

Üç Boyutlu Binaların Web Üzerinde

Otomatik Olarak JOGL ile Modellenmesi

Aybars Uğur, Eray Hangül, Tahir Emre Kalaycı, Doğan Aydın

Ege Üniversitesi, Bilgisayar Mühendisliği Bölümü, 35100, Bornova / İzmir

aybars.ugur@ege.edu.tr, erayhangul@gmail.com, tahir.kalayci@ege.edu.tr, dogan.aydin@ege.edu.tr

Özet: Bu çalışmada, Java programlama dili üzerinden OpenGL yordamlarının etkin bir şekilde kullanımını sağlayan JOGL ('Java Open Graphics Library') uygulama programlama arayüzü incelenmiştir. N katlı herhangi bir binanın, JOGL ile web tabanlı olarak üç boyutlu modellenmesini sağlayan bir yazılım geliştirilmiş ve bina görselleştirmeyi otomatikleştiren bu uygulamadan elde edilen sonuçlar, üç boyutlu içerik oluşturmada kullanılan teknikler bazında tartışılarak irdelenmiştir.

Anahtar Kelimeler: Görselleştirme, Üç Boyutlu Bilgisayar Grafikleri, Web3D, Modelleme, JOGL Web Based 3D Automatic Building Generation with JOGL

Abstract: In this academic work, JOGL (Java Open Graphics Library) API which provides use of OpenGL methods through calling them from Java, is examined. A web based application that constructs automatically any building with N floor(s) is developed with JOGL. The results we obtained from our automatic building generation application are discussed through comparison of 3D content generation methods.

Keywords: Visualization, 3D Computer Graphics, Web3D, Modelling, JOGL.

1. GİRİŞ

JOGL, Java tabanlı OpenGL destekli grafiksel uygulamalar geliştirmek için, açık kaynak kod projesi şeklinde geliştirilip ortaya çıkarılmış bir programlama arayüzüdür. JOGL kullanılarak oyunlar, etkileşimli eğitim amaçlı uygulamalar, grafiksel modelleme/tasarım yapılabilen editörler vb. geliştirilmektedir.[7] [8] [9]

JOGLun çalışma mantığı, C programlama dili ile yazılmış OpenGL kütüphanesine ait yordamların, arka planda JNI (Java Native Interface) kullanılarak çağırılması çerçevesi üzerine oturtulmuştur. Dolayısıyla JOGL'un kullanılmak istendiği platform, OpenGL'in çalıştırılmasını desteklemelidir. JOGL, Java üzerinden OpenGL kullanımını sağlayan Java3D, GL4Java gibi diğer uygulama programlama arayüzlerinin aksine; OpenGL çağrılarını belirli bir kaç sınıfın içerdiği metodların prosedürel olarak

çağırılması şeklinde kullanımına imkan vermekte; sonuçta bu tarz bir yaklaşım daha hızlı bir şekilde grafiksel görüntüleme yapılmasına büyük katkı sağlamaktadır.

Yapılan bu çalışmamızda, kullanıcının belirttiği kat sayısı ve kat uzunluk, genişlik ve derinlik parametrelerine göre otomatik olarak üç boyutlu bir binanın modeli hızlı bir şekilde oluşturulmaktadır. Kullanıcılar geliştirdiğimiz bu yazılım sayesinde internet üzerinde oluşturdukları bu modelleri, üç boyutlu (3B) temel dönüşüm işlemlerini (taşımaya, döndürme, ölçeklendirme) uygulayarak etkileşimli olarak inceleyebilmektedirler.

2. Java Tabanlı 3B İçerik Oluşturma Kütüphanelerinin Karşılaştırılması

Java programlama dili ile üç boyutlu içerik oluşturmanın temelleri ilk defa 1996 yılında

Intel, Silicon Graphics, Apple ve Sun firmalarının girişimiyle başlamıştır. 1998 yılı sonlarında Java3D'nin ortaya çıkışı ile başlayan süreç, 2000 yıllarında GL4Java'nın ve 2003 - 2004 döneminde Java3D'nin gelişimine ara verildiği sırada JOGL'un ortaya çıkışı ile farklı bir boyut kazanmıştır. [6]

• JOGL vs. Java3D

Java'da, 3D içerik oluşturmada, JOGL öncesi kullanılan en yaygın uygulama programlama arayüzü JavalD'dir. Günümüzde Java3D ile geliştirilmiş bir çok grafiksel uygulama mevcuttur. Java3D'nin çalışma mantığı, ağaç veri yapısında yer alan düğüm gruplarının genel görüntüleyici tarafından belirlenen hiyerarşik yapıya göre grafiksel içeriğin oluşturulması şeklinde belirlenmiştir. Bu yaklaşım, uygulama geliştiriciler açısından programlama seviyesinde kolaylık getirirken, uygulamaların çalışma hızının düşük olmasına sebep olmaktadır. Ayrıca grafiksel öğelerin oluşturulmasında OpenGL'de olduğu kadar gerçekçilik verilememektedir. JOGL, hız ve gerçekçiliğin artırılması ve uygulamalardaki belirsiz hatalar ile uyumsuzlukların giderilmesi sonucu; Java3D'nin belirtilen eksikliklerini kapatarak, 3D içerik oluşturmada daha etkin bir çözüm durumuna gelmiştir.

• JOGL vs. LWJGL

JOGL 'a alternatif olabilecek belli başlı uygulama programlama arayüzlerinden biri de LWJGL ('Lightweight Java Game Library')dir. Bu kütüphane Microsoft DirectX 'in Sun Java tarafındaki karşılığı olarak düşünülebilir. [10]

Her iki kütüphane de arka planda yapılacak bir çok işi kendi yapıları içinde çözümlenmektedirler ancak LWJGL kendi pencere sistemi üzerinden çalıştığından var olan Java uygulamaları ile birleştirilmesi pek de kolay değildir. Ayrıca JOGL, Java'nın AWT, Swing gibi bileşenleriyle rahatlıkla birleştirilerek kullanılabilir. LWJGL grafiksel bir uygulamayı geliştirmek

için hazır bir paket gibi kullanılabilirken JOGL sadece grafiksel öğelerin görüntülenmesinde kullanılır.

Örneğin ses ve ses efekti işlemleri için JOAL('Java Open Audio Library') ve girdi kontrolü (klavye, fare, oyun çubuğu vb.) için JInput kütüphaneleri kullanılarak bütünsel sonuca gidilebilir. [2]

3. JOGL ile Web Tabanlı Etkileşimli 3B Uygulamalar Geliştirme

3.1 JOGL Kurulum

Geliştirilen uygulamaların çalıştırılması için yapılması gereken kurulum işlemleri çoğu zaman kullanıcıları ve uygulama geliştiricileri sıkıktır. Özellikle JOGL'un kurulumunda bu sıkıntıların yaşandığı görülmektedir. Kurulumda bazı püf noktalarının eksiksiz yerine getirilmesi gerekmektedir. [1]

Kurulum ile ilgili dosyaları projenin <https://iogl.dev.java.net> sitesinden indirdikten sonra aşağıdaki işlemler yapılarak geliştirme/test ortamı hazırlanabilir :

- İndirilen sıkıştırılmış haldeki dosya içerisindeki sistem kütüphanelerinin (windows için *.dll, linux için *.so uzantılı olmak üzere) Java çalışma ortamının kurulduğu dizin altındaki '/bin' dizini altına kopyalanması
- *.jar uzantılı uygulama programlama arayüzü kütüphane dosyalarının ise Java çalışma ortamının kurulduğu dizin altındaki '/lib/ext' dizini altına kopyalanması

Örneğin windows yüklü bir makinada Java çalışma ortamının yüklü olduğu dizini

C:\Java\jre1.5.0_08

olarak ele alırsak 'jogl.dll', 'jogl_awt.dll', 'jogl_cg.dll' dosyaları

C:\Java\jre1.5.0_08\bin altına; jogl.jaf dosyası ise C:\Java\jre1.5.0_08\lib\ext altına kopyalanmalıdır.

Günümüzde “Eclipse”, ‘JBuilder’, ‘IntelliJ’, ‘NetBeans’... vb. bir çok Java uygulama geliştirme editörü kullanılabilir. Kullandığımız editörden sistemimizde yüklü olan Java çalışma ortamını JOGL kurulumunda kullandığımız çalışma ortamı olarak belirlediğimiz anda, JOGL uygulamalarımızı geliştirebilecek ve test edebilecek şartlar sağlanmış olacaktır.

Kurulumun başarı ile gerçekleşip gerçekleşmediğini öğrenmek amacıyla aşağıdaki gibi basit bir test kodu çalıştırılabilir

“System.loadLibrary(“jogl”);”

Bu kod satırının, ‘UnsatisfiedLinkException’ gibi bir istisna fırlatması; Java çalışma ortamının sistem kütüphanelerini bulamaması sonucudur. Kurulum yeniden gözden geçirilmelidir.

3.2 JOGL Genel Sınıf Yapısı

Aşağıda JOGL içerisindeki temel sınıf ve arayüzler (‘interface’) kısaca açıklanmaktadır :

- **javax.media.opengl.GL** OpenGL’e erişim için kullanılan temel arayüz sınıfıdır. Grafiksel içeriği tutar.
- **javax.media.opengl. GLAutoDrawable** Olay tabanlı işleme mekanizması temelinde görüntülenme işleminin yapılmasını sağlayan arayüz sınıfıdır.
- **javax.media.opengl. GLEventListener** Uygulamada etkileşim sırasında görüntülemeyi değiştirebilecek olayların görüntülemeye olan etkisini sağlayan arayüz sınıfıdır.
- **javax.media.opengl.GLCanvas** AWT pencere sistemi üzerinden grafiksel öğelerin görüntülenmesini sağlayan ağır bir komponenti tanımlayan sınıfıdır.
- **javax.media.opengl.GLJPanel** Swing üzerinden grafiksel öğelerin görüntülenmesini sağlayan hafif bir komponenti tanımlayan sınıfıdır.
- **javax.media.opengl.GLCapabilities** Sistemde kurulu olan OpenGL üzerinden, görüntüleme aşamasında nelerin yapılması gerektiğini belirleyen sınıfıdır.

JOGL kütüphanesinde bunların haricinde bir çok sınıf bulunmaktadır. Şekil - 1’de üç adet farklı renkteki (kırmızı, yeşil ve mavi) dişli çarkın JOGL kullanılarak birbirine bağımlı olarak dönme olayını örnekleyen uygulamanın ekran görüntüsü verilmiştir :



Şekil 1. JOGL Gears Demo Ekran Görüntüsü

4. Web Tabanlı Üç Boyutlu Bina Modelleme

İnternetin hızlı bir şekilde büyümesi ile birlikte kullanıcılar, günlük hayatlarındaki bir çok işlemlerini internet üzerinden gerçekleştirmeye başlamışlardır. Bir çok firma ürün ve hizmetlerini internet üzerinden pazarlamaktadır. Bu yaklaşım; insanların, alacağı ürün veya hizmetler için görsel yönden güçlü tanıtımları talep etmelerine ön ayak olmuştur. Böylece internet üzerinden yayınlanabilecek iki veya üç boyutlu grafiksel içerik oluşturma gereksinimlerini karşılayacak yeni yapıların ortaya çıkması sağlanmıştır. Java, Java3D ve JOGLun JApplet teknolojisi üzerinden kullanılmasına imkan sağlayarak bu gereksinimi etkin bir şekilde karşılamaktadır.

İki boyutlu grafikler yıllardır internetin ve popüler yazılımların (kelime işlemciler, tablola- ma yazılımları, sunum yazılımları) doğal bir parçası olarak kullanılmaktadır. Üç boyutlu grafikler ise;

- Bilgisayar Destekli Tasarım ve Bilgisayar Destekli Üretim
- Bilim ve Bilimsel Görselleştirme
- Eğitim ve Öğretim
- Eğlence
- Reklamcılık
- Sanat
- Sanal Gerçeklik ve Güçlendirilmiş Gerçeklik gibi bir çok alanda kullanılmaktadır. [3]

Üç boyutlu modelleme genel olarak yukarıdaki başlıkların tümünde kullanılan genel uygulamalardan biridir. 3B olarak modellenecek bir binanın var olan çizim veya modelleme programlarıyla herhangi bir birey tarafından ortaya çıkarılması çok kolay olmayan bir süreçtir.

Binaların üç boyutlu olarak otomatik bir şekilde modellenmesinde prosedürel programlama yaklaşımının kullanımı son yıllarda önem kazanmaktadır. Özellikle aşağıdaki soruların tatmin edici cevaplarının elde edilmesi bu yaklaşımın doğruluğunu kanıtlayabilecek nitelikte olacaktır :

- Bu tarz uygulamalar geliştirmek için hangi şekilde prosedürel programlama yapmak gerekir?
- Gerçek zamanlı bina oluşumunu sağlayabilmek için geliştirilen kapsamlı bir uygulama yeterince hızlı sonuç üretebilir mi?
- Mimari açıdan ne kadar doğru ve görsel açıdan ne ölçüde inandırıcı sonuçlar elde edebiliriz?
- Gerçek zamanlı görüntüleme prosedürel yaklaşım aynı anda bir çok binayı farklı detay seviyesinde görüntüleyebilecek kadar yetenekli olabilir mi? [5]

Web tabanlı mimari görselleştirmede iki ve üç boyutlu teknolojileri bütünleştiren çalışmalardan birisinde de, SVG ve X3D/VRML arayüzleri kullanılmıştır. [4]

5. Çok Katlı Üç Boyutlu Bina Modelleme Çalışması

Bu makalede örneklenen çalışma ile 3B modelleme bilgisi olmayan herhangi bir bireyin; binanın kat sayısı, her kattaki asansör ve merdiven sayısı gibi temel parametreleri kullanıcı dostu bir arayüzden girerek, otomatik olarak 3B binayı oluşturabilmesi sağlanmıştır. Ayrıca fare kullanımı ile binanın herhangi bir açıdan döndürülerek görüntülenebilmesi ve klavyeden yön tuşları kullanılarak binanın pozisyonunun güncellenebilmesi imkanları sunulmuş-

tur. Böylece kullanıcı verdiği parametrelerin değişmesine göre ne tip bir 3B bina modeli ortaya çıkabileceğini hızlı ve pratik bir şekilde görebilmektedir.

Geliştirdiğimiz uygulamada, otomatik bina oluşturma yaklaşımı, önceki çalışmalardan farklı olarak; JOGL ile ilk defa web üzerinde uygulanmıştır.

Uygulamada Şekil - 2'de gösterildiği gibi kullanıcıdan bina kat sayısı ve her bir katın uzunluk, en ve tavan yükseklik değerini belirtmesi beklenmektedir.

Building Parameters	
Floor Count	2
Floor Length	4
Floor Width	4
Floor Height	2

Şekil 2. Bina kat parametreleri

Kullanıcının girdiği temel parametrelerin dışında, her bir kat için istenildiği takdirde; belirtilen sayıda, kata ait hedef noktası,merdiven ve asansör yerleştirilebilmektedir. (Şekil 3.)



Şekil 3. Binanın ilk katı için kat parametreleri

Örneğin Şekil - 3'de binanın birinci katı için 'T' harfi ile kırmızı renkte belirtilen elemanlar 2 adet hedef noktasını, 'L' harfi ile turuncu renkte belirtilen elemanlar 2 adet asansörü ve

'SC' ifadesi ile belirtilen mavi renkteki elemanlar ise merdivenleri göstermektedir. Parametrelerin dağılımı ve yerleştirilmesi kullanıcının siyah renk ile belirtilen alanı, ilgili parametre tipini seçtikten sonra tıklaması ile yerleştirilebildiği gibi; 'Random' butonuna basılarak da istenilen sayıda rastgele elemanın dağıtılması şeklinde de yapılabilmektedir.

Yukarıdaki gibi ilgili parametreler girildikten sonra binanın oluşturulması istendiğinde, öncelikle yukarıdaki Swing JPanel'inde yer alan parametrelerin koordinat değerleri JOGL'un koordinat sistemine göre normalize edilerek pozisyon değerleri güncellenmektedir. Bu işleme ait temel kod bloğu şöyledir :

X, Z eksenlerine ait normalize edilmiş koordinatların bulunması :

```
// Z eksenini için...
public float getZCoordinateValue(float
floorWidth, float normalizedZ) {
    float returnValue = -1;
    returnValue = floorWidth *
        normalizedZ * 0.5f; return
        returnValue;
}
// X eksenini için...
public float getXCoordinateValue(float
normalizedX, float floorLength) {
    float returnValue = -1;
    returnValue =
        Math.abs((floorLength/2)
        (normalizedX * floorLength));
    return returnValue;
}
```

Y eksenini için binanın her katının tavan yükseklik değeri referans kabul edildiğinden bu ekseninde herhangi bir dönüşüm işlemine gerek kalmamıştır.

Bina çiziminde öncelikle her bir kat tel kafes ('wireframe') küp şeklinde her bir kenarı farklı renkte olmak üzere çizdirilmiştir. Daha sonra her bir katın kendisine ait parametreleri sırasıyla iterasyonlarla çizdirilerek sonuca ulaşılmıştır.

Bina kat sayısı kadar iterasyonda her bir katın çizdirilmesi :

```
mainDisplayList= gl.glGenLists(1);
gl.glNewList(mainDisplayList,
GL.GL_COMPILE);
for (int i = 0;
    i<=building.getFloorCount() - 1;
    i + + )
{
    drawFloor(building.getFloor(i));
}
gl.glEndList();
```

Bir katın tabanı ile çizdirilmesini örnekleyen kod bloğu CdrawFloof) :

```
gl.glColor3d(
Math.random(), Math.random(), Math.
random());
gl.glTranslatef(0, building.
getFloorHeight(), 0);
/*
Poligon Çizici sınıfımız
kullanılarak kat çizdiriliyor...
*/
polygonDrawer.drawWireCube(
    building.getFloorLength(),
    building.getFloorHeight(),
    building.getFloorWidth());
gl.glPushMatrix();
gl.glTranslatef(0,
    building.getFloorThick()/2-
    building.getFloorHeight()/2,
    0);
gl.glScalef(1,
    building.getFloorThick(), 1);
/*
Poligon Çizici sınıfımız
kullanılarak kat tabanı
çizdiriliyor...
*/
polygonDrawer.drawSolidCube(
    building.getFloorLength(),
    building.getFloorThick(),
    building.getFloorWidth());
gl.glPopMatrix();
```

Dönüşüm İşlemi Örneği (Matris Kullanımı) :

```
gl.glPushMatrix();  
gl.glTranslatef(  
    newPoint.getX(),  
    building.getFloorHeight() /  
(MFTSPUtility.TARGET_SCALE_RATE*  
    building.getFloorHeight() / 2),  
    newPoint.getZ());  
gl.glScalef(  
    building.getFloorLength() /  
    MFTSPUtility.TARGET_SCALE_RATE,  
    building.getFloorHeight() /  
    MFTSPUtility.TARGET_SCALE_RATE,  
    building.getFloorWidth() /  
    MFTSPUtility.TARGET_SCALE_RATE);  
glut.glutSolidCube(1);  
gl.glPopMatrix();
```

Yukarıdaki kod bloğunda; var olan grafiksel içeriğin 'glPushMatrix' metodu ile korunmasından sonra 'glTranslatef' metodu ile x, y ve z eksenlerinde öncelikle taşıma dönüşüm işlemi yapılmış ve daha sonra üç eksen üzerinde de 'glScalef' metodu ile ölçeklendirme işlemi gerçekleştirilmiştir. Bir birim ayrıt uzunluğuna sahip birim küp nesnesi bu dönüşümleri kendi üzerinde yansıtabilecek şekilde 'glutSolidCube' metodu ile çizdirilerek, 'glPopMatrix' metodu ile bu işlemlerin başında korunan grafiksel içerik yeniden gösterilmektedir.

Aşağıda, uygulamanın etkinliği için kullanılan 'Display List' tekniği örneklendirilmiştir :

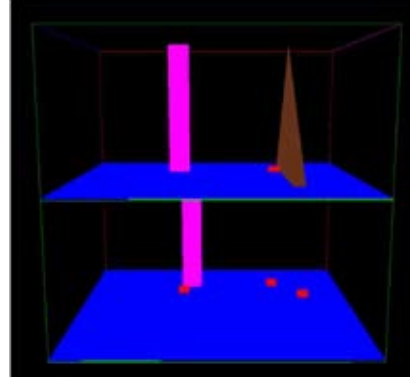
Görüntüleme Listesi Kullanımı (Display List)

```
mainDisplayList = gl.glGenLists(1);  
gl.glNewList(mainDisplayList,  
    GL.GL_COMPILE);  
for (  
    int i = 0;  
    i <= building.getFloorCount() - 1; i++) {  
    drawFloor(building.getFloor(i));  
}  
gl.glEndList();
```

Uygulamanın Ürettiği Farklı Bina Görüntü Örnekleri

2 Katlı Bina Model Örneği (Şekil 4.) :

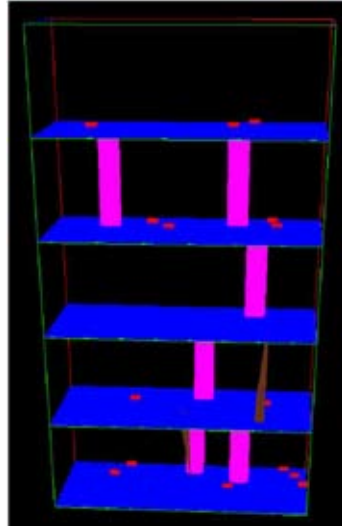
- (1. Kat : 3 adet hedef noktası, 1 adet asansör)
- (2. Kat : 1 adet hedef noktası, 1 adet asansör, 1 adet merdiven)



Şekil 4. 2 Katlı Bina Model Örneği

5 Katlı Bina Model Örneği (Şekil 5.) :

- (1. Kat : 6 adet hedef noktası, 2 adet asansör, 1 adet merdiven)
- (2. Kat : 2 adet hedef noktası, 1 adet asansör, 1 adet merdiven)
- (3. Kat : 1 adet asansör)
- (4. Kat : 4 adet hedef noktası, 2 adet asansör)
- (5. Kat : 3 adet hedef noktası)



Şekil 5. 5 Katlı Bina Model Örneği

6. Sonuçlar

6.1 Grafik Arayüzlerinin Karşılaştırılması

Uygulama geliştirilirken JOGL dışındaki grafiksel içerik oluşturma kütüphaneleri ve teknikleri gerçekleştirim sırasında kullanılan teknikler bazında irdelenerek karşılaştırılmıştır.

Elde edilen sonuçlar belirlenen başlıklar açısından Tablo - 1’de listelenmektedir :

	JOGL	Java3D	OpenGL
Grafiksel İçerik Kalitesi	Çok İyi	İyi	Çok İyi
Görüntüleme Hızı	iyi	Orta	Çok İyi
Programlama Kolaylığı	İyi	Çok İyi	Orta
Platform Uyumluluğu	İyi	İyi	Çok İyi
Web Yayınlama Kolaylığı	Çok İyi	Çok İyi	Az
Farklı Uygulamalarla Ortak Çalışabilme	İyi	İyi	Çok İyi

Tablo 1. - JOGL, Java3D, OpenGL kütüphanelerinin karşılaştırılması

Gerçekleştirilen bu uygulama sonucunda JOGL’un web tabanlı üç boyutlu içerik oluşturmada gayet başarılı olduğu anlaşılmıştır. OpenGL ve Java ortamlarının birleştirilerek web üzerinden kullanılabilmesini sağlaması, bir çok projenin daha çok kişiye; çok daha hızlı bir şekilde ulaşabilmesini sağlayacaktır. Java geliştiricileri üç boyutlu grafiksel programlamayı rahatlıkla kullanabileceklerdir.

6.2 Programın Web Tabanlı Bina Oluşturma Sürecine Etkisi

Özellikle, üç boyutlu etkileşimli olarak geliştirilen grafiksel uygulamaların web üzerinden kullanılması, anlatılmak istenen konuyu; bu konu hakkında hiç bir bilgisi olmayan kullanıcılara dahi kolay bir şekilde sunabilmesi bakımından önemli bir hale gelmiştir.

Geliştirilen uygulama Java’nın ‘Web Start’ teknolojisi kullanılarak internet üzerinden kullanılabilir hale getirilmiştir.

<http://yzgrafik.ege.edu.tr/projects/MFVis> adresinden uygulama etkileşimli olarak kullanılabilir. Bu yaklaşım, çözümün daha hızlı ve kolay bir şekilde kullanıcıya sunulması bakımından önemlidir. Günümüzde bir çok uygulama web hatta mobil tabanlı çalışabilecek şekilde geliştirilerek bilgiye erişimdeki sınırların kalktığı örneklenmektedir.

Kullanıcıların, istedikleri bir bina modelini, internet üzerinden kolaylıkla oluşturabildikleri gözlemlenmiştir. Yazılım geliştiriciler açısından ise, OpenGL altyapısının kullanılması sayesinde çok karmaşık modeller hızlı bir şekilde oluşturularak, üzerlerinde işlemler yapılabildiği anlaşılmıştır. Çalışmamızda, desen kaplama (‘Texture Mapping’), hazır 3B model yükleme (‘3D Object Loading’) ve karmaşık aydınlatma (‘Illumination Effects’) gibi modüllerin eklenmesi aşamasına gelinmiştir. Böylece uygulama, mimari ve görsel açıdan daha gerçekçi bina modellerini de destekleyecektir.

Arka planda OpenGL kullanılması, gerçekçiliği ve hızı arttırırken; nesneye dayalı programlama yaklaşımı da programlama aşamasında harcanan çabayı minimum düzeye indirerek daha etkin bir şekilde uygulama geliştirilebilmesini sağlamaktadır. Bu özellikleri sayesinde, gelecekte JOGL’un kullanım oranı artacak ve değişik alanlardaki yazılım geliştiriciler tarafından tercih edilecektir.

7. Kaynaklar

- [1]. Canroy, K., JOGL : A Beginner’s Guide and Tutorial, 2 Eylül 2004.
- [2]. Tvilleagear, D., Kesselman, J., Goldberg, A., Petersen, D., Soto, C. J., Melissinos, C., Java Technologies for Games.

[3]. Uğur, A., Bilgisayar Grafikleri Ders Notları, EGE Üniversitesi, Bilgisayar Mühendisliği Bölümü, 2002 - 2003.

[4]. Wei, Y., Integrating web 2D and 3D technologies for architectural visualization: applications of SVG and X3D/VRML in environmental behavior simulation, Web3D 2006: 11th International Symposium on 3D Web Technology, 2006.

[5]. Whelan, G., Automatic Building Generation, Institute of Technology Blanchardstown, Graphics and Gaming Group, 2006.

[6]. Xu, Z., Yen, Y., Chen, X. J., OpenGL Programming in Java .

[7]. JOGL : Java Bindings for OpenGL, <https://jogl.dev.java.net>

[8]. Nehe Productions, <http://nehe.gamedev.net>

[9]. OpenGL, <http://www.opengl.org>, SGI

[10]. Sun Developer Network, <http://java.sun.com>, Sun Microsystems