**YUV (Raw Data) Çözücü**

Ali Recep KARACA, Mehmet Fırat KÖMÜRCÜ

Bilgisayar Mühendisliği Bölümü

[recepkaraca@yandex.com](mailto:recepkaraca@yandex.com) , [mehmetfiratkomurcu@hotmail.com](mailto:mehmetfiratkomurcu@hotmail.com)

*Bu proje; .yuv formatında verilen video dosyalarının içerisindeki Y ışıklılık bileşeni, U (Cb) ve V (Cr) renk bileşenlerini ayırıp, bunları RGB formatına dönüştürdükten sonra, bu oluşan RGB formatındaki pixellerin bitmatp (bmp) resim türüne çevrilmesi ve bu çevrilen resimlerin çerçeve çerçeve ekranda oynatılmasını amaçlamaktadır.*

**1. Problemin Tanımlanması**

Bu projede çözülmesi gereken ilk sorun; YUV dosya türünde kaç farklı format olduğunun çözümlenmesidir. Bizlerden projede istenildiğine göre üç farklı format bulunmaktadır. Bunlar; 4:4:4, 4:2:2 ve 4:2:0 formatlarıdır. Bu formatların birbirlerine göre farkları; .yuv uzantılı dosya içerisinde Y,U,V değerleri okuması yapılırken hangi satırların Y, hangi satırların U ve V değerlerine karşılık geldiğinin değişmesidir. Okunması gereken dosyanın hangi formata karşılık geldiği biliniyor ise o formata göre okuma işlemi yapılmalıdır.

**2. Giriş**

Verilen projenin konusu; bir arayüz içerisinden YUV uzantılı dosyanın 4:4:4, 4:2:2, 4:2:0 formatlarından hangisi olduğunun alınması, YUV dosyasının çözünürlük bilgisininin alınması, daha sonra alınan bu YUV dosyasının verilen bilgiler doğrultusunda parse işlemine tabi tutulması ve bu parse işlemi sonrasında oluşan görüntünün .bmp uzantılı bir şekilde kaydedilerek ekranda oynatılmasıdır.

Projede YUV renk türünü RGB formatına çevirmek için kullanılmış olan formüller şunlardır:

R = 1.164 (Y - 16) + 1.596 (V – 128) **(1)**

G = 1.164 (Y – 16) – 0.813 ( V – 128) – 0.391 (U – 128) **(2)**

B = 1.164 (Y – 16) + 2.018 (U – 128) **(3)**

Yukarıda görülen formüller içerisindeki U ve V değerlerinin olduğu kısımlar, projede bizlerden sadece Y değerlerinin kullanılması istenildiğinden dolayı alınmamış olunup, sadece Y kullanıldığı kısımlar kullanılmıştır.

Verilen projenin amacı; YUV türündeki bir video dosyasının parse işlemine tabi tutulması gerektiğinden dolayı, YUV video dosyalarının nasıl bir yapıya sahip olduğu, bu video dosyalarının öğrenci tarafından okunabilmesini sağlamak, yine bu video dosyası içerisinde Y,U,V değerlerinin RGB formatına dönüştürülebilmesi ve bu son dosyanın ekran oynatılmasını sağlamaktır. Bu amaçlar doğrultusunda projenin genel amacı ise; öğrencilerin, video dosya türü olan .YUV dosyaları üzerinde tam hakimiyet kurmalarını sağlamak, kodlu halde verilen dosyaların içerisindeki kodların çözülerek ekran oynatma aşamalarının da öğrencinin kavramasını sağlamaktır.

**3. Temel Bilgiler**

Projeyi gerçekleştirme aşamasında C# programlama dili kullanılmıştır. Ayrıca; yapmış olduğumuz proje; C# programlama dili kullanılarak nesneye yönelik programlama mantığında oluşturulmuştur.

Yazılan bu program Windows 10 üzerindeki Visual Studio Community derleyicisi üzerinde derlenmiş olup, derleme aşamasında herhangi bir hatayla karşılaşılmamıştır. Ayrıca derlenen bu program yine Windows 10 üzerinde çalıştırılmış ve programın çalışması sırasında yine herhangi bir hatayla karşılaşılmamıştır.

**4. Genel Yapı**

Bu projede ilk olarak; bir arayüz oluşturuldu ve içerisine YUV videosunun çözünürlüğünün girilebileceği en-boy bilgilerini tutan iki adet “textfield” yerleştirildi. Ardından, seçilecek olan YUV videosunun hangi YUV formatında olduğunu belirlemek adına üç adet radio button eklendi ve bu radio buttonların etiketleri sırasıyla “4:4:4”, “4:2:2” ve “4:2:0” olarak ayarlandı. Bu radio buttonların ardından, seçilecek YUV dosyasının yolunun gözükeceği bir “textfield” daha eklendi ve bu “textfield” ın içine dosya yolunun eklenebilmesi için dosyanın seçilmesi gerektiğinden, bu dosyanın seçimini sağlayacak olan bir “button” daha yerleştirildi. Bunların ardından, tüm işlemleri başlatacak olan bir “button” daha form içerisine konuldu ve bu butonun yazısı “Convert .YUV to .BMP” şeklinde ayarlandı. Tüm bunların haricinde, gerekli işlemler bittikten sonra oluşan resim çerçevelerinin ekran gösterilebilmesi adına “picturebox” eklendi ve bu “picturebox” ın altına, oynatılacak olan çerçevelerin oynatılıp durdurulabilmesini sağlayan iki buton daha koyuldu. Bu butonların yazıları sırasıyla “Start” ve “Stop” olarak ayarlandı.

Form ekranı ayarlandıktan sonra, kullanıcının YUV video dosyasının çözünürlük bilgisinin girileceği en ve boy bilgilerinin tutulmakta olduğu “textfield” lar için sadece tam sayı değerinin girilmesi, boş bırakılamaması sağlandı. Ayrıca YUV video dosyasının seçileceği kısımda da, seçilen video dosyasının türünün .yuv olduğu ve seçilen dosya yolunun boş dönmediği kontrolleri de yapılmış olunup, programın çalışması sırasında yaşanılacak problemlerin çözümlenmesi amaçlanmıştır.

Projenin neredeyse tüm işlevlerinin yerine getiren “renderParse” fonksiyonu içerisinde ilk olarak; önceden kullanıcı tarafından seçilmiş olan dosyanın yolu bir değişken içerisine aktarılmıştır. Ardından, oluşturulacak olan “.bmp” türünde bitmap çerçeve dosyalarının içerisine konulacağı bir “frames” isimli klasörün oluşturulması sağlanmıştır. Burada, eğer bu isimli bir klasör zaten daha önce oluşturulmuşsa klasörün yeniden oluşturulmaması sağlanmıştır. Daha sonra videonun kullanıcı tarafından alınmış olunan yolu içerisindeki video dosyasının okunması sağlanmıştır. Burada, eğer böyle bir dosya bulunmuyorsa “File not found!” şeklinde bir hata kullanıcıya gösterilmekte olup, programın geri kalan kısmında karşılaşılabilinecek bir problemin çözümlenmesi sağlanmıştır. Bu hata ile karşılaşan kullanıcı dosya yolunu tekrardan seçmelidir.

Kullanıcı tarafından seçilmiş olan dosyanın kontrolleri yapıldıktan sonra, bu dosyanın okuma işlemine geçilmiştir. Burada karşılaşılan ilk sorun, “.yuv” formatında olan bu dosya içerisinde bulunan verilerin hexadecimal olarak kodlanmış olmasından kaynaklanmaktadır. Burada dosya okuma işlemi yapılırken, C# içerisinde bulunan “BinaryReader” sınıfından yararlanılmış olunup, bu sınıf ile alınan değerler bir “bytes” isimli diziye aktarılarak okunan tüm bu hexadecimal pixel değerleri bu dizi içerisinde saklanmaktadır. Burada, bu dizi içerisinde okunmakta olan YUV dosyasının içerisinde bulunan tüm pixel değerleri saklanmaktadır. Daha sonra, dosyanın işlenmesi kısmında, kullanıcı tarafından belirlenen .yuv dosya formatına göre (4:4:4, 4:2:2, 4:2:0) bu pixeller üstünde atlama işlemleri yapılacaktır.

Kullanıcı tarafından girilmiş olunan en ve boy bilgisi çarpılarak video dosyası içerisindeki her çerçevenin çözünürlük bilgisine ulaşılmıştır. Daha sonra, YUV dosyası RGB türüne dönüştürüleceğinden dolayı, sırasıyla “R”, “G” ve “B” isimli tek boyutlu diziler oluşturulmuştur.

Kullanıcının seçmiş olduğu format bilgisine bakılarak (4:4:4, 4:2:2, 4:2:0) frame sayısını hesaplamak için kullanılacak olan “multiplier” ve bizlerden sadece YUV dosyası içerisindeki Y değerlerinin okunması istenildiğinden dolayı sadece Y değerlerine ulaşılmasını, U ve V değerlerinin atlanmasını sağlayan “jumpValue” isimli değişkenlerin değer atamaları yapılmıştır.

Eğer format olarak 4:4:4 formatı seçilmişse; bu YUV formatı içerisinde bir oranında Y, bir oranında U ve yine bir oranında V değerleri bulunmaktadır. Bunun için, eğer sadece Y değerlerine ulaşılmak isteniyorsa, Y değerlerinin iki katı uzunluğunda olan U ve V değerlerinin atlanması gerekmektedir. Bunun için “jumpValue” isimli değişken “(2 x resolution)” şeklinde belirlenmiştir. Buradaki “resolution” isimli değişken ise en ve boy bilgisi çarpılarak bulunan, her Y,U,V çerçevesinin kapladığı toplam pixel sayısını vermektedir. Burada, her biri aynı oranda Y,U ve V çerçeveleri olduğundan dolayı, toplam çerçeve sayısını bulmamızı sağlayacak olan “multiplier” isimli değişken “3” olarak belirlenmiştir.

Eğer format olarak 4:2:2 formatı seçilmişse; bu YUV formatı içerisinde iki oranında Y, bir oranında U ve yine bir oranında V değerleri bulunmaktadır. Bunun için, eğer sadece Y değerlerine ulaşılmak isteniyorsa, Y değerlerinin yarım katı uzunluğunda olan U ve V değerlerinin atlanması gerekmektedir. Bunun için “jumpValue” isimli değişken “(resolution)” şeklinde belirlenmiştir. Burada, iki oranında Y, bir oranında U ve bir oranında V çerçeveleri olduğundan dolayı, toplam çerçeve sayısını bulmamızı sağlayacak olan “multiplier” isimli değişken “2” olarak belirlenmiştir.

Eğer format olarak 4:2:0 formatı seçilmişse; bu YUV formatı içerisinde dört oranında Y, bir oranında U ve yine bir oranında V değerleri bulunmaktadır. Bunun için, eğer sadece Y değerlerine ulaşılmak isteniyorsa, Y değerlerinin çeyrek katı uzunluğunda olan U ve V değerlerinin atlanması gerekmektedir. Bunun için “jumpValue” isimli değişken “(resolution / 2)” şeklinde belirlenmiştir. Burada, dört oranında Y, bir oranında U ve bir oranında V çerçeveleri olduğundan dolayı, toplam çerçeve sayısını bulmamızı sağlayacak olan “multiplier” isimli değişken “1.5” olarak belirlenmiştir.

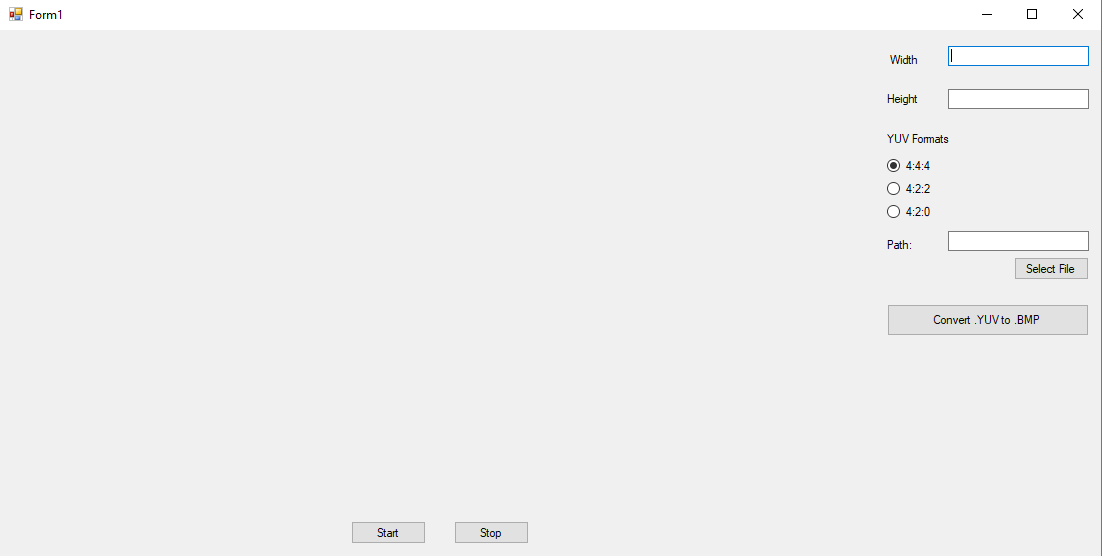
Dosya format bilgisi alındıktan sonra, çerçeve miktarını belirlemek için “(toplam pixel) / (resolution \* multiplier)” formülünden yararlanılmıştır.

YUV dosyasının RGB türüne çevrilip kaydedilmesi aşamasında ilk olarak toplam pixel uzunluğunda dönecek bir döngü oluşturulmuştur. Bu döngü içerisinde okunan her pixel için döküman içerisinde daha önce belirtilmiş olunan **(1)**, **(2)** ve **(3)** denklemlerinden yararlanılarak her pixel YUV formatından RGB formatına dönüştürülmüştür. Burada sadece Y çerçevelerinin okunması gerektiğinden dolayı, daha önce belirlenmiş olan “jumpValue” değişkeninin değerine göre pixeller içerisinde okuma işlemi yapılırken atlama yapılması sağlanmıştır. Daha sonra bu RGB türündeki pixeller çerçeve çerçeve bitmap türündeki resim dosyalarına kaydedilmiştir.Tüm bu dosyalar daha sonra içerisinden okuma yapılacak olan “frames” klasörüne kaydedilmektedir.

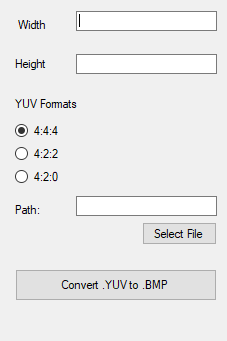
Bu dosyaların okunması aşamasında, bitmap türündeki resim çerçeveleri, bir zamanlayıcı yardımıyla sırasıyla “picturebox” içerisinde gösterilmekte olup, sanki video oynuyormuş gibi bir izlenim oluşturulmaktadır. Ayıca oynatılan bu video butonlar yardımıyla durdurulup yeniden başlatılabilmektedir.

**5. Proje Hakkında Diğer Veriler**

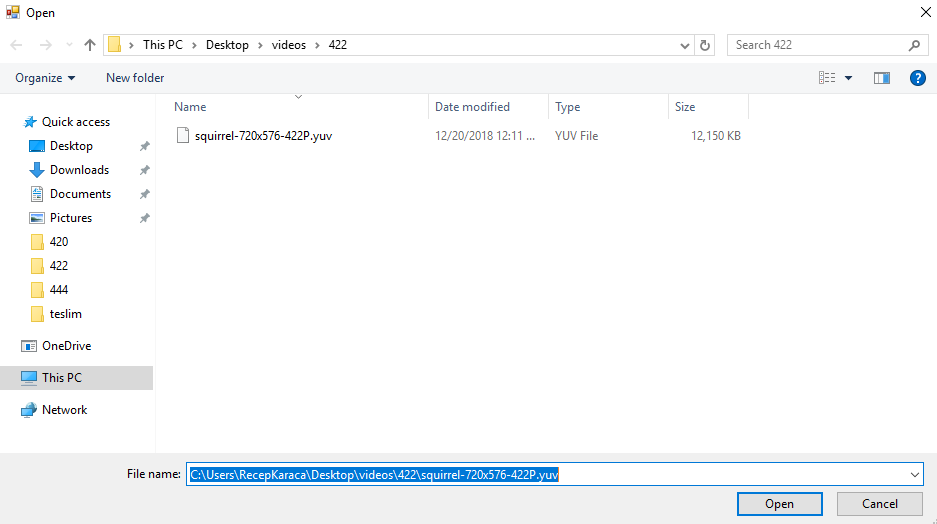
**5.1 Programın Çalışması Sırasından Bazı Görseller:**



*Şekil 1*: Karşılama Ekranı



*Şekil 2*: Ayarlar Kısmı



*Şekil 3*: Dosya Seçim Ekranı



*Şekil 4*: Videnun oynatılması

**6. Sonuçlar**

* Kullanıcıdan YUV dosyasının en ve boy değerleri istenir.
* Kullanıcıdan dosya formatı istenir.(Seçenekler: 4:4:4, 4:2:2, 4:2:0).
* Kullanıcıdan dosyanın seçilmesi istenir.
* Seçilen YUV dosyası RGB türüne dönüştürülür.
* Dönüştürülen dosya belirli zaman aralığıyla kullanıcıya frameler halinde gösterilir.
* YUV dosyasının en ve boy değerleri tam sayı olarak alınır.

Yukarıdaki değerlendirmeler ışığında, yapmış olduğumuz program, proje sırasında bizlerden istenen tüm adımları yerine getirmekte olup, hiçbir eksik fonksiyonu bulunmamaktadır.

**6. Kazanımlar**

* Resim türleri hakkında temel bilgiler elde edildi.
* .yuv uzantılı dosya türlerinin diğer dosya türlerinden farkı öğrenildi.
* C# dili hakkında temel bilgiler öğrenildi.
* YUV dosya formatları ve bu formatlar arasındaki farklar öğrenildi.
* RGB renk uzayı hakkında yeni bilgiler elde edildi.
* Fotoğraflarda en ve boy bilgilerinin ne işe yaradıkları öğrenildi
* Framelerden nasıl bir video oluşturulabileceği hakkında bilgiler elde edildi.

**7. Akış Şeması**

****

*Şekil 5*: Akış Şeması

**8. Referanslar**

http://www.15rec.com/yuvycbcr-nedir/ (Access Date : 15.12.2018)

https://www.w3schools.com/colors/colors\_rgb.asp (Access Date : 16.12.2018)

https://docs.microsoft.com/tr-tr/dotnet/api/system.drawing.bitmap (Access Date : 16.12.2018)

https://stackoverflow.com/questions/9744026/image-sequence-to-video-stream (Access Date : 16.12.2018)

https://docs.microsoft.com/tr-tr/dotnet/csharp/programming-guide/file-system/how-to-read-from-a-text-file (Access Date : 16.12.2018)

https://www.c-sharpcorner.com/UploadFile/mahesh/textbox-in-C-Sharp/ (Access Date : 17.12.2018)

https://www.youtube.com/watch?v=UyesZMeBcLM (Access Date : 17.12.2018)

https://www.youtube.com/watch?v=KKX08oOTMkk (Access Date : 17.12.2018)

https://www.youtube.com/watch?v=15aqFQQVBWU (Access Date : 17.12.2018)

https://code2flow.com/app (Access Date : 17.12.2018)

https://docs.microsoft.com/en-us/windows/desktop/medfound/recommended-8-bit-yuv-formats-for-video-rendering (Access Date : 17.12.2018)