# cub3D

## miniLibX ile İlk RayCaster'ım

**Özet:** Bu proje, şimdiye kadar yaratılmış ilk FPS olarak kabul edilen dünyaca ünlü Wolfenstein 3D oyunundan esinlenmiştir. Ray-casting'i keşfetmenizi sağlayacaktır. Amacınız, bir labirentte yolunuzu bulmanız gereken dinamik bir görünüm oluşturmak olacaktır.

**Versiyon:** 11.0

---

## İçindekiler



---

## Bölüm I - Önsöz

Id Software tarafından geliştirilen, dünyaca ünlü John Carmack ve John Romero liderliğindeki ve 1992'de Apogee Software tarafından yayınlanan Wolfenstein 3D, video oyunları tarihindeki ilk gerçek "First Person Shooter"dır.

**Şekil I.1:** John Romero (solda) ve John Carmack (sağda) gelecek nesiller için poz veriyor.

Wolfenstein 3D, video oyunları dünyasında ek ebedi dönüm noktaları olan Doom (Id Software, 1993), Doom II (Id Software, 1994), Duke Nukem 3D (3D Realm, 1996) ve Quake (Id Software, 1996) gibi oyunların atasıdır.

Şimdi, Tarihi yeniden yaşama sırası sizde...

---

**Wolfenstein 3D oyunu orijinal olarak Nazi Almanyası'nda geçer, bu potansiyel olarak rahatsız edici olabilir. Bu oyunun resimleri ve tarihi size yalnızca teknik nedenlerle ve pop/geek kültür nedenleriyle sunulmaktadır, çünkü oyun her ikisi için de bir başyapıt olarak kabul edilmiştir.**

---

## Bölüm II - Hedefler

Bu projenin amaçları, bu ilk yılın tüm amaçlarına benzer: titizlik, C kullanımı, temel algoritmalar, bilgi araştırması, vb.

Bir grafik tasarım projesi olarak, cub3D bu alanlarda becerilerinizi geliştirmenizi sağlayacaktır: window'lar, renkler, event'ler, şekilleri doldurma, vb.

Sonuç olarak, cub3D, özellikleri anlamak zorunda kalmadan matematiğin eğlenceli pratik uygulamalarını keşfetmek için harika bir oyun alanıdır.

İnternette mevcut çok sayıda dokümanın yardımıyla, zarif ve verimli algoritmalar oluşturmak için matematiği bir araç olarak kullanacaksınız.

Bu size uygunsa, bu projeye başlamadan önce orijinal oyunu test edebilirsiniz: http://users.atw.hu/wolf3d/

---

## Bölüm III - Ortak Talimatlar

- Projeniz C ile yazılmalıdır.
- Projeniz Norm'a uygun olarak yazılmalıdır. Bonus file'larınız/function'larınız varsa, bunlar norm kontrolüne dahildir ve norm hatası varsa 0 alırsınız.
- Function'larınız beklenmedik bir şekilde quit olmamalıdır (segmentation fault, bus error, double free, vb.) undefined behavior hariç. Bu gerçekleşirse, projeniz işlevsel olmayan olarak kabul edilecek ve değerlendirme sırasında 0 alacaktır.
- Heap'te allocate edilen tüm memory gerektiğinde düzgün bir şekilde free edilmelidir. Memory leak'lerine tolerans gösterilmeyecektir.
- Subject gerektiriyorsa, -Wall, -Wextra ve -Werror flag'leri ile cc kullanarak kaynak file'larınızı gerekli output'a compile eden bir Makefile submit etmelisiniz. Ayrıca, Makefile'ınız gereksiz relinking yapmamalıdır.
- Makefile'ınız en azından şu rule'ları içermelidir: $(NAME), all, clean, fclean ve re.
- Projeniz için bonusları submit etmek için, Makefile'ınıza projenin ana bölümünde izin verilmeyen tüm çeşitli header'ları, library'leri veya function'ları ekleyecek bir bonus rule eklemelisiniz. Bonuslar _bonus.{c/h} file'larına yerleştirilmelidir, subject aksi belirtmedikçe. Mandatory ve bonus bölümlerinin değerlendirmesi ayrı ayrı yapılır.
- Projeniz libft'nizi kullanmanıza izin veriyorsa, kaynaklarını ve ilişkili Makefile'ını bir libft folder'ına kopyalamalısınız. Projenizin Makefile'ı, Makefile'ını kullanarak library'yi compile etmeli, ardından projeyi compile etmelidir.
- Bu çalışmanın submit edilmesine ve değerlendirilmesine gerek olmasa da, projeniz için test programları oluşturmanızı teşvik ediyoruz. Bu size çalışmanızı ve arkadaşlarınızın çalışmalarını kolayca test etme fırsatı verecektir. Bu testleri özellikle defence sırasında faydalı bulacaksınız. Gerçekten de, defence sırasında, kendi testlerinizi ve/veya değerlendirdiğiniz arkadaşınızın testlerini kullanmakta özgürsünüz.
- Çalışmanızı atanan Git repository'sine submit edin. Yalnızca Git repository'sindeki çalışma değerlendirilecektir. Deepthought çalışmanızı değerlendirmek üzere atanmışsa, bu peer-evaluation'larınızdan sonra gerçekleşecektir. Deepthought'un değerlendirmesi sırasında çalışmanızın herhangi bir bölümünde bir hata olursa, değerlendirme duracaktır.

---

## Bölüm IV - AI Talimatları

### Context

Öğrenme yolculuğunuz sırasında, AI birçok farklı görevde yardımcı olabilir. AI araçlarının çeşitli yeteneklerini ve çalışmanızı nasıl destekleyebileceklerini keşfetmek için zaman ayırın. Ancak, onlara her zaman dikkatle yaklaşın ve sonuçları eleştirel olarak değerlendirin. İster code, ister dokümantasyon, fikir veya teknik açıklamalar olsun, sorunuzun iyi formüle edildiğinden veya üretilen içeriğin doğru olduğundan asla tam olarak emin olamazsınız. Arkadaşlarınız, hataları ve kör noktaları önlemenize yardımcı olacak değerli bir kaynaktır.

**Ana mesaj**

☛ Tekrarlayan veya sıkıcı görevleri azaltmak için AI kullanın.
☛ Gelecek kariyerinize fayda sağlayacak - hem coding hem de non-coding - prompting becerileri geliştirin.
☛ Yaygın riskleri, önyargıları ve etik sorunları daha iyi tahmin etmek ve önlemek için AI sistemlerinin nasıl çalıştığını öğrenin.
☛ Arkadaşlarınızla çalışarak hem teknik hem de power skill'leri geliştirmeye devam edin.
☛ Yalnızca tam olarak anladığınız ve sorumluluğunu alabileceğiniz AI tarafından üretilen içeriği kullanın.

**Learner kuralları:**

- AI araçlarını keşfetmek ve nasıl çalıştıklarını anlamak için zaman ayırmalısınız, böylece onları etik bir şekilde kullanabilir ve potansiyel önyargıları azaltabilirsiniz.
- Prompting yapmadan önce probleminiz üzerinde düşünmelisiniz - bu, doğru kelime dağarcığı kullanarak daha net, daha ayrıntılı ve daha alakalı prompt'lar yazmanıza yardımcı olur.
- AI tarafından üretilen her şeyi sistematik olarak kontrol etme, gözden geçirme, sorgulama ve test etme alışkanlığı geliştirmelisiniz.
- Her zaman peer review aramalısınız - yalnızca kendi doğrulamanıza güvenmeyin.

**Phase çıktıları:**

- Hem genel amaçlı hem de domain'e özgü prompting becerileri geliştirin.
- AI araçlarını etkili kullanımla verimliliğinizi artırın.
- Computational thinking, problem çözme, uyum sağlama ve işbirliğini güçlendirmeye devam edin.

**Yorumlar ve örnekler:**

- Sınavlar, değerlendirmeler ve daha fazlası gibi gerçek anlayışı göstermeniz gereken durumlarla düzenli olarak karşılaşacaksınız. Hazırlıklı olun, hem teknik hem de kişilerarası becerilerinizi geliştirmeye devam edin.
- Mantığınızı açıklamak ve arkadaşlarınızla tartışmak genellikle anlayışınızdaki boşlukları ortaya çıkarır. Peer learning'i öncelik haline getirin.
- AI araçları genellikle özel bağlamınızdan yoksundur ve genel yanıtlar verme eğilimindedir. Ortamınızı paylaşan arkadaşlarınız, daha alakalı ve doğru içgörüler sunabilir.
- AI en olası cevabı üretme eğilimindeyken, arkadaşlarınız alternatif perspektifler ve değerli nüanslar sağlayabilir. Onlara bir kalite kontrol noktası olarak güvenin.

✓ **İyi pratik:** AI'ya soruyorum: "Bir sorting function'ını nasıl test ederim?" Bana birkaç fikir veriyor. Bunları deniyorum ve sonuçları bir arkadaşımla gözden geçiriyorum. Yaklaşımı birlikte geliştiriyoruz.

✗ **Kötü pratik:** AI'dan tüm bir function yazmasını istiyorum, projeme kopyala-yapıştır yapıyorum. Peer-evaluation sırasında ne yaptığını veya neden yaptığını açıklayamıyorum. Güvenilirliğimi kaybediyorum - ve projemde başarısız oluyorum.

✓ **İyi pratik:** Bir parser tasarlamak için AI kullanıyorum. Sonra mantığı bir arkadaşımla gözden geçiriyorum. İki bug yakalıyoruz ve birlikte yeniden yazıyoruz - daha iyi, daha temiz ve tamamen anlaşılmış.

✗ **Kötü pratik:** Projemimin önemli bir parçası için Copilot'un code üretmesine izin veriyorum. Compile oluyor, ama pipe'ları nasıl handle ettiğini açıklayamıyorum. Değerlendirme sırasında, justify edemiyorum ve projemde başarısız oluyorum.

---

## Bölüm V - Mandatory part - cub3D

| | |
|---|---|
| **Program adı** | cub3D |
| **Teslim edilecek file'lar** | Tüm file'larınız |
| **Makefile** | all, clean, fclean, re, bonus |
| **Argümanlar** | *.cub formatında bir map |
| **External function'lar** | • open, close, read, write, printf, malloc, free, perror, strerror, exit, gettimeofday<br>• Math library'nin tüm function'ları (-lm man man 3 math)<br>• gettimeofday()<br>• MinilibX library'nin tüm function'ları |
| **Libft authorized** | Evet |
| **Açıklama** | Bir labirentin içinin first-person perspektifinden "gerçekçi" bir 3D grafik temsili oluşturmalısınız. Bu temsili daha önce bahsedilen ray-casting prensiplerini kullanarak oluşturmalısınız. |

Kısıtlamalar şunlardır:

- miniLibX kullanmalısınız. Ya işletim sisteminde mevcut olan versiyon ya da kaynaklarından. Kaynaklarla çalışmayı seçerseniz, Ortak Talimatlar bölümünde libft için yazılanla aynı kuralları uygulamanız gerekecektir.
- Window'unuzun yönetimi sorunsuz kalmalıdır: başka bir window'a geçiş, minimize etme, vb.
- Duvarın hangi tarafa baktığına bağlı olarak değişen farklı duvar texture'ları (seçim sizin) görüntüleyin (North, South, East, West).
- Programınız zemin ve tavan renklerini iki farklı renge ayarlayabilmelidir.
- Program image'ı bir window'da görüntüler ve aşağıdaki kurallara uyar: ◦ Klavyenin sol ve sağ ok tuşları labirentte sola ve sağa bakmanızı sağlamalıdır. ◦ W, A, S ve D tuşları bakış açısını labirentte hareket ettirmenizi sağlamalıdır. ◦ ESC'ye basmak window'u kapatmalı ve programdan temiz bir şekilde çıkmalıdır. ◦ Window'un frame'indeki kırmızı çarpıya tıklamak window'u kapatmalı ve programdan temiz bir şekilde çıkmalıdır. ◦ minilibX library'nin image'larının kullanımı şiddetle tavsiye edilir.
- Programınız ilk argüman olarak .cub uzantılı bir scene description file almalıdır. ◦ Map yalnızca 6 olası karakterden oluşmalıdır: boş bir alan için 0, duvar için 1 ve player'ın başlangıç pozisyonu ve spawn orientation'ı için N, S, E veya W.

Basit bir geçerli map örneği:

```
111111
100101
101001
1100N1
111111
```

◦ Map duvarlarla kapalı/çevrili olmalıdır, değilse program bir hata return etmelidir.
◦ Map content dışında, her element türü bir veya daha fazla boş satırla ayrılabilir.
◦ Her zaman son olması gereken map content dışında, her element türü file'da herhangi bir sırada ayarlanabilir.
◦ Map dışında, bir element'ten gelen her bilgi türü bir veya daha fazla space ile ayrılabilir.
◦ Map, file'da göründüğü gibi parse edilmelidir. Space'ler map'in geçerli bir parçasıdır ve bunları handle etmek size kalmıştır. Map kurallarına saygı gösterdiği sürece her türlü map'i parse edebilmelisiniz.
◦ Map dışında, her element, type identifier'ı (bir veya iki karakterden oluşur) ile başlamalı, ardından belirli bir sırayla özel bilgileri gelmelidir:

**North texture:**

```
NO ./path_to_the_north_texture
```

· identifier: NO
· north texture'a path

**South texture:**

```
SO ./path_to_the_south_texture
```

· identifier: SO
· south texture'a path

**West texture:**

```
WE ./path_to_the_west_texture
```

· identifier: WE
· west texture'a path

**East texture:**

```
EA ./path_to_the_east_texture
```

· identifier: EA
· east texture'a path

**Floor color:**

```
F 220,100,0
```

· identifier: F
· [0,255] aralığında R,G,B renkleri: 0, 255, 255

**Ceiling color:**

```
C 225,30,0
```

· identifier: C
· [0,255] aralığında R,G,B renkleri: 0, 255, 255

**Minimalist .cub scene ile mandatory part örneği:**

```
NO ./path_to_the_north_texture
SO ./path_to_the_south_texture
WE ./path_to_the_west_texture
EA ./path_to_the_east_texture

F 220,100,0
C 225,30,0

111111111111111111111111
100000000011000000000001
101100000111000000000001
100100000000000000000001
11111111101100000111000000000001
10000000001100000111011111111111
11110111111111011100000010001
11110111111111011101010010001
11000000110101011100000010001
10000000000000001100000010001
10000000000000001101010010001
11000001110101011111011110N0111
11110111 1110101 101111010001
11111111 1111111 111111111111
```

◦ File'da herhangi bir türde herhangi bir yanlış yapılandırma ile karşılaşılırsa, program düzgün bir şekilde exit olmalı ve "Error\n"i ve ardından seçtiğiniz açık bir hata mesajını return etmelidir.

---

## Bölüm VI - Bonus part

Bonuslar yalnızca mandatory part'ınız mükemmelse değerlendirilecektir.

**Mükemmel ile doğal olarak, eksiksiz olması gerektiğini, yanlış kullanım gibi kötü hatalar durumunda bile başarısız olamayacağını kastediyoruz. Bu, mandatory part'ınız değerlendirme sırasında TÜM puanları alamazsa, bonuslarınızın tamamen GÖRMEZDEN GELİNECEĞİ anlamına gelir.**

Bonus listesi:

- Wall collision'ları.
- Bir minimap sistemi.
- Açılıp kapanabilen door'lar.
- Animated sprite'lar.
- Mouse ile bakış açısını döndürme.

**Daha sonra daha iyi oyunlar oluşturabileceksiniz, çok fazla zaman kaybetmeyin!**

**Değerlendirmeniz sırasında kullanımları justify edilebildiği sürece, bonus part'ı tamamlamak için başka function'lar kullanmanıza veya map'e symbol'ler eklemenize izin verilir. Ayrıca ihtiyaçlarınıza uyacak şekilde beklenen scene file formatını değiştirmenize de izin verilir. Akıllı olun!**

---

## Bölüm VII - Örnekler

**Şekil VII.1:** RayCasting kullanarak Wolfeinstein3D orijinal tasarım replikası.

**Şekil VII.2:** Mandatory part'a göre projenizin nasıl görünebileceğine dair örnek.

**Şekil VII.3:** Minimap, zemin ve tavan texture'ları ve ünlü bir kirpinin animated sprite'ı ile bonus part örneği.

**Şekil VII.4:** HUD, sağlık barı, gölge efekti ve ateş edebilen silah ile başka bir bonus örneği.

**Şekil VII.5:** Seçtiğiniz bir silah ve tavana bakan player ile başka bir bonus oyun örneği.

**Not:** Bu doküman telif hakkıyla korunan image'lar içerebilir. Eğitim amacıyla oluşturulan bu subject'in Fair Use yönergeleri kapsamına girdiğini düşünüyoruz.

---

## Bölüm VIII - Teslim ve peer-evaluation

Ödevinizi her zamanki gibi Git repository'nize submit edin. Defence sırasında yalnızca repository'nizdeki çalışma değerlendirilecektir. File'larınızın isimlerinin doğru olduğundan emin olmak için iki kez kontrol etmekten çekinmeyin.

Evaluation sırasında, projenin kısa bir modifikasyonu bazen istenebilir. Bu, küçük bir davranış değişikliği, yazılacak veya yeniden yazılacak birkaç satır code veya eklenmesi kolay bir özellik içerebilir.

Bu adım her proje için geçerli olmasa da, evaluation guideline'larında belirtilmişse buna hazırlıklı olmalısınız.

Bu adım, projenin belirli bir bölümünü gerçekten anlayıp anlamadığınızı doğrulamak içindir. Modifikasyon, seçtiğiniz herhangi bir development ortamında (örneğin, olağan setup'ınız) gerçekleştirilebilir ve belirli bir zaman dilimi evaluation'ın bir parçası olarak tanımlanmadıkça birkaç dakika içinde yapılabilir olmalıdır.

Örneğin, bir function veya script'te küçük bir güncelleme yapmanız, bir display'i değiştirmeniz veya yeni bilgileri depolamak için bir data structure'ı ayarlamanız istenebilir, vb.

Ayrıntılar (scope, target, vb.) evaluation guideline'larında belirtilecektir ve aynı proje için bir evaluation'dan diğerine değişebilir.

???????????? XXXXXXXXXX = 3$796ba5a53df1352e06cc7b0f3ad2a41d