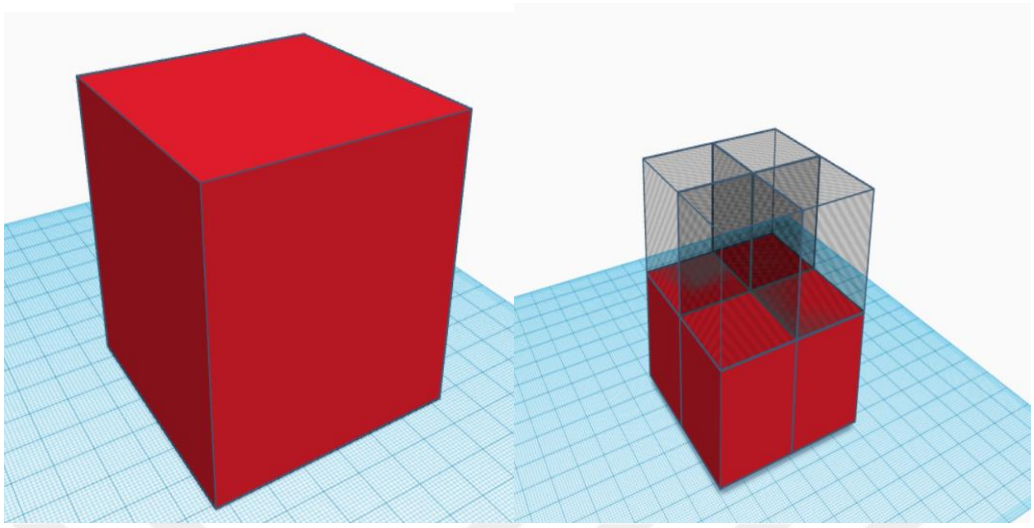


Şekil 36. QuadTree algoritmasına göre Biçer Mahallesinin sonuç görüntüsü

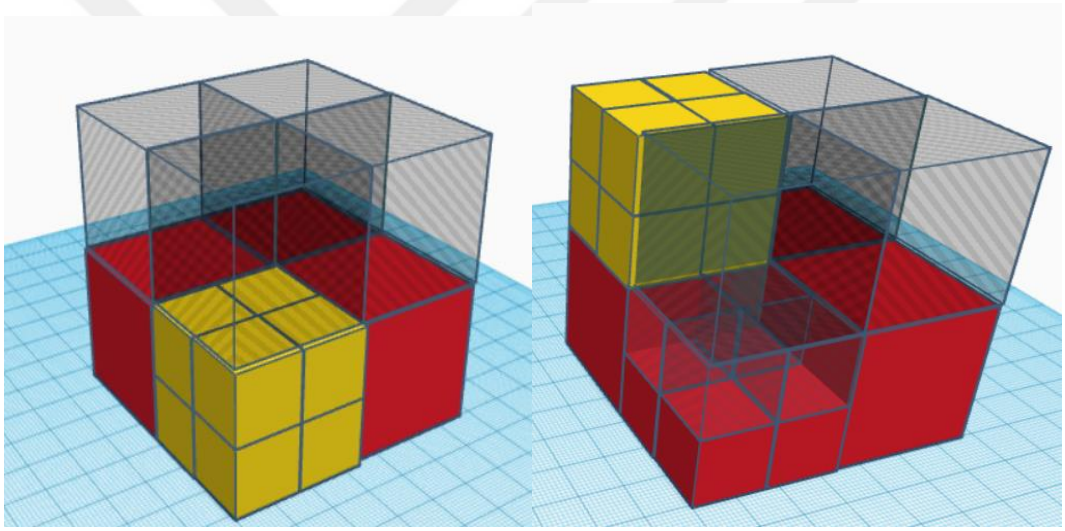
5.2. Octree Algoritmasına Göre Elde Edilen Sonuçlar

Octree algoritmasının diğer bölümleme algortimlarına nazaran en büyük farkı z indeksinin de bölümleme işlemine dahil edilmesidir. Şekil 37'de, Hakkari Biçer Mahallesinden seçilen 21 bina için Octree algoritmasına göre maksimum bina sayısının 5 olarak belirlendiği ve sınırlayıcı hacimlerin bulunduğu işlem adımları görülmektedir. Görsellerde bulunan sarı alanlar o an ki işlem adımında bölümleme gerçekleştirilen alanı temsil etmektedir.

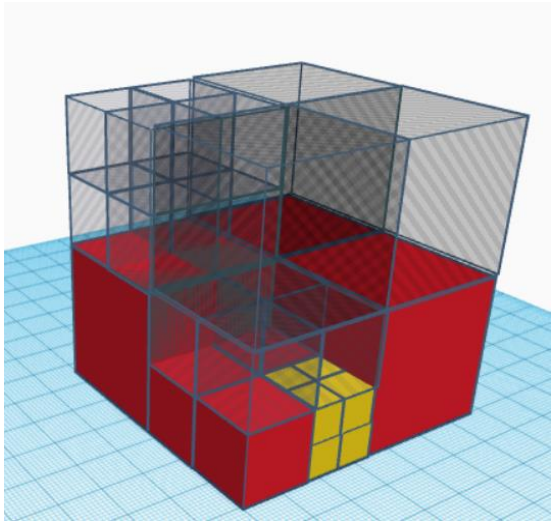
Şekil 37. Octree algoritmasına göre oluşan sınırlayıcı hacimler



(a) (b)

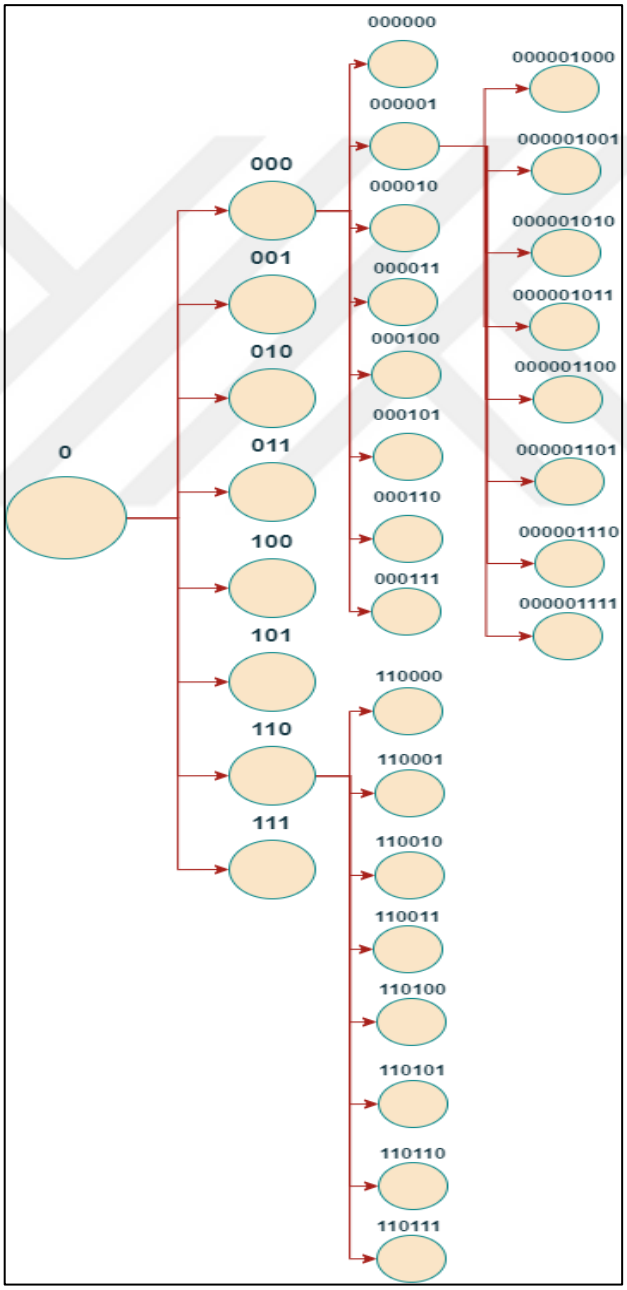


(c) (d)



(e)

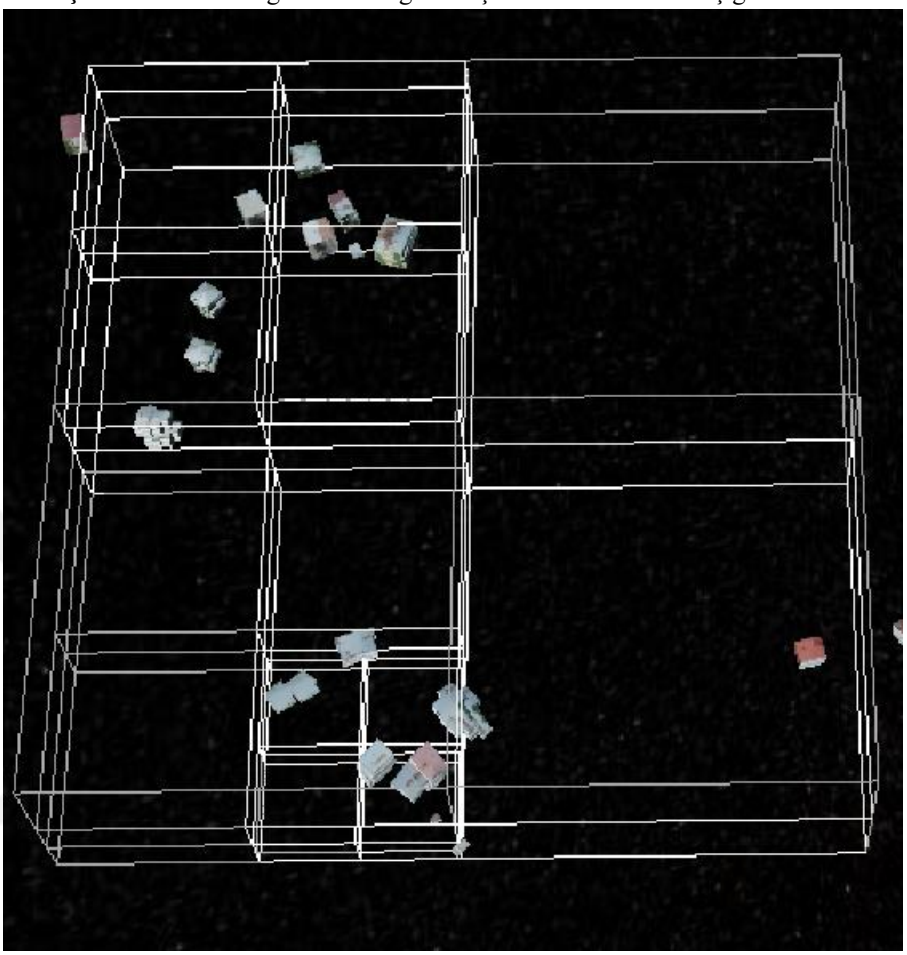
Gerçekleşen bölümleme sonucunda 3 seviyeli bir ağaç yapısı elde edildiği gözlemlenmiştir. Octree algoritmasına göre bölümleme işleminde z indekleride hesaba katıldığı için her sevide oluşan çocuk sayısı 8 adetdir. Elde edilen sınırlayıcı hacimlerin tile koordinartları belirlenmiştir. Tile koordinarlarının veriliş sıralaması ise Morton Z-ordercurve doldurma eğrisi baz alınarak oluşturulmuştur. Böylece oluşan GLTF dosyaları ile tilese.json dosyası sorunsuz bir şekilde haberleşme gerçekleştirebilmiştir. Oluşturulan SH'ler sonucunda elde edilen 3 seviyeli ağaç yapısı Şekil 38'de görülmektedir.



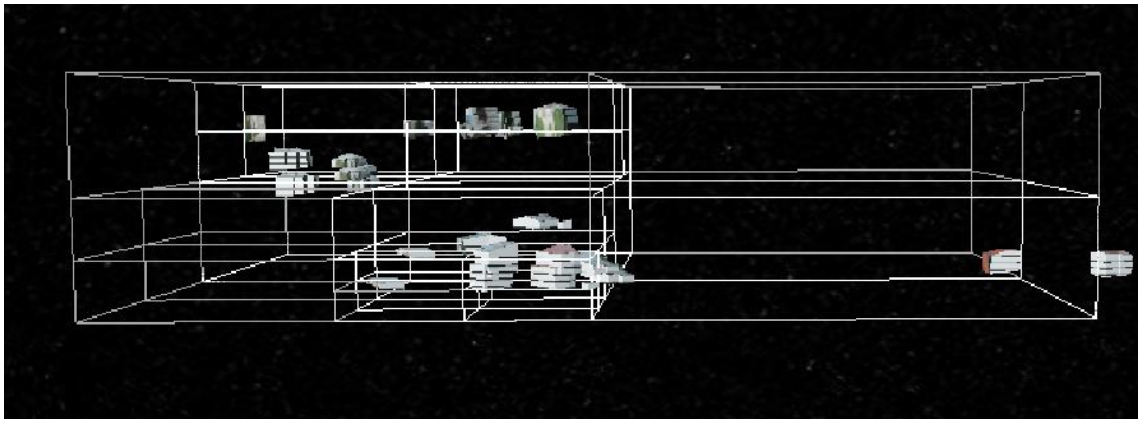
Şekil 38. Octree algoritmasına göre çalışma alanının bölümleme sonucu elde edilen ağaç yapısı

Şekil 39’da 21 bina için Octree algoritması uygulanarak oluşturulmuş tilenmış verinin 3 boyutlu olarak uzaydaki görüntüsü görülmektedir.

Şekil 39. Octree algoritmasına göre Biçer Mahallesinin sonuç görüntüsü



(a)

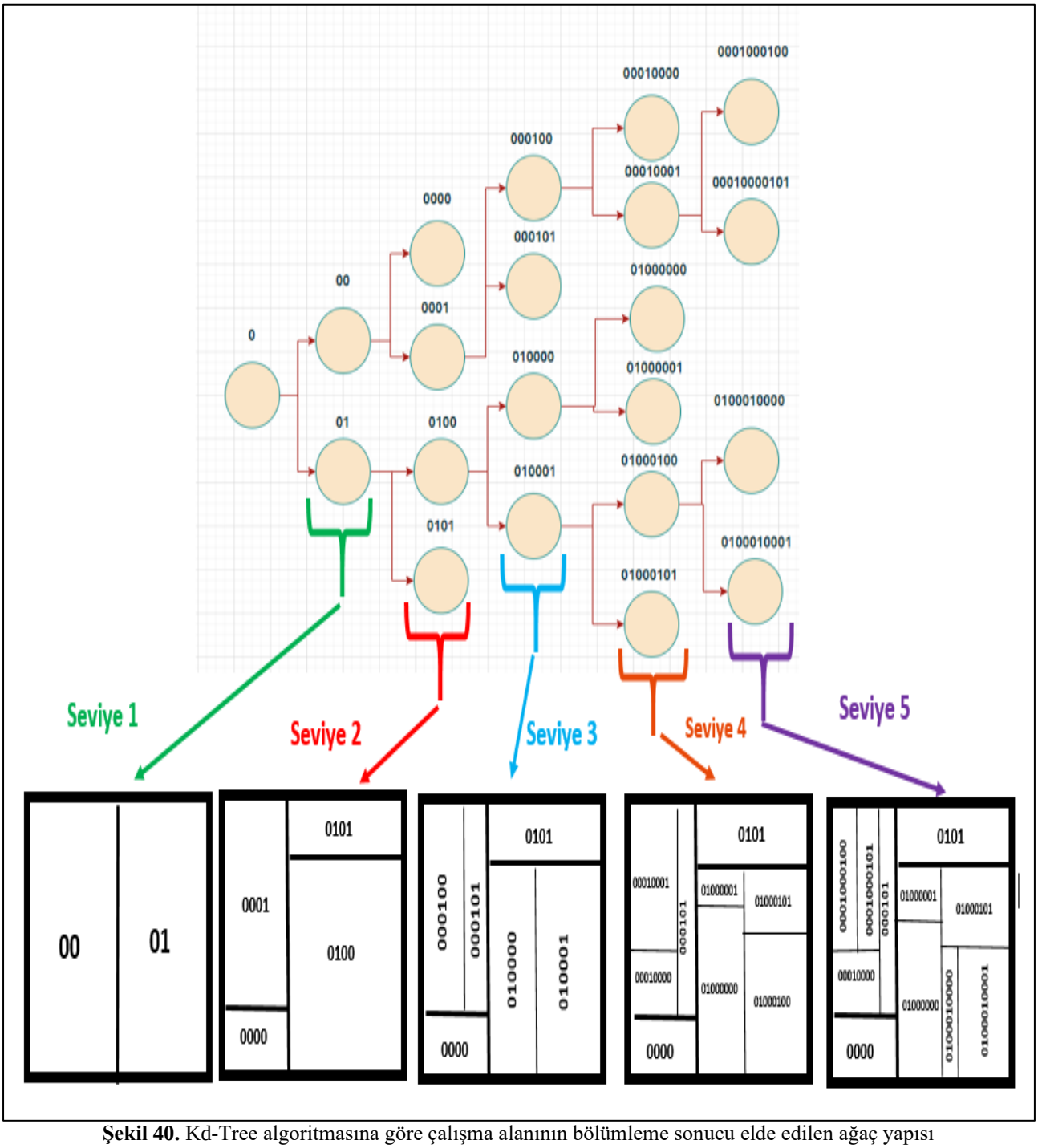


(b)

5.3. Kd-Tree Algoritmasına Göre Elde Edilen Sonuçlar

Hakkari Biçer Mahallesinden seçilen 21 bina versinin Kd-Tree algoritmasına göre maksimum bina sayısı 5 olarak belirlenerek bölümlenmiş ve oluşan yapının tile

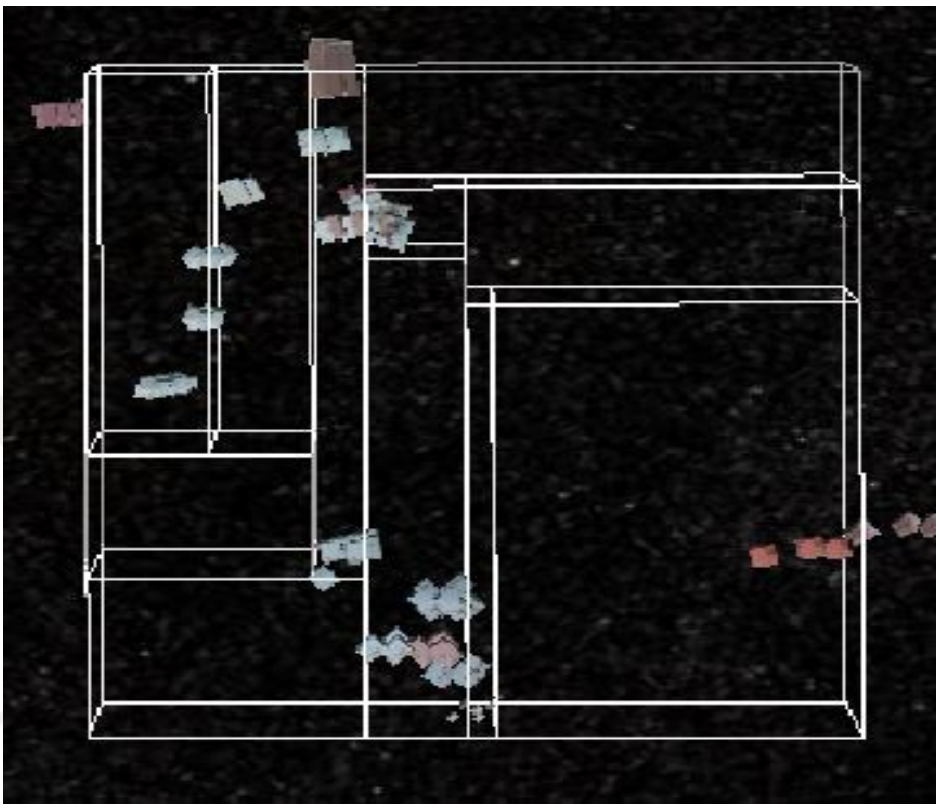
kooordinatları oluşturulmuştur. Tile koordinarlarının veriliş sıralaması ise Morton Z-ordercurve doldurma eğrisi temel alınarak oluşturulmuştur. Elde edilen 3D Tiles ağaç yapısı ve yerleştirimi Şekil 40'da görülmektedir.



Şekil 40. Kd-Tree algoritmasına göre çalışma alanının bölümleme sonucu elde edilen ağaç yapısı

Gerçekleşen bölümleme sonucunda 5 seviyeli bir ağaç yapısı elde edildiği gözlemlenmiştir. Diğer algoritmaların aksine Kd-Tree algoritmasına göre bölümleme işleminde her işlem adımında x ve ye yönünde olmak üzere iki yönlü bölümleme gerçekleştirilmez. Tüm alan önce x düzlemine göre bölünür, bir sonraki işlem adımda

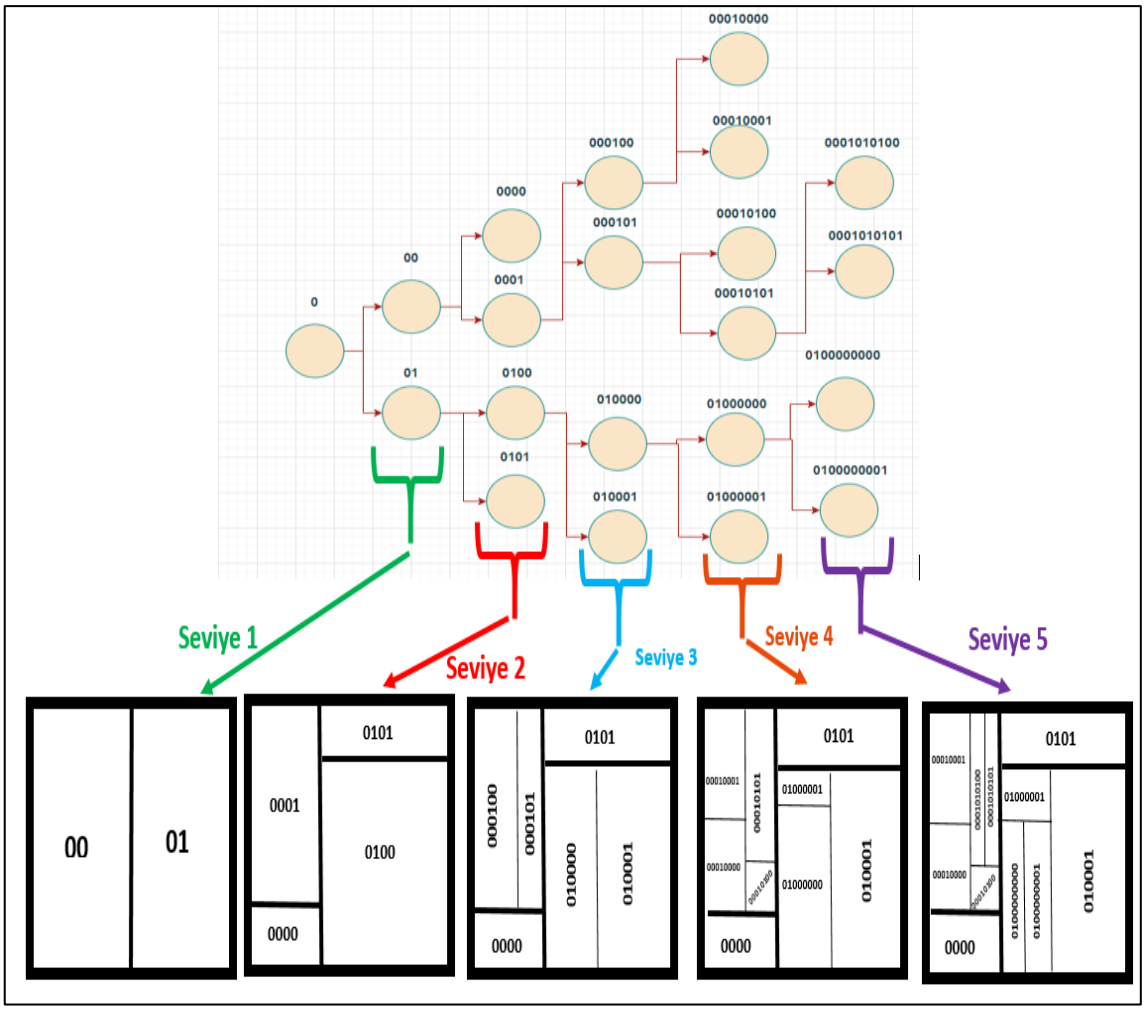
ise y düzlemine göre bölümleme gerçekleştirilir. Bu durumdan kaynaklı olarak her sevide oluşan çocuk sayısı 2 adetdir. Şekil 41'da 21 bina için Kd-Tree algoritması uygulanarak oluşturulmuş tile yapısına bina verileride yerleştirilmiş halinin 3 boyutlu uzaydaki görüntüsü görülmektedir.



Şekil 41. Kd-Tree algoritmasına göre Biçer Mahallesinin sonuç görüntüsü

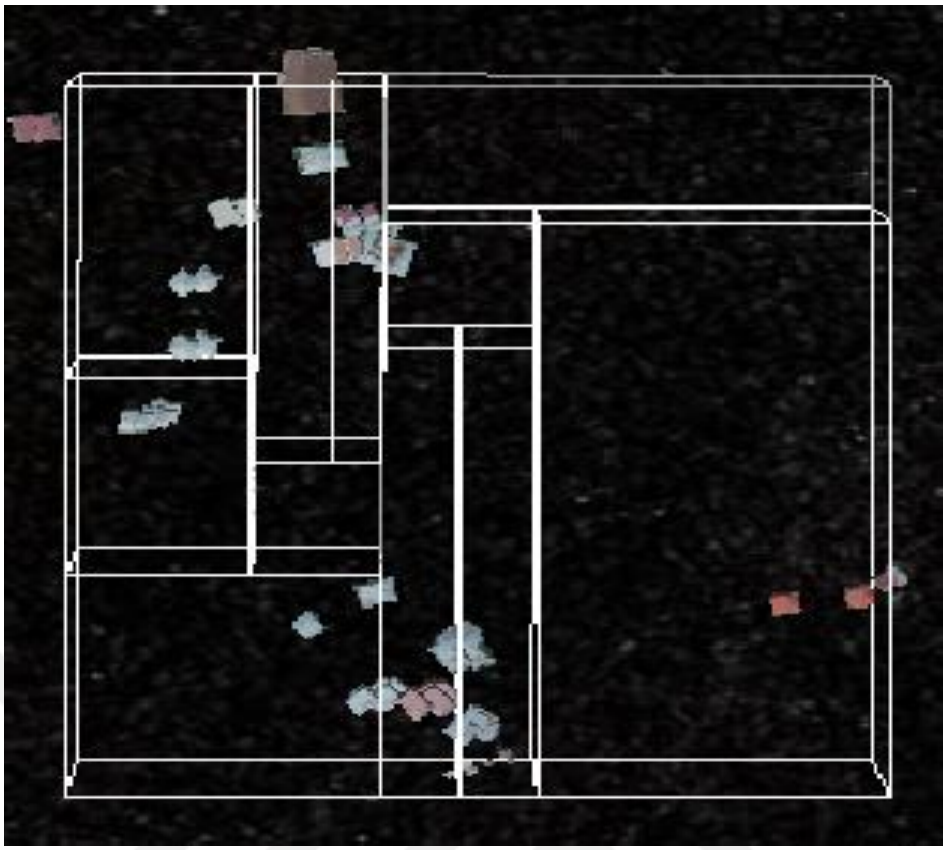
5.4. R-Tree Algoritmasına Göre Elde Edilen Sonuçlar

Hakkari Biçer Mahallesinden seçilen 21 bina versinin R-Tree algoritması uygulanarak elde edilen tilelardan oluşan ağaç yapısı Şekil 42'de görülmektedir.



Şekil 42. R-Tree algoritmasına göre çalışma alanının bölümleme sonucu elde edilen ağaç yapısı

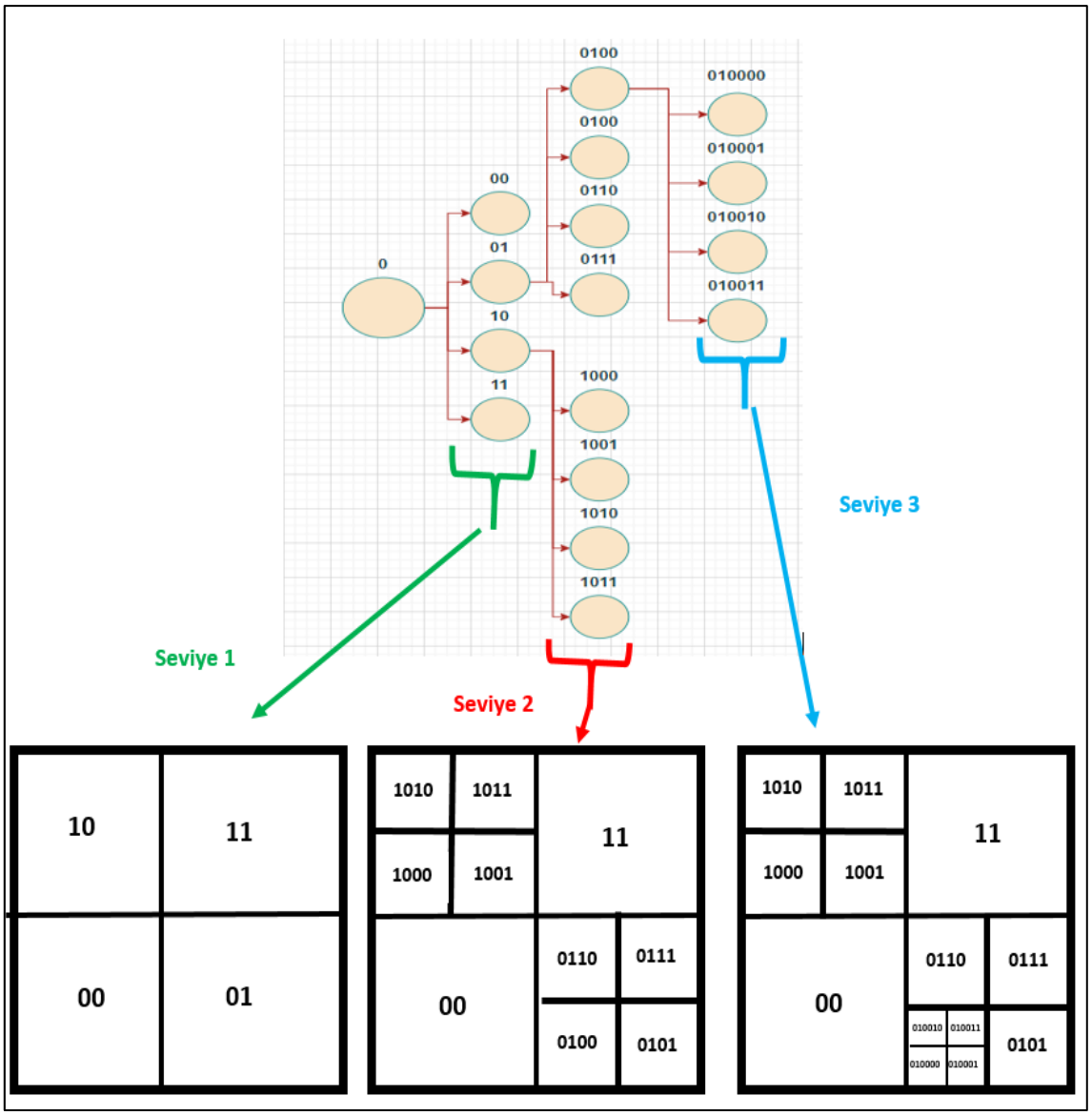
Gerçekleşen bölümleme sonucunda 5 seviyeli bir ağaç yapısı elde edildiği gözlemlenmiştir. Diğer algoritmaların aksine R-Tree algoritmasında da bölümleme işlemi her işlem adımında x ve ye yönünde olmak üzere iki yönlü bölümleme gerçekleştirilmez. Tüm alan önce x düzlemine göre bölünür, bir sonraki işlem adımda ise y düzlemine göre bölümleme gerçekleştirilir. Bu durumdan kaynaklı olarak her sevideoluşan çocuk sayısı 2 adetdir. Şekil 43'de 21 bina için R-Tree algoritması uygulanarak oluşturulmuş tilenmış verinin 3 boyutlu olarak uzaydaki görüntüsü görülmektedir.



Şekil 43. R-Tree algoritmasına göre Biçer Mahallesinin sonuç görüntüsü

5.5. Optimize QuadTree Algoritmasına Göre Elde Edilen Sonuçlar

Hakkari Biçer Mahallesinden seçilen 21 bina versinin önerilen Optimize QuadTree algoritmasına göre maksimum bina sayısı 5 belirlenerek bölümlenmiş ve oluşturulan SH'ler sonucunda elde edilen ağaç yapısı Şekil 44'de görülmektedir.



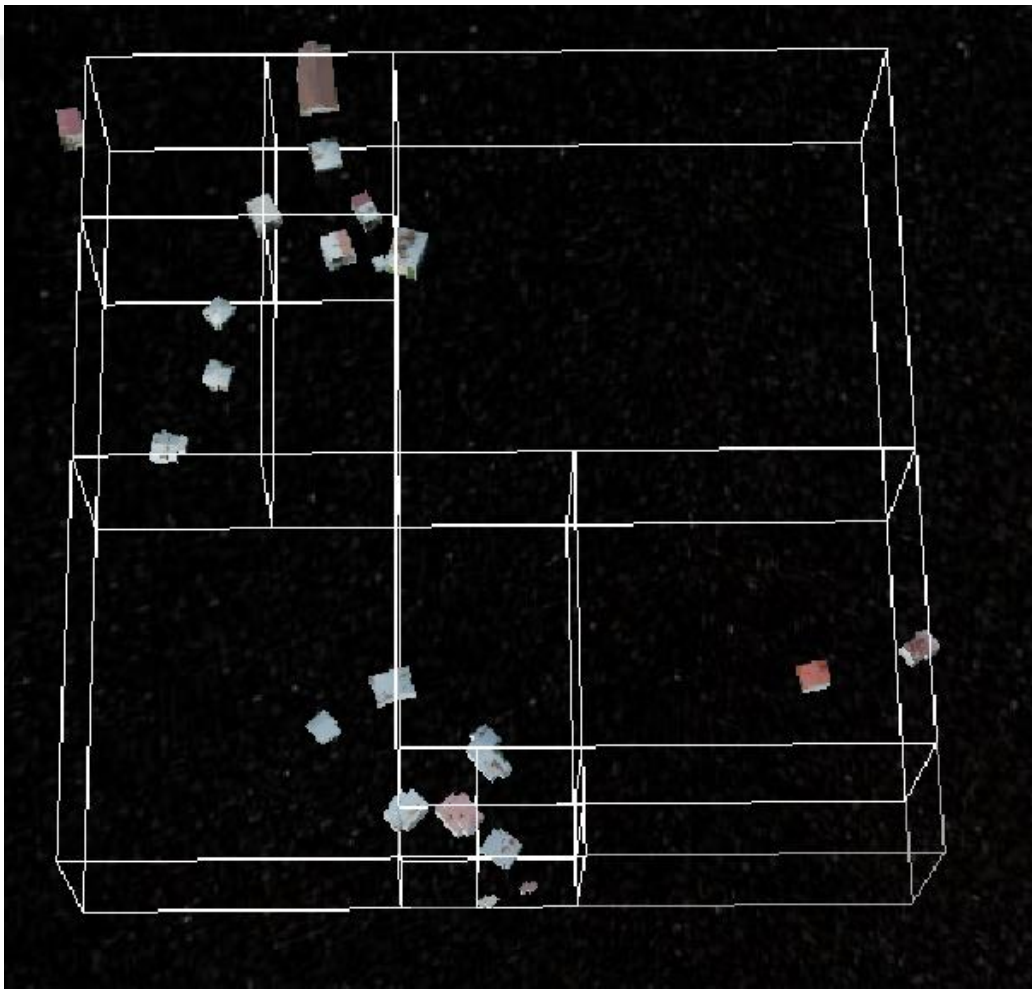
Şekil 44. Avarage Tree algoritmasına göre çalışma alanının bölümleme sonucu elde edilen ağaç yapısı

Gerçekleşen bölümleme sonucunda 3 seviyeli bir ağaç yapısı elde edildiği gözlemlenmiştir. Elde edilen sınırlayıcı hacimlerin tile koordinartları belirlenmiştir. Tile koordinarlarının veriliş sıralaması ise Morton Z-ordercurve doldurma eğrisi temel alınarak oluşturulmuştur. Böylece oluşan GLTF dosyaları ile tilese.json dosyası sorunsuz bir şekilde haberleşme gerçekleştirebilmiştir.

Oluşan sonuç dosyaları önerilen algoritmaya en yakın olan QuadTree algoritması ile kıyaslandığında oluşan ağaç seviyelerinin aynı olduğu ancak oluşturulma süresinin QuadTree algoritmasında 00.01.41 (dk/sn/sl), Optimize QuadTree algoritmasında 00.01.22 (dk/sn/sl) olduğu gözlemlenmiş, önerilen algoritmanın daha hızlı olduğu tespit edilmiştir. Bunun yanı sıra, bölümleme koordinatları kıyaslandığında aynı veri kümesi için QuadTree algoritmasında genellikle alanın tam orta noktasında

bölümleme yapılırken, Optimize QuadTree algoritmasında bölümleme koordinatlarının mevcut alanın sağ tarafına doğru kaydığı gözlemlenmiştir. Bu durum, Optimize QuadTree algoritmasının daha çok bina verisi içeren bölgelerde çalıştığında verimliliği ve sunum hızını artırabileceğini öngörmektedir. Çünkü bina verilerinin yoğun olduğu bölgelerde daha küçük ve daha hassas bölümler oluşturulması, 3D Tiles setlerinin daha etkili bir şekilde oluşturulmasını ve sunumun daha hızlı yapılmasını sağlamaktadır. Bu şekilde, kullanıcılar daha hızlı bir veri akışıyla daha kapsamlı ve detaylı 3D görsellere erişebilir, bu da genel verimliliği ve sunum hızını artırır.

Şekil 45'de 21 bina için avarage tree algoritması uygulanarak oluşturulmuş tilenmış verinin 3 boyutlu olarak uzaydaki görüntüsü görülmektedir.



Şekil 45. R-Tree algoritmasına göre Biçer Mahallesinin sonuç görüntüsü

6.SONUÇLAR VE ÖNERİLER

6.1. 3D Tiles Problemi İçin Deneysel Sonuçlar

6.1.1. İzmir Narlıdere

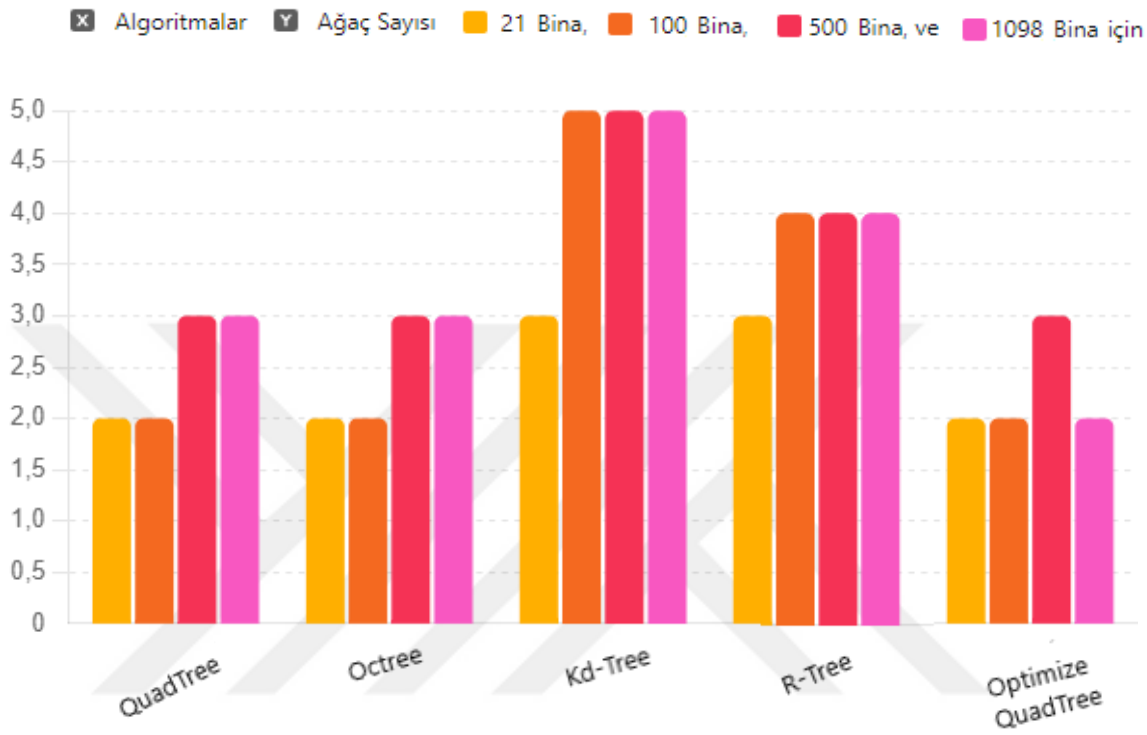
İzmir Narlıdere ilçesi çalışma alanı içerdiği 3D bina sayısı bakımından tezde kullanılan diğer çalışma alanlarının arasından toplam 1098 bina adedi ile en küçük alandır. Bu çalışma alanı için toplam bina sayısı performans kıyaslaması için 21, 100, 500 ve 1098 adet bina olmak üzere 4 parçaya ayrılarak beş algoritmanın ağaç uzunluğu sayısı, oluşturma süreleri ve dosya boyutları sonuçları Tablo 2'de görülmektedir. Algoritmalar çalıştırılırken maksimum nokta sayısı, toplam bina sayısının %25'i olacak şekilde verilmiştir.

Tablo 2. Narlıdere ilçesi yöntemler sonrası ağaç, süre ve dosya boyutu sonuçları

	21 Bina (Max_Point = 5)			100 Bina (Max_Point = 25)			500 Bina (Max_Point = 125)			1098 Bina (Max_Point = 275)		
	Ağaç Sayısı	Süre (dk/sn/sl)	Dosya Boyutu	Ağaç Sayısı	Süre (dk/sn/sl)	Dosya Boyutu	Ağaç Sayısı	Süre (dk/sn/sl)	Dosya Boyutu	Ağaç Sayısı	Süre (dk/sn/sl)	Dosya Boyutu
QuadTree	2	00.01.41	2.20 MB	2	00.02.52	10.8 MB	3	00.04.01	67 MB	3	02.34.05	332 MB
Octree	2	00.02.16	2.22 MB	2	00.01.12	8.7 MB	3	00.03.32	63 MB	3	02.52.45	298 MB
Kd-Tree	3	00.02.32	2.20 MB	5	00.04.04	10.3 MB	5	00.04.03	67 MB	5	03.09.03	332 MB
R-Tree	3	00.01.01	2.21 MB	4	00.03.29	10.8 MB	4	00.06.25	67 MB	4	02.59.82	332 MB
Optimize QuadTree	2	00.01.22	2.20 MB	2	00.01.20	10.8 MB	3	00.03.22	67 MB	2	02.32.20	332 MB

Tablo 2 incelendiğinde 5 algoritmanın ağaç sayıları kıyaslandığında, her işlem adımında tek eksen yönünde bölme işlemi gerçekleştirildiğinden Kd-Tree ve R-Tree algoritmalarının ağaç sayısı diğer algoritmalara göre daha fazla olduğu görülmektedir. Tile yapılarının oluşturulma süreleri kıyaslandığında küçük veri kümesinde R-Tree algoritmasının daha hızlı olduğu ancak dosya boyutu artıkça Optimize QuadTree algoritmasının hızlandığı görülmektedir. Dosya boyutları kıyaslandığında ise yakın boyutlar elde edildiği gözlemlenmiştir.

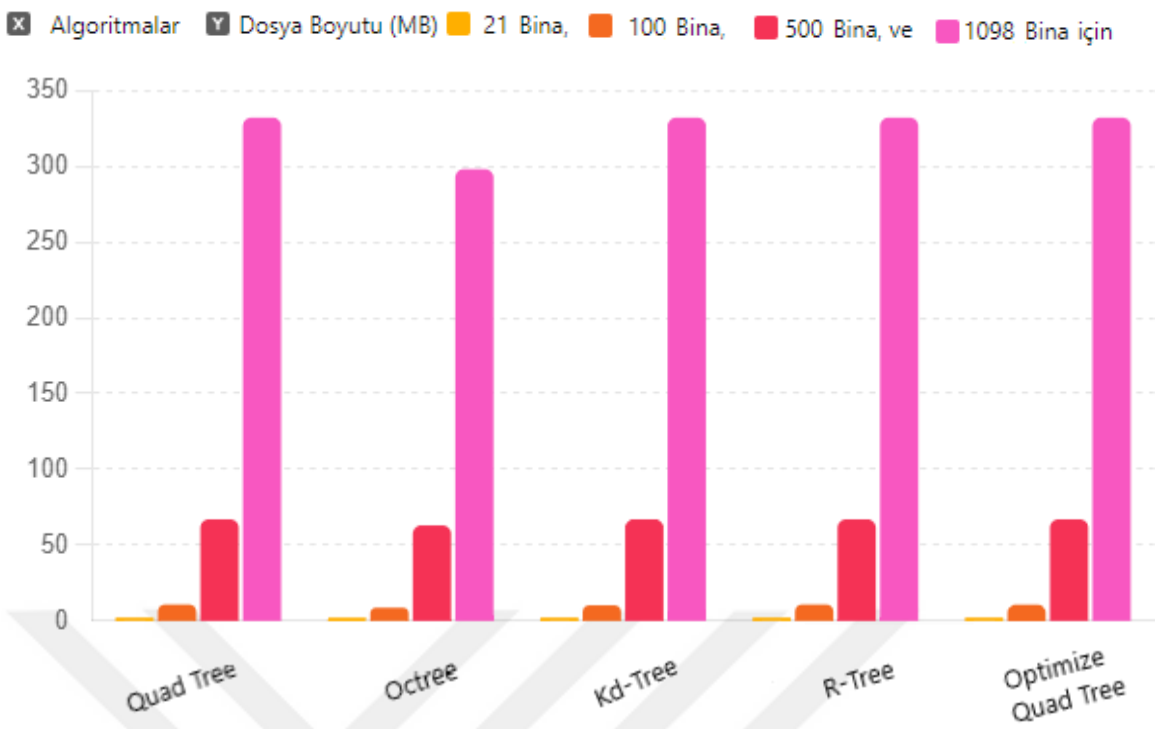
Şekil 46’daki grafikte, farklı bina grupları için ağaç sayısı performansını görselleştirerek, çeşitli algoritmaların hangi durumlarda daha verimli olduğunu değerlendirmede yardımcı olmaktadır. Bu görselleştirme, QuadTree, Octree, Kd-Tree, R-Tree ve Optimize QuadTree algoritmalarının her birinin performansını karşılaştırmalı olarak analiz etmeyi mümkün kılmaktadır.



Şekil 46. Narlıdere ilçesi, farklı bina grupları için ağaç sayısı karşılaştırması

Şekil 46 incelendiğinde, QuadTree ve Octree algoritmalarının ağaç sayısı bakımından benzer performans sergilediği ve ortalama olarak 2 ile 3 arasında değişen ağaç sayısına sahip oldukları görülmektedir. Kd-Tree algoritmasının ise en yüksek ağaç sayısına sahip algoritma olduğu gözlemlenmiştir. R-Tree algoritmasının performans açısından Kd-Tree'ye yakın olduğu ve yüksek ağaç sayısı ile ikinci sırada yer aldığı dikkat çekmektedir. Optimize QuadTree algoritmasının ise diğer algoritmalarla kıyaslandığında daha düşük ağaç sayısına sahip olduğu görülmektedir.

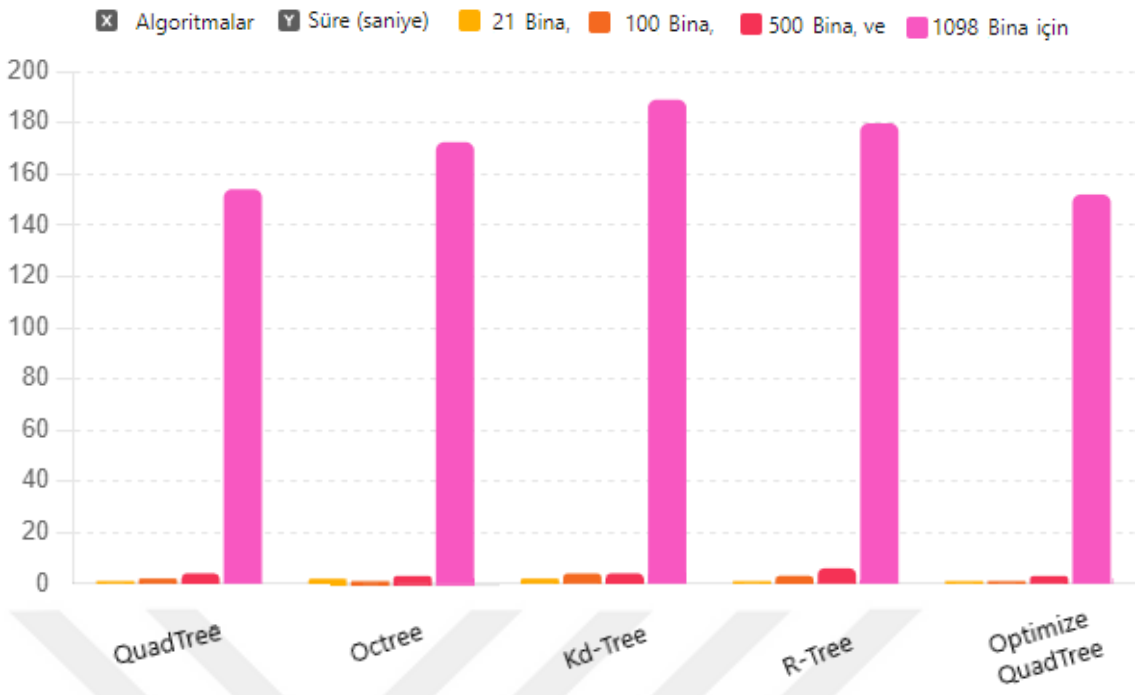
Şekil 47’deki grafikte, farklı algoritmaların çeşitli bina grupları için dosya boyutu performansı görülmektedir.



Şekil 47. Narlıdere ilçesi, farklı bina grupları için dosya boyutu karşılaştırması

QuadTree ve Optimize QuadTree algoritmalarının, 21 bina için düşük dosya boyutlarına sahip olduğu gözlemlenmiştir. Ancak, 100, 500 ve 1098 bina gruplarında bu algoritmaların dosya boyutları artmaktadır. Octree algoritması da 21 bina için düşük dosya boyutuna sahiptir ve 100 bina grubu için dosya boyutu diğer algoritmalarla benzer seviyededir. Fakat, 500 ve 1098 bina gruplarında Octree algoritmasının dosya boyutları artış göstermektedir. Kd-Tree ve R-Tree algoritmaları ise benzer dosya boyutlarına sahip olup, 21 bina grubu için düşük dosya boyutları sergilemekte, ancak 100, 500 ve 1098 bina gruplarında dosya boyutları artmaktadır.

Şekil 48'deki grafikte, farklı algoritmaların çeşitli bina grupları için süre performansı görselleştirilmektedir.



Şekil 48. Narlıdere ilçesi, farklı bina grupları için süre karşılaştırması

Şekil 48 incelendiğinde, 21 bina grubu için en düşük süreye sahip algoritmaların QuadTree, Octree ve Optimize QuadTree olduğu görülmektedir. 100 bina grubu için Octree, diğer algoritmalara kıyasla biraz daha düşük süreye sahiptir. 500 ve 1908 bina gruplarında ise tüm algoritmalar benzer süre değerlerine sahip olup, bu sürelerin oldukça yüksek olduğu gözlemlenmektedir.

6.1.2. Hakkari Biçer

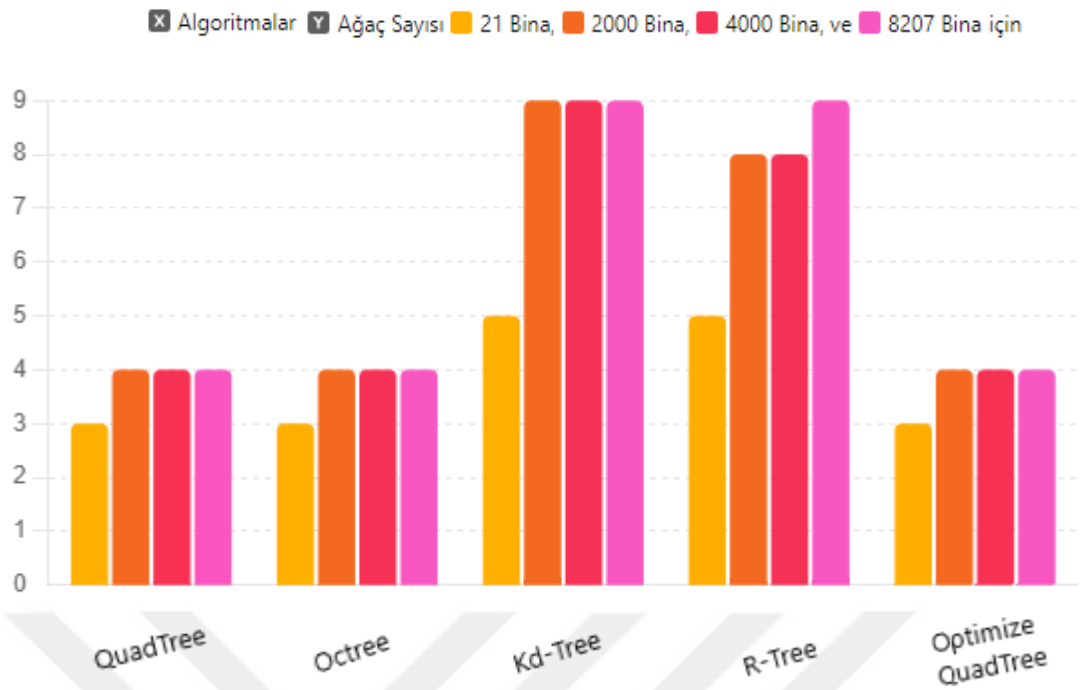
Hakkari Biçer Mahallesi çalışma alanı içerdiği 3D bina sayısı bakımından tezde kullanılan diğer çalışma alanlarının arasından toplam 8207 bina adedi ile orta düzey bir alandır. Bu çalışma alanı için toplam bina sayısı performans kıyaslaması 21, 2000, 4000 ve 8207 adet bina olmak üzere 4 parçaya ayrılarak beş algoritmanın ağaç uzunluğu sayısı, oluşturma süreleri ve dosya boyutları sonuçları Tablo 3'de görülmektedir. Algoritmalar çalıştırılırken maksimum nokta sayısı, toplam bina sayısının %25'i olacak şekilde verilmiştir.

Tablo 3. Biçer Mahallesi yöntemler sonrası ağaç,süre ve dosya boyutu sonuçları

	21 Bina (Max_Point = 5)			2000 Bina (Max_Point = 50)			4000 Bina (Max_Point = 100)			8207 Bina (Max_Point = 200)		
	Ağaç Sayısı	**Süre (dk/sn/sl)**	**Dosya Boyutu**	**Ağaç Sayısı**	**Süre (dk/sn/sl)**	**Dosya Boyutu**	**Ağaç Sayısı**	**Süre (dk/sn/sl)**	**Dosya Boyutu**	**Ağaç Sayısı**	**Süre (dk/sn/sl)**	**Dosya Boyutu**
QuadTree	3	00.02.50	5.85 MB	4	00.40.37	515 MB	4	01.33.17	0.98 GB	4	02.39.57	1.89 GB
Octree	3	00.03.16	5.87 MB	4	00.36.46	375 MB	4	00.51.02	803 MB	4	02.52.45	1.59 GB
Kd-Tree	5	00.03.08	5.85 MB	9	00.49.07	521 MB	9	01.28.31	0.98 GB	9	03.09.34	1.89 GB
R-Tree	5	00.02.18	5.86 MB	8	00.46.29	515 MB	8	01.41.56	0.989 GB	9	02.59.28	1.89 GB
Optimize QuadTree	3	00.02.34	5.85 MB	4	00.36.08	515 MB	4	01.27.24	0.986 GB	4	02.34.58	1.89 GB

Tablo 3 incelendiğinde QuadTree, Octree ve Optimize QuadTree yapıları genellikle ağaç sayısını artırırken, Kd-Tree ve R-Tree belli bir bina sayısından sonra sabit ağaç sayısı ile işlem yapmaktadır. Veri setinin boyutu arttıkça, genellikle her ağaç yapısı için ağaç sayısı artmaktadır. QuadTree, Octree ve Optimize QuadTree, diğer ağaç yapılarına göre daha hızlı oluşturulur. Kd-Tree ve R-Tree, daha fazla zaman gerektiren ağaç yapılarıdır. Veri setinin boyutu arttıkça, ağaç oluşturma süresi genellikle artmaktadır. QuadTree, Octree ve Optimize QuadTree, genellikle daha küçük dosya boyutlarına sahiptir. Kd-Tree ve R-Tree, daha büyük dosya boyutlarına sahiptir.

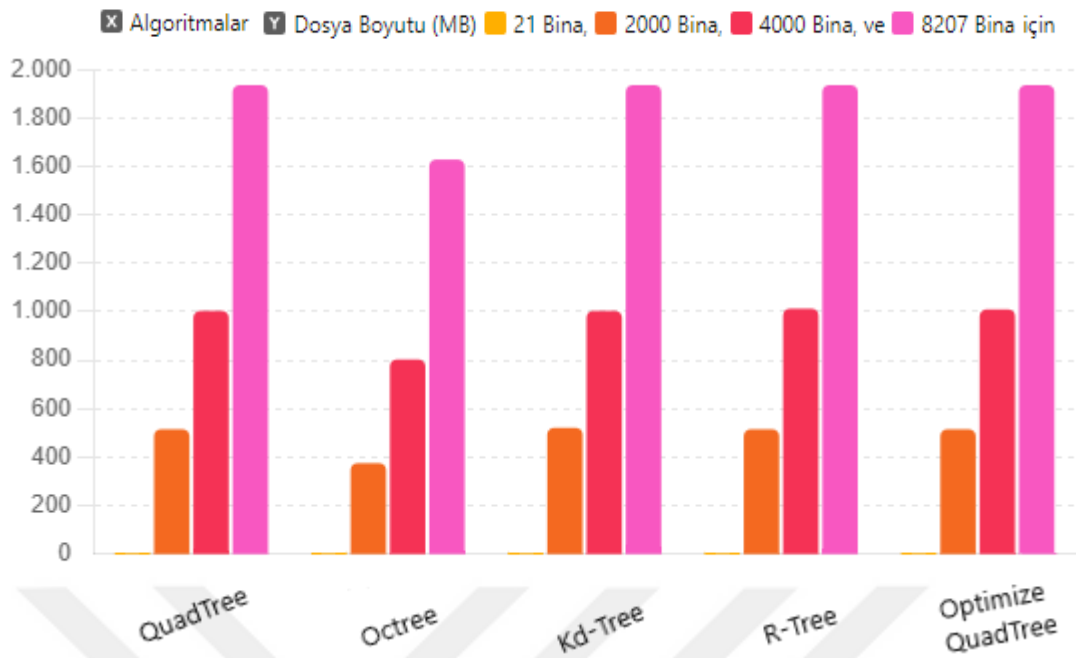
Şekil 49'daki grafik, her bir algoritmanın farklı veri seti boyutları için oluşturduğu ağaç sayısını göstermektedir. Bu değerlendirme, algoritmaların ölçeklenebilirliğini ve büyük veri setlerine karşı nasıl performans gösterdiklerini anlamak için bilgiler sunmaktadır.



Şekil 49. Biçer Mahallesi, farklı bina grupları için ağaç sayısı karşılaştırması

Algoritmaların ağaç sayıları üzerindeki performansı değerlendirildiğinde, QuadTree, Octree ve Optimize QuadTree algoritmalarının büyük veri setlerinde dahi ağaç sayısını sabit tutarak ölçeklenebilirlik açısından üstünlük sağladığı gözlemlenmektedir. Bu algoritmalar, veri seti boyutuna bakılmaksızın tutarlı bir performans sergilerken, Kd-Tree ve R-Tree algoritmalarının bina sayısı arttıkça daha fazla ağaç gerektirdiği ve bu nedenle daha karmaşık veri yapıları oluşturduğu görülmektedir.

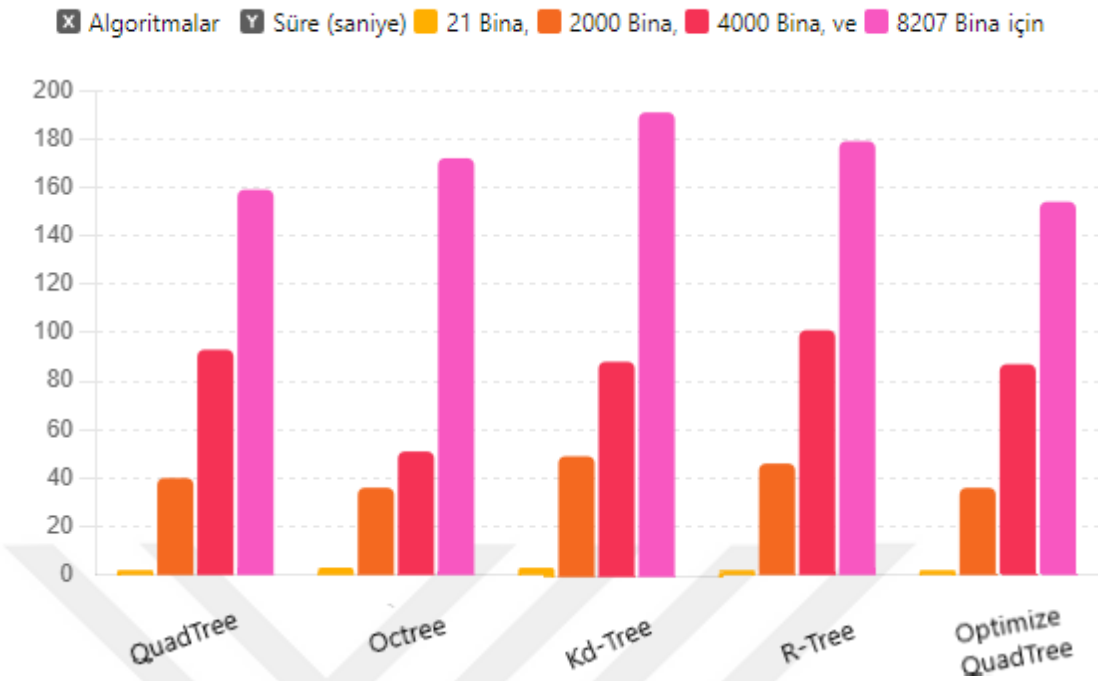
Şekil 50'deki grafikte, farklı algoritmaların çeşitli bina grupları için dosya boyutu performansı görülmektedir.



Şekil 50. Biçer Mahallesi, farklı bina grupları için dosya boyutu karşılaştırması

Algoritmaların dosya boyutları üzerindeki performansı değerlendirildiğinde, QuadTree, Octree, Kd-Tree, R-Tree ve Optimize QuadTree algoritmalarının dosya boyutlarının bina sayısı arttıkça arttığı gözlemlenmektedir. Özellikle Octree algoritması, küçük ve orta ölçekli veri setlerinde diğer algoritmalara göre daha düşük dosya boyutları sunmaktadır. Bu durum, Octree'nin daha verimli bir veri yapısına sahip olduğunu ve veri sıkıştırma konusunda avantaj sağladığını göstermektedir. QuadTree, Kd-Tree, R-Tree ve Optimize QuadTree algoritmaları, büyük veri setlerinde benzer dosya boyutlarına sahiptir. Bu algoritmalar, veri seti boyutu arttıkça daha fazla yer kaplamakta ve büyük veri setlerinde tutarlı performans göstermektedirler.

Şekil 51'deki grafikte, farklı algoritmaların çeşitli bina grupları için süre performansı görselleştirilmektedir.



Şekil 51. Biçer Mahallesi, farklı bina grupları için süre karşılaştırması

Algoritmaların işlem süreleri üzerindeki performansı değerlendirildiğinde, QuadTree, Octree, Kd-Tree, R-Tree ve Optimize QuadTree algoritmalarının bina sayısı arttıkça işlem sürelerinde önemli bir artış gösterdiği gözlemlenmektedir. Octree algoritması, küçük ve orta ölçekli veri setlerinde diğer algoritmalara göre daha hızlıdır, ancak büyük veri setlerinde performans kaybı yaşamaktadır. Optimize QuadTree algoritması, özellikle büyük veri setlerinde daha iyi performans göstermektedir. Bu durum, optimize edilmiş algoritmaların büyük veri setlerinde daha verimli olduğunu göstermektedir.

6.1.3. Ankara Çankaya

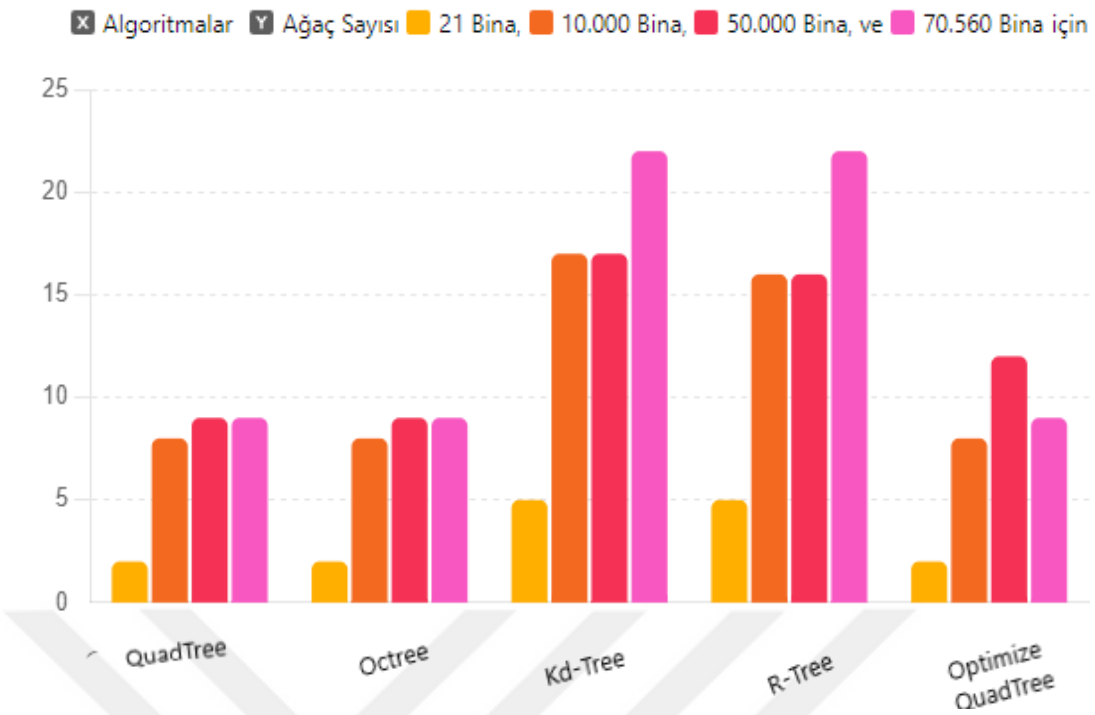
Ankara Çankaya ilçesi çalışma alanı, içerdiği 3D bina sayısı bakımından tezde kullanılan diğer çalışma alanlarının arasından toplam 70560 bina adedi ile en büyük ve karmaşık yapıya sahip olan arazidir. Bu çalışma alanı için toplam bina sayısı performans kıyaslaması 21,10.000,50.000 ve 70.560 adet bina olmak üzere 4 parçaya ayrılarak beş algoritmanın ağaç uzunluğu sayısı, oluşturma süreleri ve dosya boyutları sonuçları Tablo 4'de görülmektedir. Algoritmalar çalıştırılırken maksimum nokta sayısı, en fazla 1000 obje içerecek şekilde bina sayısına oranlanarak verilmiştir.

Tablo 4. Çankaya ilçesi yöntemler sonrası ağaç, süre ve dosya boyutu sonuçları

	21 Bina (Max_Point = 5)			10.000 Bina (Max_Point = 200)			50.000 Bina (Max_Point = 500)			70.560 Bina (Max_Point = 1000)		
	Ağaç Sayısı	**Süre (dk/sn/sl)**	**Dosya Boyutu**	**Ağaç Sayısı**	**Süre (dk/sn/sl)**	**Dosya Boyutu**	**Ağaç Sayısı**	**Süre (dk/sn/sl)**	**Dosya Boyutu**	**Ağaç Sayısı**	**Süre (dk/sn/sl)**	**Dosya Boyutu**
QuadTree	2	00.03.50	8.15 MB	8	03.36.37	2.31 GB	9	15.37.31	13.72 GB	9	23.33.11	16.53 GB
Octree	2	00.03.16	8.19 MB	8	03.36.13	2.03 GB	9	14.32.22	11.51 GB	9	25.55.55	16.51 GB
Kd-Tree	5	00.04.45	8.15 MB	17	03.49.04	2.17 GB	17	15.13.02	13.72 GB	22	27.03.20	16.54 GB
R-Tree	5	00.03.21	8.18 MB	16	03.46.29	2.31 GB	16	15.41.58	13.72 GB	22	25.42.22	16.53 GB
Optimize QuadTree	2	00.03.43	8.15 MB	8	03.40.01	2.31 GB	12	15.11.11	13.72 GB	12	21.52.41	16.53 GB

Veri setinin büyüklüğü arttıkça, her ağaç yapısı için oluşturulan ağaç sayısı artar. Kd-Tree ve R-Tree, en fazla ağaç sayısına sahip olabilirken; QuadTree ve Octree daha az ağaç oluşturabilir. Veri setinin boyutu arttıkça, ağaç oluşturma süresi genellikle artar. Kd-Tree ve R-Tree, daha büyük veri setlerinde daha uzun süreler gerektirebilir. Veri setinin boyutu arttıkça, ağaç yapıları genellikle daha büyük dosya boyutlarına sahip olur. Kd-Tree ve R-Tree, daha büyük veri setlerinde daha büyük dosya boyutlarına sahip olabilir.

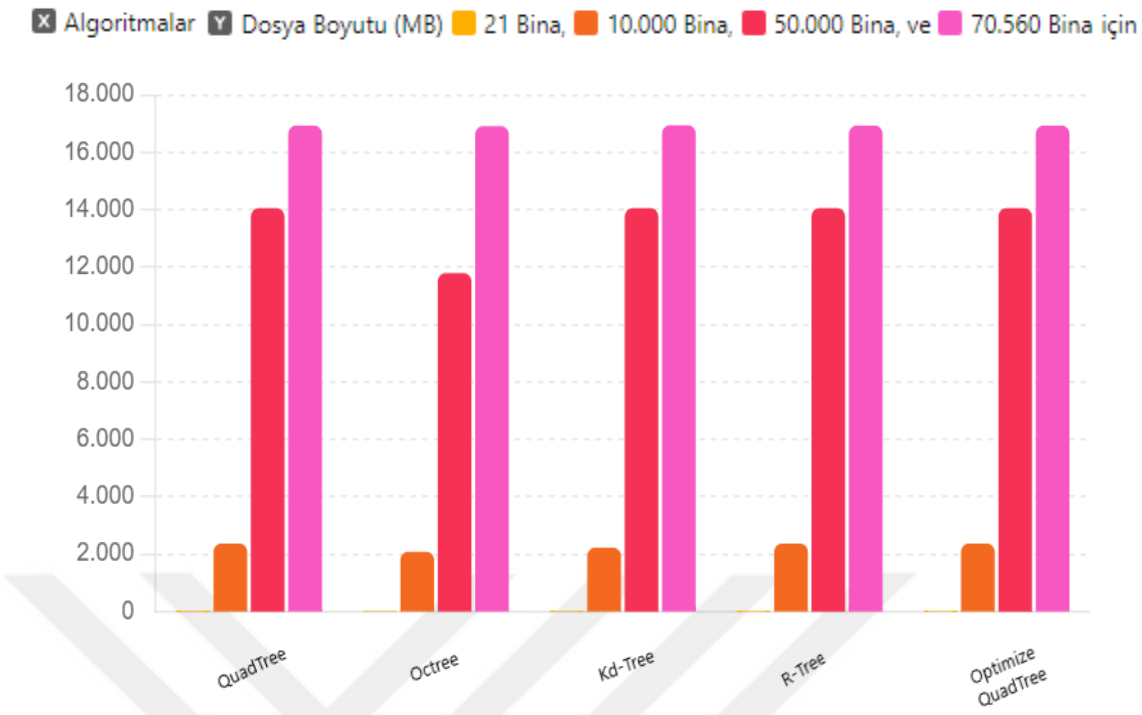
Şekil 52' deki grafikte, farklı bina grupları için ağaç sayısı performansını görselleştirerek, çeşitli algoritmaların hangi durumlarda daha verimli olduğunu değerlendirmede yardımcı olmaktadır. Bu görselleştirme, QuadTree, Octree, Kd-Tree, R-Tree ve Optimize QuadTree algoritmalarının her birinin performansını karşılaştırmalı olarak analiz etmeyi mümkün kılmaktadır.



Şekil 52. Çankaya ilçesi, farklı bina grupları için ağaç sayısı karşılaştırması

Şekil 52' deki grafik incelendiğinde, Kd-Tree ve R-Tree algoritmalarının artan bina sayısına karşılık ağaç sayısını önemli ölçüde artırdığı gözlemlenmektedir. QuadTree ve Octree algoritmalarının ağaç sayısında ise çok az bir artış göstermesi, bu algoritmaların daha sabit bir performansa sahip olduğunu ortaya koymaktadır. Optimize QuadTree algoritması, bina sayısına göre daha dinamik bir değişim sergilemekte, ancak en yüksek bina sayısında bile düşük ağaç sayısı ile dikkat çekmektedir.

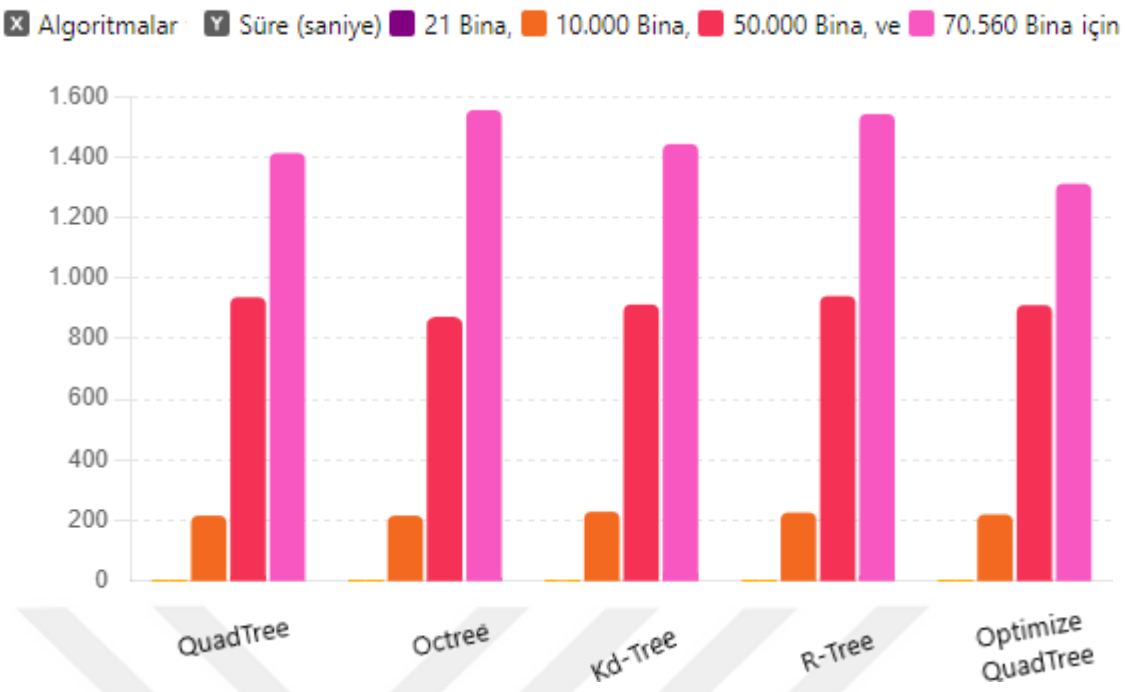
Şekil 53’ deki grafikte, farklı algoritmaların çeşitli bina grupları için dosya boyutu performansı görülmektedir.



Şekil 53. Çankaya ilçesi, farklı bina grupları için dosya boyutu karşılaştırması

Şekil 53'deki grafik incelendiğinde 21 bina için tüm algoritmaların benzer dosya boyutlarına sahip olduğu gözlemlenmiştir (8.15-8.19 MB). 10.000 bina için ise Octree algoritması daha düşük bir dosya boyutu sunmaktadır (2073.92 MB). 50.000 ve 70.560 bina için dosya boyutları algoritmalar arasında nispeten benzer görünmektedir, ancak Octree bu kategorilerde biraz daha düşük dosya boyutlarına sahiptir.

Şekil 54'deki grafikte, farklı algoritmaların çeşitli bina grupları için süre performansı görselleştirilmektedir



Şekil 54. Çankaya ilçesi, farklı bina grupları için süre karşılaştırması

Algoritmaların işlem süreleri üzerindeki performansı değerlendirildiğinde, QuadTree ve Optimize QuadTree algoritmaları arasındaki farklar dikkate değerdir. Optimize QuadTree, büyük veri setlerinde belirgin şekilde daha iyi performans göstermektedir. Bu durum, optimizasyonların büyük veri setlerinde önemli faydalar sağladığını göstermektedir. Octree ve Kd-Tree, genel olarak dengeli bir performans sergilemektedir, ancak büyük veri setlerinde performansları düşmektedir. R-Tree algoritması ise büyük veri setlerinde diğer algoritmalara göre daha az etkilidir.

6.2. Sonuç

3D Tiles, coğrafi bilgi sistemleri (CBS) ve sanal dünya uygulamaları için hayati bir öneme sahip olan veri standardıdır. Bu standardın kullanımı, büyük ölçekli coğrafi verilerin etkili bir şekilde depolanması, işlenmesi ve iletilmesi için kritik bir rol oynar. 3D Tiles, coğrafi verilerin daha kolay erişilebilir olmasını sağlamakla kalmayıp aynı zamanda veri boyutlarını optimize etmek ve işlem gücünü artırmak açısından da büyük bir avantaj sağlamaktadır. Özellikle mobil cihazlarda ve web tabanlı uygulamalarda, veri boyutlarının optimize edilmesi ve işlem gücünün etkin bir şekilde kullanılması oldukça önemlidir. 3D Tiles standardı, bu ihtiyaçlara yanıt vermek için tasarlanmıştır ve CBS ve sanal dünya uygulamalarının performansını artırmak için etkili bir araçtır. Bu çalışma, farklı 3D Tiles veri setleri üzerinde beş farklı algoritmanın performansını değerlendirerek, bu algoritmaların 3D Tiles veri setlerinin etkin bir şekilde işlenmesine

ve görselleştirilmesine nasıl katkı sağladığını incelemektedir. Bu süreçte, ham formattaki nesneleri dışarıdan herhangi bir müdahale gerektirmeden işleyebilen ve sonuç olarak gltf formatında sunuma hazır nihai ürünler üretebilen açık kaynaklı bir yazılım geliştirilmiştir. Geliştirilen bu yazılım sayesinde kullanıcılar, herhangi bir yazılım bileşeni kurmadan, özellikle veri yoğunluğunun yüksek olduğu bölgelerde 3D nesnelerini hiyerarşik olarak bölümlendirerek, bölgedeki maksimum bina sayısını belirleyerek tek bir kodla işlem gerçekleştirebilirler.

Çalışma alanı olarak İzmir Narlıdere ilçesi, Hakkari Biçer Mahallesi ve Ankara Çankaya ilçesi olarak 3 arazi kullanılmıştır. Bu araziler üzerinde QuadTree, Octree, Kd-Tree, R-Tree ve önerilen Optimize QuadTree algoritmaları uygulanmıştır. Her bir algoritmanın ağaç sayıları, dosya boyutları ve işlem süreleri açısından performansları incelenmiştir.

İzmir ilçesine ait veri setleri üzerinde yapılan analizlerde, QuadTree, Octree ve Optimize QuadTree algoritmalarının benzer dosya boyutlarına sahip olduğu ancak işlem süreleri açısından QuadTree ve Optimize QuadTree'nin avantajlı olduğu gözlemlenmiştir.

Hakkari iline ait veri setleri üzerinde yapılan analizlerde ise, QuadTree ve Octree algoritmalarının ölçeklenebilirlik açısından üstünlük sağladığı, ancak Octree algoritmasının dosya boyutları açısından avantajlı olduğu görülmüştür. Ayrıca, Optimize QuadTree'nin büyük veri setlerinde daha iyi performans gösterdiği ve optimizasyonların büyük veri setlerinde önemli faydalar sağladığı belirlenmiştir.

Ankara iline ait veri setleri üzerinde yapılan analizler ise, Optimize QuadTree'nin büyük veri setlerinde daha iyi performans gösterdiğini ve optimizasyonların büyük veri setlerinde önemli faydalar sağladığını göstermiştir.

Kd-Tree ve R-Tree algoritmaları genellikle QuadTree, Optimize QuadTree ve Octree'ye göre daha fazla ağaç kullanarak çalıştığı gözlemlenmektedir. Bu, özellikle daha karmaşık ve detaylı yapıları temsil etmek için daha fazla bölünmeye ihtiyaç duydukları anlamına gelmektedir. QuadTree, Optimize QuadTree ve Octree, verimlilik açısından daha iyi performans sergileyebildikleri ve daha az ağaç kullanmalarına rağmen işlemlerini hızlı bir şekilde gerçekleştiriyor oldukları elde edilmiştir.

Genel olarak, yapılan kıyaslamada dosya boyutları açısından QuadTree ve Octree'nin daha etkili olduğu görülmüştür. Ancak, oluşturulma süresi, ağaç sayısı ve dosya boyutunun ortalama performansı dikkate alındığında, önerilen yöntem olan Optimize QuadTree algoritmasının daha avantajlı olduğu görülmüştür. Bu çalışma, bu

algoritmaların performansını değerlendirmede bir temel sağlamakta ve gelecekteki araştırmalar için bir yol haritası sunmaktadır.

Gelecekteki çalışmalarda, 3D Tiles veri setlerinin işlenmesi için farklı bölümleme algoritmaları kullanılabilir ve veri setlerinin boyutlarını azaltmak amacıyla gelişmiş sıkıştırma algoritmalarının uygulanması ve etkilerinin incelenmesi sağlanabilir.

KAYNAKLAR

Anon(2019-a). (2019). *3D tiles format specification*. GitHub. https://github.com/AnalyticalGraphicInc/3d-tiles/tree/master/specification#tile-format-specification[Ziyaret Tarihi : 15 Mart 2023]

Benner, J., Geiger, A., Gröger, G., Häfele, K.-H., & Löwner, M.-O. (2013). Enhanced LOD concepts for virtual 3D city models. *ISPRS annals of the photogrammetry, remote sensing and spatial information sciences*, *2*, 51-61.

Biljecki, F., Ledoux, H., & Stoter, J. (2016). An improved LOD specification for 3D building models. *Computers, Environment and Urban Systems*, *59*, 25-37.

Buyukdemircioglu, M., & Kocaman, S. (2020). Reconstruction and efficient visualization of heterogeneous 3D city models. *Remote Sensing*, *12*(13), 2128.

Cesium. *Cesium-ion*. Cesium,https://cesium.com/platform/cesium-ion[Ziyaret Tarihi: 22 Aralık 2022]

Chaturvedi, K., Yao, Z., & Kolbe, T. H. (2015). Web-based Exploration of and interaction with large and deeply structured semantic 3D city models using HTML5 and WebGL. Bridging Scales-Skalenübergreifende Nah-und Fernerkundungsmethoden, 35. Wissenschaftlich-Technische Jahrestagung der DGPF.

Chen, J., Li, J., & Li, M. (2016). Progressive visualization of complex 3D models over the internet. *Transactions in GIS*, *20*(6), 887-902.

Chen, Y., Shooraj, E., Rajabifard, A., & Sabri, S. (2018). From IFC to 3D tiles: An integrated open-source solution for visualising BIMs on cesium. *ISPRS International Journal of Geo-Information*, *7*(10), 393

Cozzi, P. (2015). 3D tiles. Cesium,https://cesium.com/blog/2015/08/10/introducing-3d-tiles/[Ziyaret Tarihi: 20 Ocak 2023]

Cozzi , P., & Lilley, S. (2022). *3D Tiles Specification Open Geospatial Consortium*,https://docs.ogc.org/cs/22-025r4/22-025r4.html [Ziyaret Tarihi : 01 Ocak 2022]

Csillag, F. (2023). Quadtrees: hierarchical multiresolution data structures for analysis of digital images. In *Scale in remote sensing and GIS* (pp. 247-271). Routledge.

Fan, Z., & Ortega, A. (2010). Optimization of overlapped tiling for efficient 3d image retrieval. 2010 Data Compression Conference.

Gaillard, J., Vienne, A., Baume, R., Pedrinis, F., Peytavie, A., & Gesquière, G. (2015). Urban data visualisation in a web browser. Proceedings of the 20th International Conference on 3D Web Technology,

Gesquière, G., & Manin, A. (2012). 3D visualization of urban data based on CityGML with WebGL. *International Journal of 3-D Information Modeling (IJ3DIM)*, *1*(3), 1-15.

GitHub-objto3dtiles. Convert obj model file to 3d tiles. https://github.com/PrincessGod/objTo3d-tiles.[Ziyaret Tarihi : 05 Haziran 2023]

GitHub-3DTiles. (2020). *Specification for streaming massive heterogeneous 3D geospatial datasets*.https://github.com/CesiumGS/3d-tiles[Ziyaret Tarihi :10 Şubat 2023]

Guo, N., Xiong, W., Wu, Y., Chen, L., & Jing, N. (2019). A geographic meshing and coding method based on adaptive Hilbert-Geohash. *IEEE Access*, *7*, 39815-39825.

Haverkort, H. J., & Toma, L. (2016). Quadtrees and Morton Indexing.

Hillmann, M., Igelbrink, F., & Wiemann, T. (2022). Computing Arbitrarily Large Meshes with Level-Of Support for Cesium 3d Tiles. *ISPRS-International*

Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, *48*, 109-116.

Hu, Z., Guo, J., & Zhang, X. (2020). Three-dimensional (3D) parametric modeling and organization for web-based visualization of city-scale pipe network. *ISPRS International Journal of Geo-Information*, *9*(11), 623.

Jaillot, V. (2020). *3D, temporal and documented cities: formalization, visualization and navigation* Université de Lyon].

Khayyal, H. K., Zeidan, Z. M., & Beshr, A. (2022). Creation and spatial analysis of 3D city modeling based on GIS data. *Civil Engineering Journal*, *8*(1), 105.

Kim, T. H., Jang, H. S., Yoo, S. H., & Go, J. H. (2017). 3D Tile Application Method for Improvement of Performance of V-world 3D Map Service. *Journal of Korean Society for Geospatial Information Science*, *25*(1), 55-61.

Klimke, J. (2019). *Web-based provisioning and application of large-scale virtual 3D city models* Dissertation, Potsdam, Universität Potsdam, 2019].

Lai, C.-F. (2016). *Tile-based modeling and its applications in computer graphics*

Lilley, S. (2021). *3d Tiles Reference Card*. https://github.com/CesiumGS/3d-tiles/blob/main/3d-tiles-reference-card.pdf[Ziyaret Tarihi: 11 Ocak 2023]

Lights, L. (2021). *3dTiles geometrik hatalarının ayrıntılı açıklaması*. BY-NC-SA. https://www.cnblogs.com/onsummer/p/13357226.html [Ziyaret Tarihi : 22 Mart 2023]

Lu, M., Wang, X., Liu, X., Chen, M., Bi, S., Zhang, Y., & Lao, T. (2021). Web-based real-time visualization of large-scale weather radar data using 3D tiles. *Transactions in GIS*, *25*(1), 25-43.

Mete, M. O., Guler, D., & Yomralioglu, T. (2018). Development of 3D web GIS application with open source library. *Selçuk Üniversitesi Mühendislik, Bilim Ve Teknoloji Dergisi*, *6*, 818-824.

Murshed, S. M., Al-Hyari, A. M., Wendel, J., & Ansart, L. (2018). Design and implementation of a 4D web application for analytical visualization of smart city applications. *ISPRS International Journal of Geo-Information*, *7*(7), 276.

Murtiyoso, A., Holm, S., Riihimäki, H., Krucher, A., Griess, H., Griess, V. C., & Schweier, J. (2024). Virtual forests: a review on emerging questions in the use and application of 3D data in forestry. International Journal of Forest Engineering, 35(1), 29-42.

Nguyễn, T. Đ., Văn, T. T., & Lê, M. T. (2022). PHÂN LỚP HÌNH ẢNH DỰA TRÊN CẤU TRÚC KD-TREE. *Hue University Journal of Science: Techniques and Technology*, *131*(2A), 103-120.

Octree. (2023). *Octree | Insertion and Searching*. geeksforgeeks. https://www.geeksforgeeks.org/octree-insertion-and-searching [Ziyaret Tarihi : 15 Ocak 2023]

Parisi, & Tony. (2012). *WebGL: up and running*. " O'Reilly Media, Inc.".

Prandi, F., Devigili, F., Soave, M., Di Staso, U., & De Amicis, R. (2015). 3D web visualization of huge CityGML models. *International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences*, *40*.

Samet, H. (2006). *Foundations of multidimensional and metric data structures*. Morgan Kaufmann.

Schilling, A., Bolling, J., & Nagel, C. (2016). Using glTF for streaming CityGML 3D city models. Proceedings of the 21st International Conference on Web3D Technology,

Shverin, J, Rissetto, H., Remondio, F., Agugario, G., and Girardi, G. 2013. The MayaArch3D Project: A 3D WebGIS for Analyzing Ancient Architecture and Landscapes. Literary and Linguistic Computing. 28, 736-753.

She, J., Gu, X., Tan, J., Tong, M., & Wang, C. (2019). An appearance-preserving simplification method for complex 3D building models. *Transactions in GIS*, *23*(2), 275-293.

Song, Z., & Li, J. (2018). A dynamic tiles loading and scheduling strategy for massive oblique photogrammetry models. 2018 IEEE 3rd International Conference on Image, Vision and Computing (ICIVC).

Sun, J. (2019). *The Integration of 3D Geodata and BIM Data in 3D City Models and 3D Cadastre* KTH Royal Institute of Technology].

Tong, C. (2002). Quick Encoding Compression Algorithm of Octree for Modeling Three Dimensional GIS. *Journal of Image and Graphics*.

Tung, T., Jiu, X., & Stenger, B. (2020). 3D model generating system, 3D model generating method, and program. In: Google Patents.

Usta, Z. (2021). The design and development of a web-based 3D geographic information management framework.

Xiang, W., Tao, S., Liang, H., Congnan, G., & Su, G. (2021). 3D Scene Management Method Combined with Scene Graphs. *Sensors and Materials*, *34*(x), 1.

Xu, P., Chang, X., Guo, L., Huang, P.-Y., Chen, X., & Hauptmann, A. G. (2020). A survey of scene graph: Generation and application. *IEEE Trans. Neural Netw. Learn. Syst*, *1*.

Vu, T., Belussi, A., Migliorini, S., & Eldway, A. (2020). Using deep learning for big spatial data partitioning. *ACM Transactions on Spatial Algorithms and Systems (TSAS)*, *7*(1), 1-37.

Yao, Z., Nagel, C., Kunde, F., Hudra, G., Willkomm, P., Donaubauer, A., Adolphi, T., & Kolbe, T. H. (2018). 3DCityDB-a 3D geodatabase solution for the management, analysis, and visualization of semantic 3D city models based on CityGML. *Open Geospatial Data, Software and Standards*, *3*(1), 1-26.

Yan, L. (2010). Studies on R-tree spatial index for 3D GIS. *Science of Surveying and Mapping*.

Yang, S. (2007). R-tree Structure Appended with Spatial Topology Restrictions in 3D GIS. *Geomatics and Information Science of Wuhan University*.

Yang, X., Huo, L., Shen, T., Wang, X., Yuan, S., & Liu, X. (2024). A large-scale urban 3D model organisation method considering spatial distribution of buildings. *IET Smart Cities*, *6*(1), 54-64.

Yang, J., Wu, F., Lai, E., Liu, M., Liu, B., & Zhao, Y. (2021). Analysis of visualization technology of 3D spatial geographic information system. *Mobile Information Systems*, *2021*, 1-9.

Yücel, M. A. (2009). Farklı ayrıntı düzeylerinde üç boyutlu kent modelleme ve uygulanabilirliğinin araştırılmas

Zhan, W., Chen, Y., & Chen, J. (2021). 3D Tiles-Based High-Efficiency Visualization Method for Complex BIM Models on the Web. *ISPRS International Journal of Geo-Information*, *10*(7), 476.

Zhou, K., Gong, M., & Guo, B. (2012). Octree construction on graphics processing units. In: Google Patents.

Zhou, Q.-Y., Park, J., & Koltun, V. (2018). Open3D: A modern library for 3D data processing. *arXiv preprint arXiv:1801.09847*.