

KOCAELİ ÜNİVERSİTESİ
MÜHENDİSLİK FAKÜLTESİ

İMGE ÇÖZÜNÜRLÜK ARTTIRMA (ARADEĞERLEME)

MÜHENDİSLİK TASARIMI – 2

Final Raporu

Mahmut Kocaman

140208033

Bölümü : Elektronik ve Haberleşme Mühendisliği

Danışman : Doç. Dr. M. Kemal Güllü

KOCAELİ, 2019

İÇİNDEKİLER

İÇİNDEKİLER	i
1.GİRİŞ	1
1.1.Görüntü İşleme.....	1
1.2. Görüntü Ölçekleme	3
1.3. Bilinear Enterpolasyon (Bilinear İnterpolation).....	5
1.4. En Yakın Komşu Enterpolasyonu (Nearest Neighbor Interpolation)	7
2.PROJEDE YAPILAN ÇALIŞMALAR	9
2.1. Kavramsal Tasarım	9
2.2. İş Zaman Çizelgesi.....	10
2.3. Deneysel Çalışmalar (Bilinear Enterpolasyon Yöntemi İçin).....	12
2.4. Deneysel Çalışmalar (En Yakın Komşu Enterpolasyon Yöntemi İçin).....	17
2.5. Proje İçin Yazılan Matlab Kodu (Bilinear Enterpolasyon Yöntemi İçin).....	19
2.6. Proje İçin Yazılan Matlab Kodu (En Yakın Komşu Enterpolasyon Yöntemi İçin)	21
3. SONUÇLAR	22
3.1. Hata – Benzerlik Oranı (Bilinear Enterpolasyon Yöntemi İçin).....	23
3.2. Hata – Benzerlik Oranı (En Yakın Komşu Enterpolasyonu Yöntemi İçin)....	24
KAYNAKLAR	25

1.GİRİŞ

1.1.Görüntü İşleme

Görüntü İşleme, görüntüyü dijital form haline getirmek ve bazı işlemleri gerçekleştirmek için geliştirilmiş, spesifik görüntü elde etmek veya ondan bazı yararlı bilgiler çıkarmak için kullanılan bir yöntemdir. Bu yöntemin girdisi video kesiti veya fotoğraf gibi bir görüntüdür. Çıktısı ise görüntünün istenilen ya da dikkat edilmesi gereken bölümüne karşılık gelir. Genellikle Görüntü İşleme sistemi, önceden belirlenmiş sinyal işleme (Signal Processing) yöntemlerini uygularken görüntüleri iki boyutlu sinyaller olarak ele alır.

Günümüzde işletmelerin çeşitli yönleriyle kullandıkları görüntü işleme sistemleri hızla büyüyen teknolojiler arasında yer alır. Görüntü İşleme, mühendislik ve bilgisayar bilimleri disiplinlerinde de temel araştırma alanını oluşturur.

Görüntü işleme temel olarak aşağıdaki üç adımı içerir.

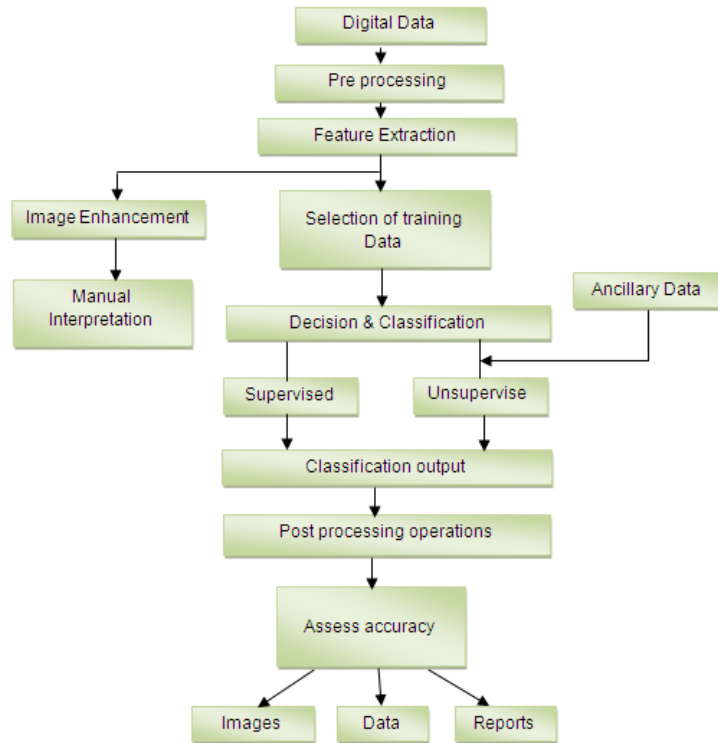
- Görüntünün optik tarayıcı ile veya dijital fotoğraflarla alınması.
- Veri sıkıştırma, görüntü iyileştirme ve uydu fotoğrafları gibi insan gözü olmayan lekelenme kalıplarını içeren görüntüyü analiz etme-kullanma.
- Çıktı, sonuçların görüntü analizine dayalı olarak değiştirilmiş, kullanıma hazır hale getirme.

Görüntü işlemenin amacı 5 gruba ayrılmıştır:

1. Görselleştirme – Görünmesi zor nesneleri gözlemleme
2. Görüntü keskinleştirme ve restorasyon – Gürültülü görüntüleri iyileştirme
3. Görüntü alımı – İlgi çekici ve yüksek çözünürlüklü görüntü arama
4. Desen Tanıma – Bir görüntüdeki çeşitli nesneleri tanımlama.
5. Görüntü Tanıma – Bir görüntüdeki nesneleri ayırt etme.

Görüntü İşleme için kullanılan iki yöntem, Analog ve Dijital Görüntü İşleme yöntemidir. Fotokopiler ve fotoğraflar gibi basılı kopyalar için analog veya görsel görüntü işleme teknikleri kullanılabilir. Görüntü analistleri, bu görsel teknikleri kullanırken yorumlamayı çeşitli temellere oturturlar. Görüntü işleme sadece teknik bilgi ile sınırlandırılmamalı, mühendislerin hayal gücü ve düşünce yeteneğine de dayanmalıdır. Görsel tekniklerle görüntü işleme alanındaki bir diğer önemli araç ise ham veri yani geçmişte toplanmış ve işlenmemiş görüntüdür. Analistler, tanımlamak istedikleri ürünler ile ilgili geçmiş işlemleri sisteme öğretir. Bir derin öğrenme kolu olarak Görüntü İşleme işlemi, geçmiş verilerin ışığında çalışır.

Dijital İşleme teknikleri dijital görüntülerin bilgisayarlarla manipüle edilmesine yardımcı olur. Uydu platformundan alınan görüntüler, algılayıcı hatası nedeniyle eksiklik içerir. Bu kusurları aşmak ve bilginin özgünlüğünü elde etmek için, çeşitli işleme aşamalarından geçmek zorundadır. Her türlü verinin dijital tekniği kullanırken geçmesi gereken üç genel aşama vardır; Ön-işleme, geliştirme ve görüntüleme, bilgi çıkarımıdır.



Şekil 1.1. Görüntü İşleme Aşamaları [1]

1.2. Görüntü Ölçekleme

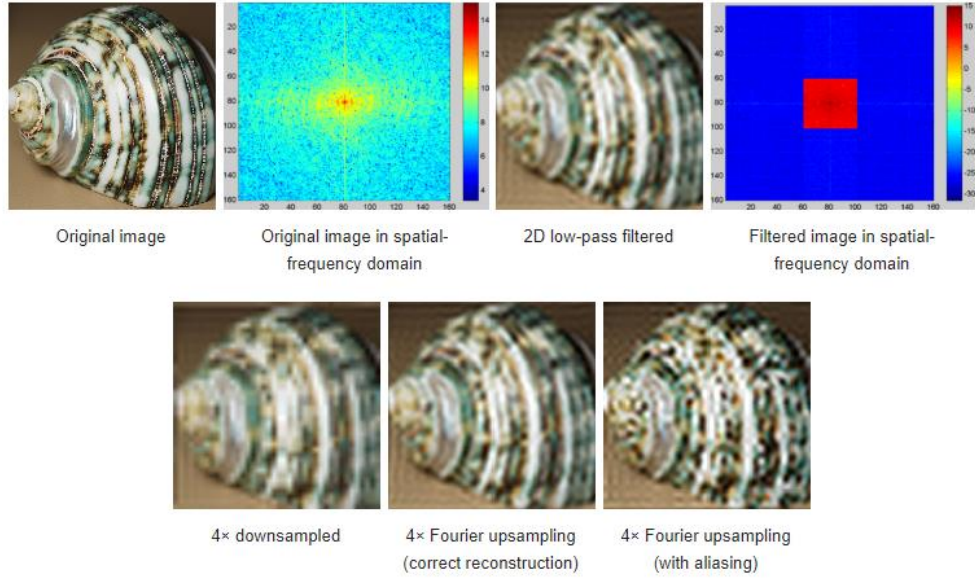
Bu projede, Görüntü İşlemenin Görselleştirme (Görünmesi zor nesneleri gözlemleme) ve Görüntü alımı (İlgi çekici ve yüksek çözünürlüklü görüntü arama) konularında kullandığımız görüntü ölçekleme aşamasını ele alacağız.

Resimlerimizin tümü, ihtiyaç duyduğumuz tam boyut değildir, elimizdeki görüntüde görmemiz gereken bir ayrıntıyı göremiyor olabiliriz. Bu yüzden bir resmin nasıl düzgün bir şekilde yeniden boyutlandırılacağını ve yeniden boyutlandırmanın nasıl çalıştığını anlamak önemlidir. Bir resim yeniden boyutlandırıldığında, piksel bilgisi de değişir. Örneğin, bir görüntünün boyutu küçültülür, gereksiz piksel bilgisi atılır. Bir resim büyütüldüğünde, tipik olarak çok pikseli veya çok yumuşak ve bulanık görünümlü bir görüntüye neden olan daha büyük bir boyuta ulaşmak için en iyi tahminlere dayanarak yeni piksel bilgileri oluşturmalı ve eklemelidir.

Yüksek kaliteli (yayıncılık) veya geniş formatlı (poster) baskılar için bir görüntü gerekiyorsa, büyütme zorluğundan dolayı mümkün olan en yüksek çözünürlük ve kalitede yakalandığından emin olunması gerekir.

Görüntü ölçekleme, bilgisayar grafik ve dijital görüntüleme için, dijital görüntünün yeniden boyutlandırılmasını ifade eder. Video teknolojisinde, dijital malzemenin büyütülmesi, yükseltme veya çözünürlük geliştirme olarak bilinir.

Görüntü ölçeklendirme matematiksel olarak, Nyquist örnekleme teoremi bakış açısıyla görüntü yeniden örnekleme veya görüntü yeniden oluşturma şekli olarak yorumlanabilir. Teoriye göre, daha yüksek çözünürlüklü bir orijinalden daha küçük bir görüntüye altör örnekleme işlemi, aliasing artifact larını önlemek için sadece uygun bir 2D kenar yumuşatma filtresi uygulandıktan sonra gerçekleştirilebilir. Görüntü, daha küçük görüntü tarafından taşınabilecek bilgilere indirgenir. Örnekleme durumunda, bir yeniden yapılandırma filtresi, kenar yumuşatma filtresinin yerini alır.



Şekil 1.2. Görüntünün çeşitli aşamalardan geçirilmiş halleri [2]

Görüntü ölçeklendirmenin çeşitli yöntemleri vardır. Bunlar : En yakın komşu enterpolasyonu, bilinear ve bicubic algoritmalar, Sinc ve Lanczos örnekleme, kutu örnekleme, mipmap, Fourier dönüşümü yöntemleri, kenar yönelimli enterpolasyon, HQX, Vektöriyal, derin evrişimli sinir ağları yöntemi olarak sıralanabilir. Bu projede kullanılacak yöntemler, en yakın komşu enterpolasyonu (Nearest-neighbor interpolation) ve bilinear enterpolasyon (Bilinear İnterpolation) dur. Ben projenin şu an ki aşamasında görüntü ölçeğinin büyütülmesi aşamasında olduğum ve ölçeklendirmede bilinear enterpolasyon kullandığım için, bu raporda bilinear enterpolasyondan bahsedeceğim.

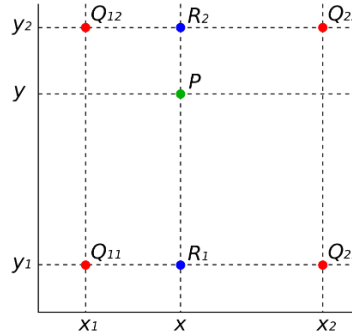
1.3. Bilinear Enterpolasyon (Bilinear İnterpolation)

Bilinear Enterpolasyon, komşu pikselleri kullanarak pikseller arasındaki boşlukları doldurduğumuz basit bir enterpolasyon tekniğidir.

Bilinear enterpolasyon, piksel renk değerlerini enterpolasyon yaparak çalışır, orijinal görüntünün ayırık geçişleri olsa bile çıktıya sürekli bir geçiş sağlar. Bu, sürekli tonlu görüntüler için arzu edilmesine rağmen, bu yöntem, kontrastı (keskin kenarlar) istenmeyecek şekilde azaltır.

Bilinear enterpolasyon, bir alt piksel değerini tahmin etmek için birinci dereceden, iki boyutlu bir polinom-düzlem kullanır.

Örneğin, dört piksel arasında bilinmeyen bir pikselimiz var ve diyelim ki bilinmeyen pikselin $f(x, y)$ olduğunu ve dört pikselle çevrili olduğunu söyleyelim:



Şekil 1.3. Bilinear Enterpolasyonda hesaplanacak olan alt pikselin gösterimi. [3]

$$Q_{11} = (x_1, y_1)$$

$$Q_{12} = (x_1, y_2)$$

$$Q_{21} = (x_2, y_1)$$

$$Q_{22} = (x_2, y_2)$$

Bu dört komşu pikselin hepsi biliniyor, şimdi Bilinear Enterpolasyon kullanarak bu bilinmeyen pikselin değerlerini bulabiliriz. Şimdi, her şeyden önce, yalnızca x yönünde hareket edeceğiz.

X faktörü için Bilinear Interpolation için kullanılan formül:

$$f(x, y_1) \approx \frac{x_2 - x}{x_2 - x_1} \cdot f(Q_{11}) + \frac{x - x_1}{x_2 - x_1} f(Q_{21})$$

$$f(x, y_2) \approx \frac{x_2 - x}{x_2 - x_1} \cdot f(Q_{12}) + \frac{x - x_1}{x_2 - x_1} f(Q_{22})$$

Şimdi bu x formüllerini hesapladıktan sonra y yönünde hareket edeceğiz ve formüller:

$$f(x, y) \approx \frac{y_2 - y}{y_2 - y_1} f(x, y_1) + \frac{y - y_1}{y_2 - y_1} f(x, y_2)$$

$$f(x, y) \approx \frac{y_2 - y}{y_2 - y_1} \left(\frac{x_2 - x}{x_2 - x_1} f(Q_{11}) + \frac{x - x_1}{x_2 - x_1} f(Q_{21}) \right) + \frac{y - y_1}{y_2 - y_1} \left(\frac{x_2 - x}{x_2 - x_1} f(Q_{12}) + \frac{x - x_1}{x_2 - x_1} f(Q_{22}) \right)$$

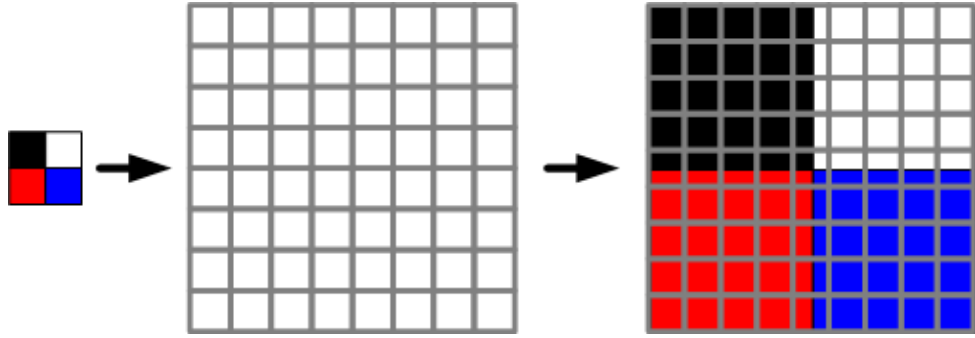
$$f(x, y) = \frac{1}{(x_2 - x_1) \cdot (y_2 - y_1)} (f(Q_{11})(x_2 - x)(y_2 - y) + f(Q_{21})(x - x_1)(y_2 - y) + f(Q_{12})(x_2 - x)(y - y_1) + f(Q_{22})(x - x_1)(y - y_1))$$

Şimdi bu formülleri kullanarak bilinmeyen piksel $f(x, y)$ 'yi Bilinear enterpolasyonunu kullanarak kolayca bulabiliriz.

1.4. En Yakın Komşu Enterpolasyonu (Nearest Neighbor Interpolation)

En yakın komşu enterpolasyonu, enterpolasyona en basit yaklaşımdır. Bazı ağırlıklandırma kriterleri ile ortalama bir değer hesaplamak veya karmaşık kurallara dayalı bir ara değer üretmek yerine, bu yöntem basitçe “en yakın” komşu pikseli belirler ve yoğunluğunun değerini alır.

Örneğin, küçültülmüş, 2×2 piksel görüntüsünü (X) aşağıda gösterildiği gibi yeniden boyutlandırmak istediğimizi ve böylece daha büyük 9×9 görüntü kalıbında, Y'yi sağa sığdırmak istediğimizi varsayalım.



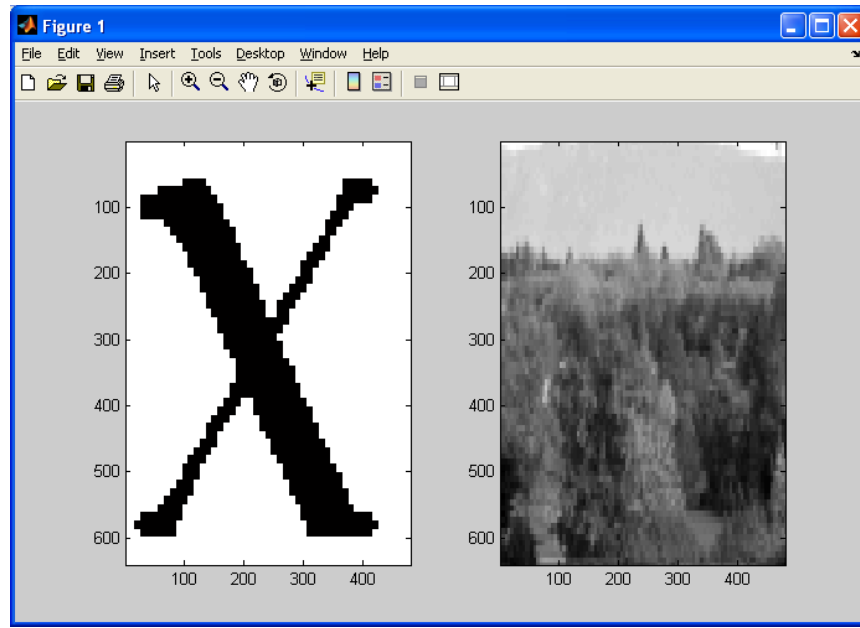
Yukarıda gösterildiği gibi, bütünleyici olmayan bir faktörle yani tam kat sayı olmayan bir oranda yeniden boyutlandırdığımızda pikseller basitçe sütun / satır ile klonlanamaz yani bunları enterpolasyona sokmamız gerekir. Örneğin en sağdaki görüntüden dikey ve yatay bir çizgi oluşturan kareler (pikselleri temsil eder), farklı renk değerleri içerebilir. Sadece tek bir renk değeri içerebilirler.

Bu yöntemin avantajları, algoritmanın basitliği ve boyutlandırma işleminin tamamlanmasının çok az zaman almasıdır. Dezavantajı ise, hızına ve sadeliğine rağmen kalitesiz görüntüler üretme eğiliminde olmasıdır. Çok basit 2 boyutlu görüntülerde kusursuz şekilde boyutlandırma yapabilmesine rağmen, fotoğraflar gibi görüntüler aşağıda gösterildiği gibi belirgin bir netlik kaybı olmasa da “bloklu” olarak ortaya çıkıyor. Buradaki kilit nokta, bunun basit ve hızlı bir algoritma olduğudur. Aşağıda, bu algoritmanın sakıncalarını gösteren iki örnek daha bulunmaktadır.

X



Şimdi, en yakın komşu algoritmasını uyguladıktan sonra aşağıdaki sonuçları elde ediyoruz:



Yukarıdaki sonuçlarda görebileceğimiz gibi sonuçlar düşük kalitededir. Orijinal görüntünün netliği korunsa da 'X' in solundaki görüntünün “pürüzlü” olduğunu ve sağdaki görüntünün pikseli ve çarpık görüldüğünü fark ettik

2.PROJEDE YAPILAN ÇALIŞMALAR

2.1. Kavramsal Tasarım

1. Matlab Öğrenme Aşaması

İlk aşama öncelikle Matlab programını öğrenmek olacak. Öğrenme aşamasında kaynaklar temin edilecek. Bunlar; matlab konu anlatım kitapları, udemy matlab kursları, Youtube matlab videoları olarak sıralanabilir.

2. Yöntem Öğrenme Aşaması

Bilinear Enterpolasyon (Bilinear Interpolation) ve En Yakın Komşu Enterpolasyonu (Nearest-neighbor interpolation) araştırılacak ve bu yöntemler hakkında bilgi edinilecek.

3. Enterpolasyon Yöntemlerinin Matlabda Kullanılması

Bu yöntemlerden elde edilen bilgiler doğrultusunda ve kullanılacak formüller ile matlab da gerekli döngüler ve fonksiyonlarla proje gerçekleştirilecektir.

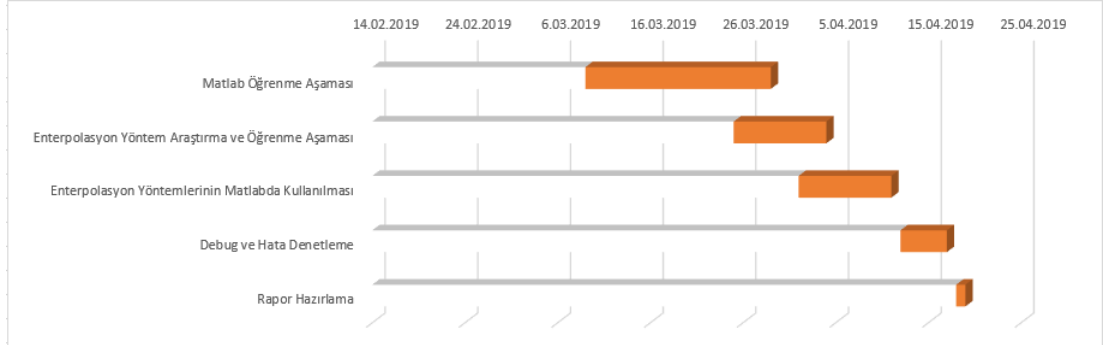
4. Debug ve Hata Denetleme

Proje gerçekleştirildikten sonra eğer varsa hatalar düzeltilecek ve gerekli düzenlemeler yapılacaktır.

5. Rapor Yazma Aşaması

Bu aşamaların hepsini gerçekleştirdikten sonra tüm ayrıntılarıyla final raporu hazırlanacak ve eksiksiz bir şekilde final raporu teslim edilecektir.

2.2. İş Zaman Çizelgesi



Şekil 2.1. Projenin hayata geçirilme aşamaları ve geçen süreleri gösteren Gantt şeması

1. Matlab Öğrenme Aşaması

Ayrıntılı bir şekilde temelden öğrenme gerçekleştirildi. Bunlar ‘Metin derleme komutları’, ‘Değişken Bastırma Komutları’, ‘Diziler’, ‘Diziler üzerinde kullanılan fonksiyonlar’, ‘for kullanımı’, ‘Matlab de görüntü işleme ile ilgili komutlar’ olarak sıralanabilir. Öğrenme aşamasında Rafael C. Gonzalez ve Richard E. Woods ‘un yazarı oldukları ‘Sayısal Görüntü İşleme’ kitabından, internetteki konuyla ilgili yazılar ve makalelerden, Youtube ve Udemy de bulunan matlab ders videolarından faydalanıldı.

2. Enterpolasyon Yöntemleri Araştırma ve Öğrenme Aşaması

Projeyi gerçekleştirmek için kullanılacak olan Bilinear Enterpolasyon (Bilinear Interpolation) ve En Yakın Komşu Enterpolasyonu (Nearest-neighbor interpolation) araştırıldı ve bu yöntemler hakkında bilgi edinildi. Bilgi edinilirken internetten ve udemy kurslarından faydalanıldı. Ara rapora ek olarak En Yakın Komşu Enterpolasyonu öğrenildi, bilgileri final raporuna eklendi ve matlab e algoritması uyarlandı.

3. Enterpolasyon Yöntemlerinin Matlabda Kullanılması

Bu yöntemlerden elde edilen bilgiler doğrultusunda ve kullanılacak formüller çıkarıldı. Formül olarak Bilinear Enterpolasyon yönteminin formülleri ara raporda olduğu gibi final raporunda da aynen yer aldı. Fakat En Yakın Komşu Enterpolasyonunda formül olmadığı için sadece o yönteme ait sözel bilgiler rapora eklendi. Bu formüller matlabda fonksiyon haline getirildi ve projede kullanıldı.

4. Debug ve Hata Denetleme

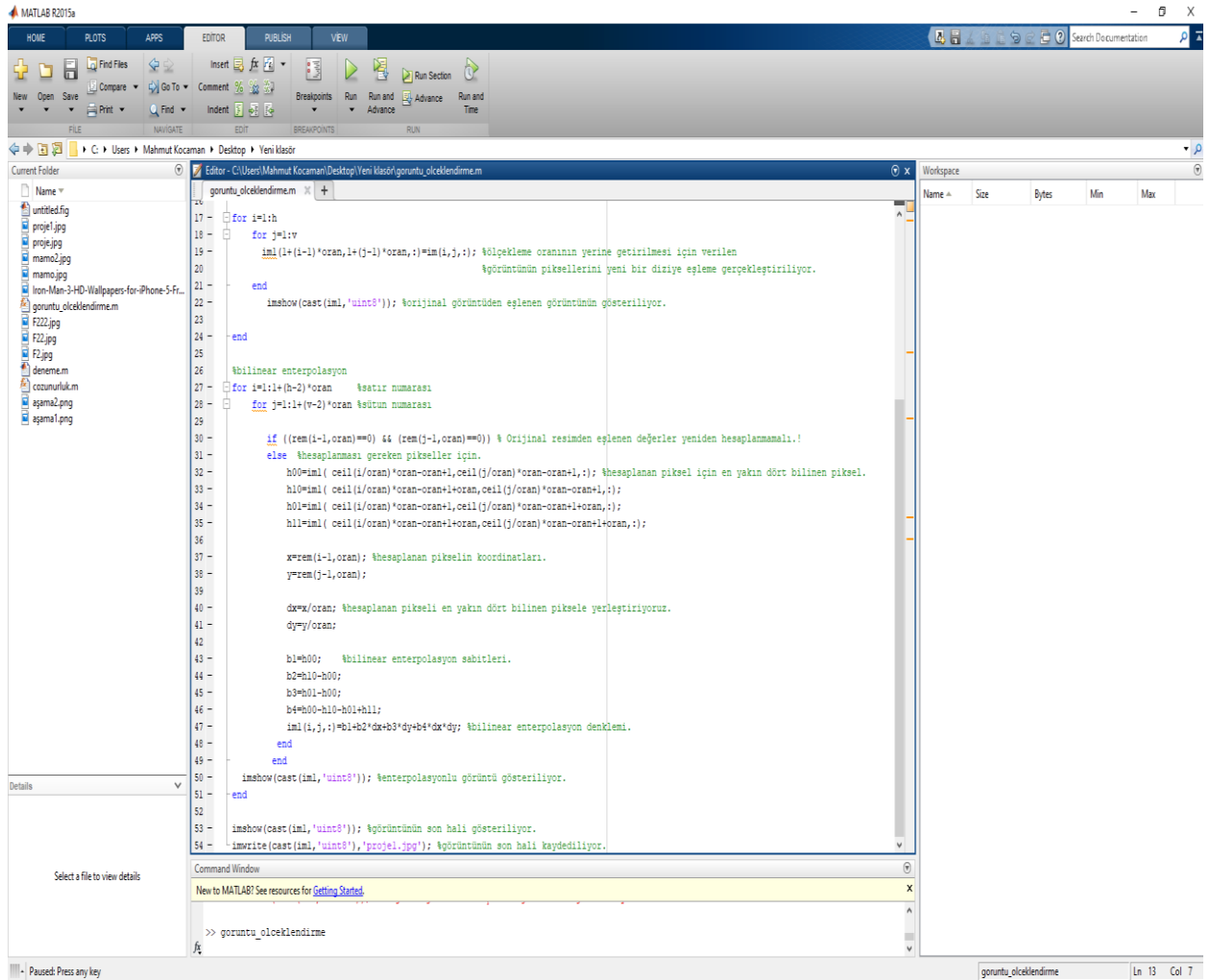
Matlabda kod yazıldıktan sonra kodu gerçekleştirme aşamasında karşılaşılan sorunlar incelendi ve gereken müdahaleler yapılarak sorunlar giderildi. En büyük sorun hata analizini gerçekleştirme sırasında yaşandı. Olması gereken en düşük hata oranını yakalamak için algoritmalar defalarca değiştirildi veya düzeltildi. Sonrasında proje gerçekleştirilerek istenilen çıktı alındı.

5. Rapor Yazma Aşaması

Bu aşamaların hepsini gerçekleştirdikten sonra tüm ayrıntılarıyla ara rapor hazırlanmıştı. Ara rapora ek olarak En Yakın Komşu Enterpolasyonu yönteminin bilgileri, algoritması ve kodu final raporuna eklendi. Hata analizi çıktıları her iki yöntem için de alındı final raporuna eklendi. Proje aşamaları anlatıldı, projede kullanılan formüller yazıldı, projede kullanılan yöntemler anlatıldı, çıktıları rapora eklendi ve projede gelinek nihai sonuca kadar bilgi içeren final raporu hazırlandı.

2.3. Deneysel Çalışmalar (Bilinear Enterpolasyon Yöntemi İçin)

Kod, matlabda yazıldı ve gerçekleştirildi. Burada gerçekleştirilen aşama ve gelinen son nokta, elimizde bulunan görüntünün çözünürlüğünü, Bilinear Enterpolasyon yöntemiyle belirlenen bir oranda arttırılmasıdır. Ara rapora ek olarak Bilinear Enterpolasyon un kodu fonksiyon olarak yazılmıştı, kod düzeltilerek boyutlandırma oranını kodun içinde gömülü olacak şekilde değiştirildi. Ayrıca bu yöntemin hata analiz sonucu çıktıları rapora eklendi.

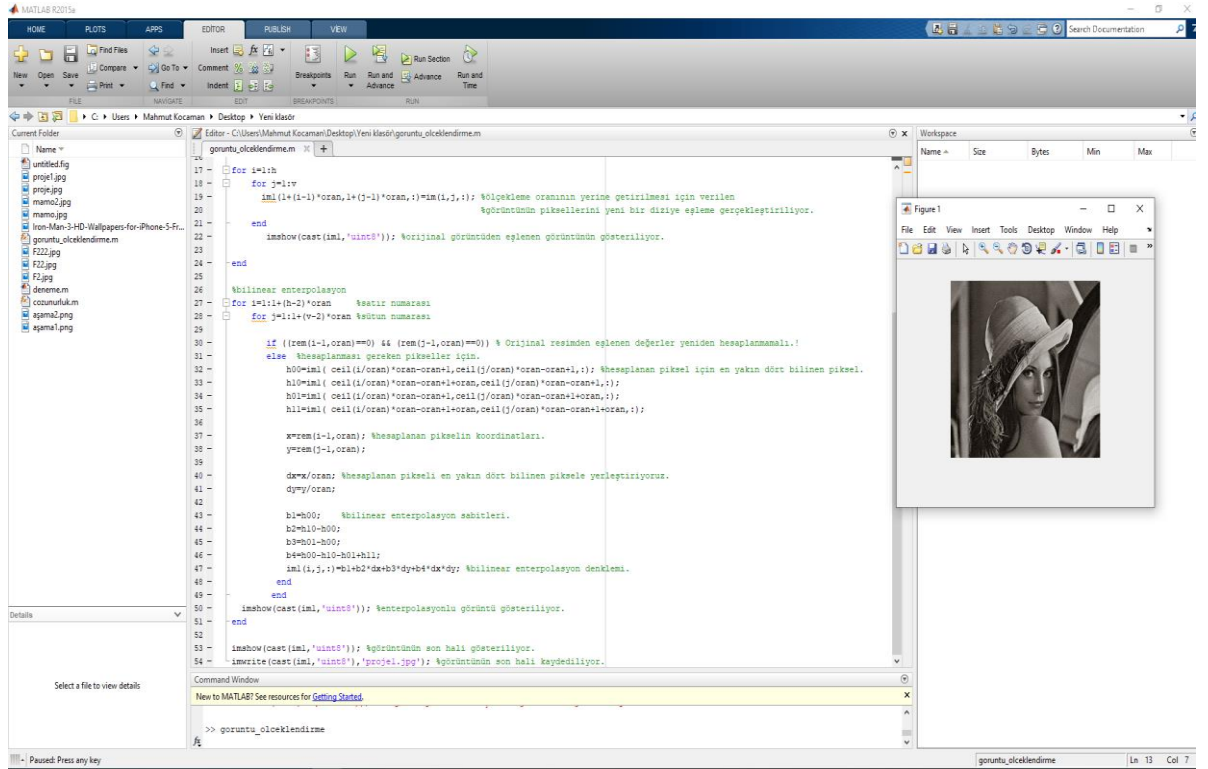


The screenshot shows the MATLAB R2015a environment. The Editor window displays a script named 'goruntu_olceklendirme.m'. The script performs bilinear interpolation on an image. It starts by reading an image 'projel.jpg' and displaying it. Then, it defines the interpolation function with parameters for the original image size (h0, w0) and the target size (h1, w1). The script calculates the scaling factors and uses a nested loop to iterate over the target image pixels. For each pixel, it finds the four nearest pixels in the original image and calculates the weighted average based on their relative positions. The result is displayed as 'goruntu_olceklendirme' and saved as 'projel.jpg'.

```
17 for i=1:h
18     for j=1:w
19         iml(i+(i-1)*oran,i+(j-1)*oran,:)=im(i,j,:); %ölçekleme oranının yerine getirilmesi için verilen
20         %görüntünün piksellerini yeni bir diziye ekleme gerçekleştiriliyor.
21     end
22     imshow(cast(iml,'uint8')); %orijinal görüntüden ekleme görüntünün gösteriliyor.
23 end
24
25 %bilinear enterpolasyon
26 for i=1:(h-2)*oran %satır numarası
27     for j=1:(w-2)*oran %sütun numarası
28
29         if ((rem(i-1,oran)==0) && (rem(j-1,oran)==0)) % Orijinal resimden ekleme değerler yeniden hesaplanmalıdır.
30             %hesaplanması gereken pikseller için.
31             h0=iml(ceil(i/oran)*oran-oran+1,ceil(j/oran)*oran-oran+1,:); %hesaplanan piksel için en yakın dört bilinen piksel.
32             h1=iml(ceil(i/oran)*oran-oran+1,oran,ceil(j/oran)*oran-oran+1,:);
33             h0=iml(ceil(i/oran)*oran-oran+1,ceil(j/oran)*oran-oran+1,oran,:);
34             h1=iml(ceil(i/oran)*oran-oran+1,oran,ceil(j/oran)*oran-oran+1,oran,:);
35             h1=iml(ceil(i/oran)*oran-oran+1,oran,ceil(j/oran)*oran-oran+1,oran,:);
36
37             x=rem(i-1,oran); %hesaplanan pikselin koordinatları.
38             y=rem(j-1,oran);
39
40             dx=x/oran; %hesaplanan pikseli en yakın dört bilinen piksele yerleştiriyoruz.
41             dy=y/oran;
42
43             b1=h0; %bilinear enterpolasyon sabitleri.
44             b2=h1-h0;
45             b3=h0-h0;
46             b4=h0-h1+h1;
47             iml(i,j,:)=b1+b2*dx+b3*dy+b4*dx*dy; %bilinear enterpolasyon denklemi.
48         end
49     end
50     imshow(cast(iml,'uint8')); %enterpolasyonlu görüntü gösteriliyor.
51 end
52
53 imshow(cast(iml,'uint8')); %görüntünün son hali gösteriliyor.
54 imwrite(cast(iml,'uint8'),'projel.jpg'); %görüntünün son hali kaydediliyor.
```

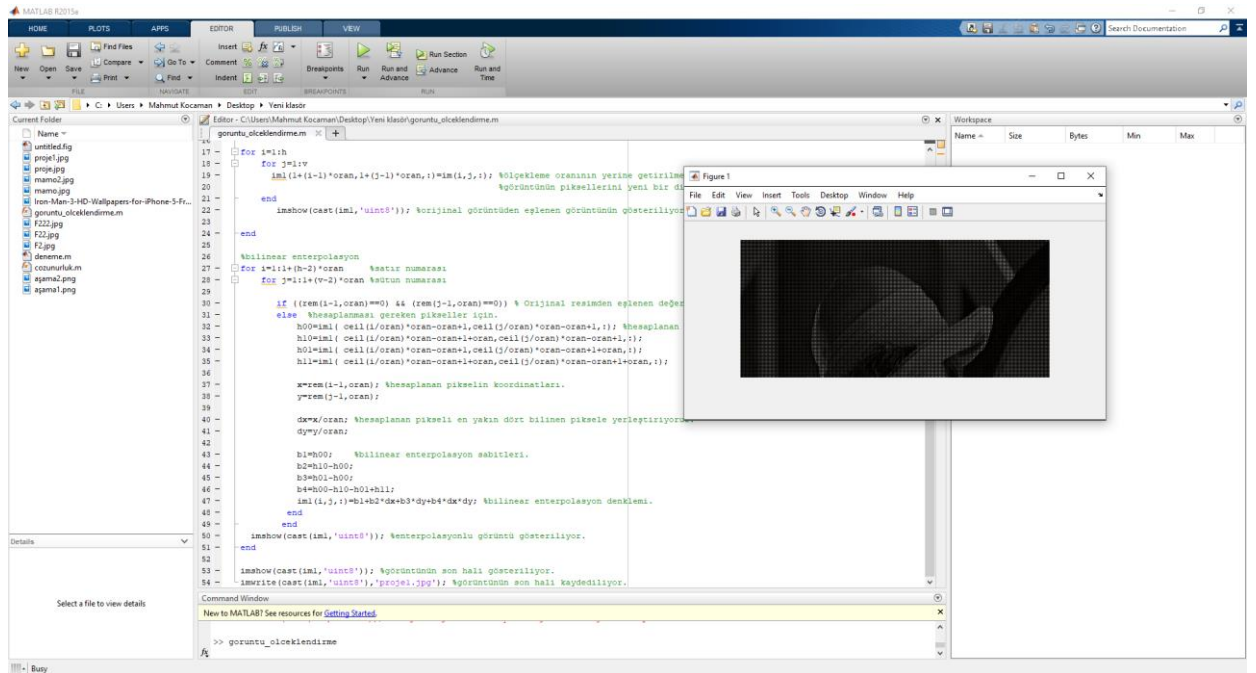
Şekil 2.3.1 Kodun Matlab programında yazılmış hali.

Burada kod matlab programına yazıldı ve çalıştırılmayı bekliyor. Kod parçalarının açıklamaları bölüm sonunda paylaşılan kodda yazmaktadır.



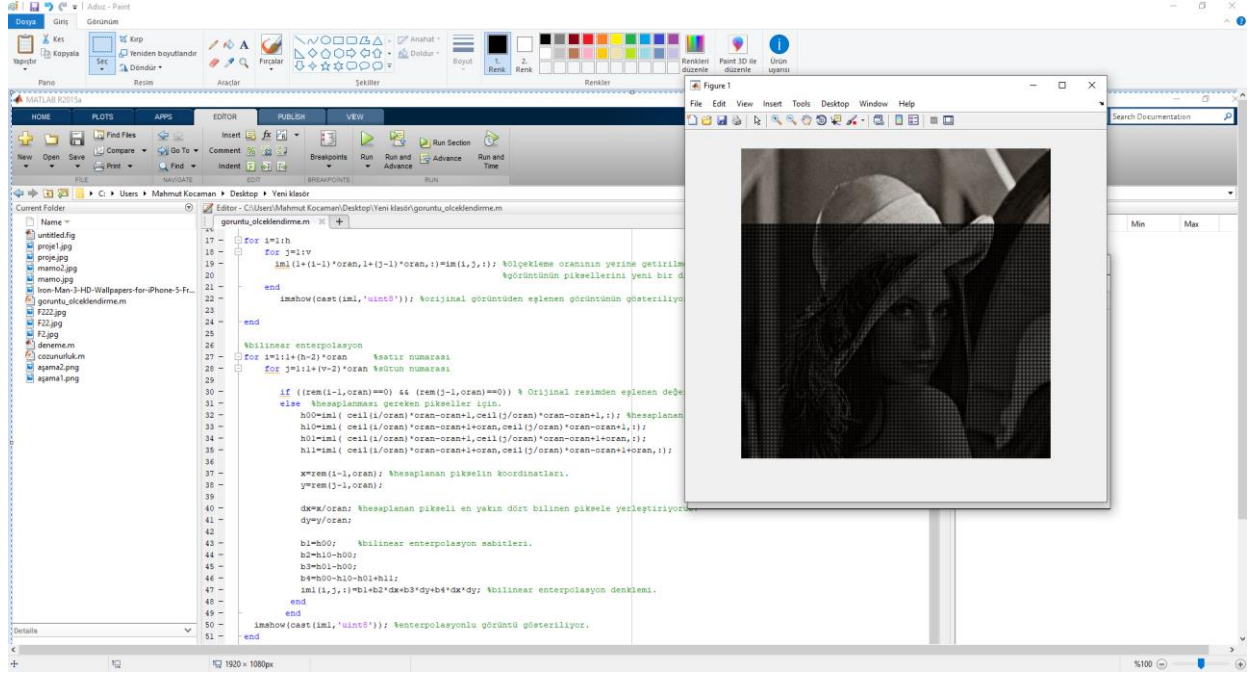
Şekil 2.3.2. Kodun ilk çalıştırılması ile birlikte orijinal görüntünün gösterilmesi.

Şekil 2.3.2. de görüldüğü üzere kodu ilk çalıştırdığımızda görüntünün orijinal 240x240 çözünürlüklü hali gösteriliyor ve kod duraklatılıyor.



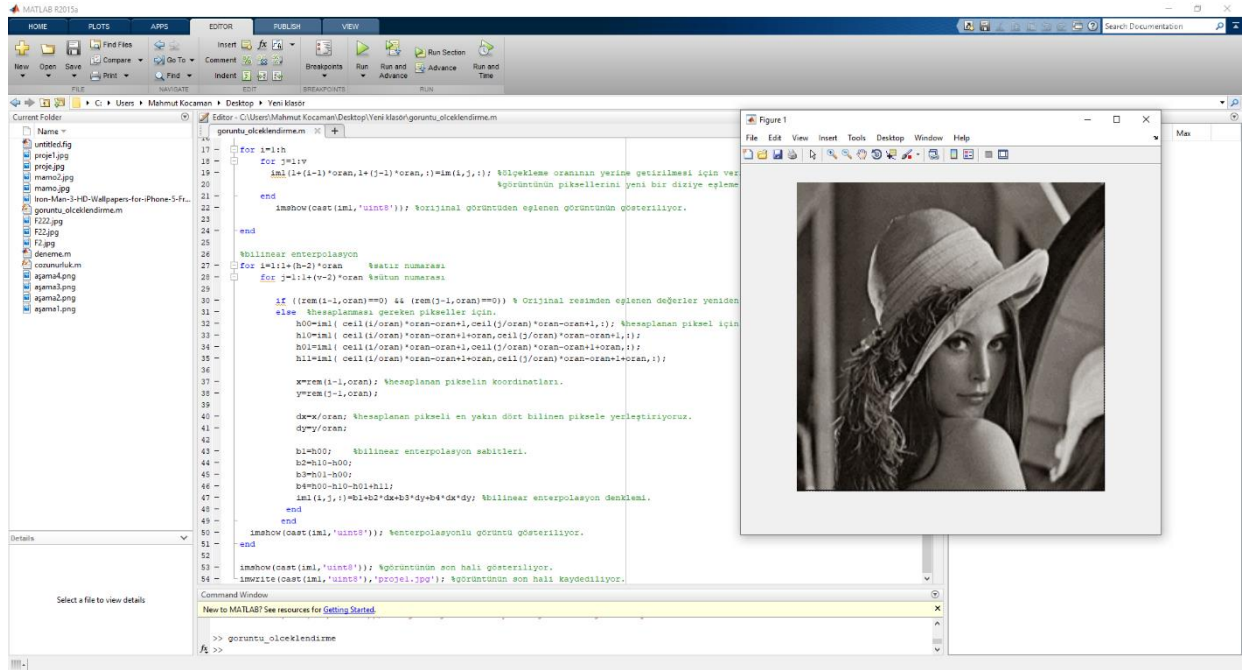
Şekil 2.3.3. Orijinal görüntüden eşlenen görüntü gösterilmeye başlıyor.

Şekil 2.3.3. de görüldüğü üzere orijinal görüntüden eşlenen görüntü gösterilmeye başlıyor. Burada orijinal görüntünün pikselleri açılıyor ve aralarına eşlenecek olan pikseller yerleştiriliyor. Bu yerleşecek olan piksellerin hesaplanmasına daha başlanmadığı için pikseller siyah olarak görülüyor.



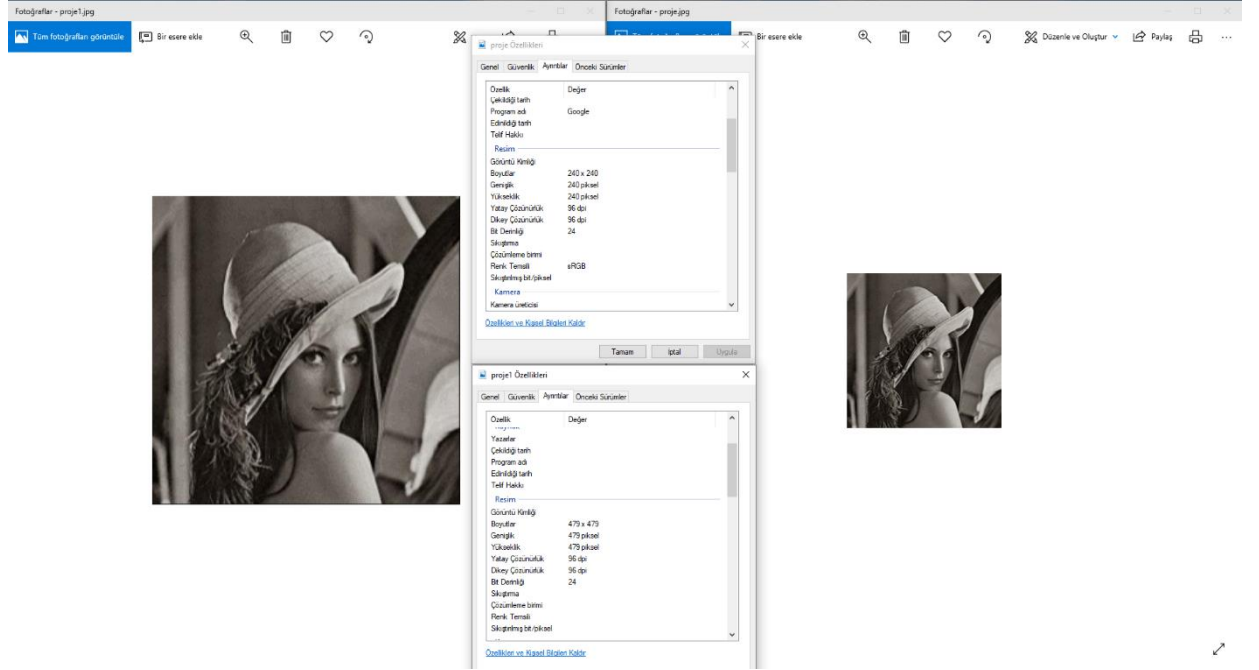
Şekil 2.3.4. Orijinal olandan eşlenen görüntüden, enterpolasyonlu görüntü elde ediliyor.

Şekil 2.3.4. de orijinal görüntü piksellerinin arasında bulunan hesaplanmamış pikseller hesaplanmaya başlıyor ve piksel değerleri atanıyor. Şekilde görüldüğü üzere hesaplanan pikseller, orijinal görüntünün piksellerine çok yakın değerler aldığı için siyahlık ortadan kalkmaya başlıyor.

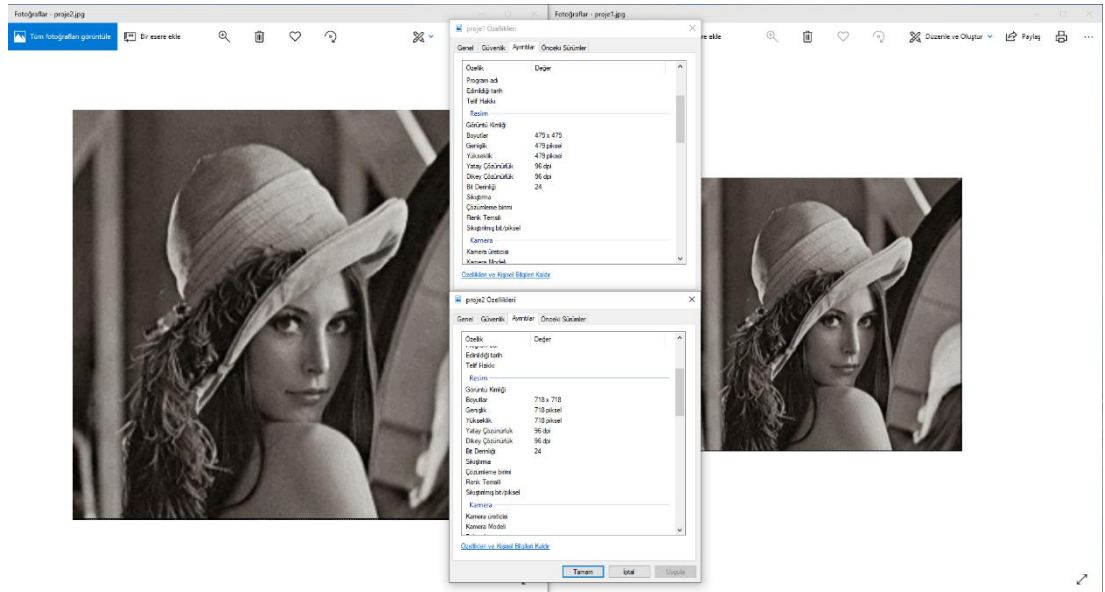


Şekil 2.3.5. Enterpolasyonlu görüntü tek başına gösteriliyor.

Program sonunda enterpolasyonlanmış görüntü elde ediliyor. Çözünürlüğü 479x479 ve orijinal görüntünün çözünürlüğünün 2 katına çıkmış durumda. Eşlenmiş pikseller hesaplanmış ve komşu 4 pikseline göre yakın değerler aldığı için resmin genel orijinalliği bozulmuyor. Fakat ortalama değerler alındığı için görüntü kalitesinde az da olsa bozulma oluyor.



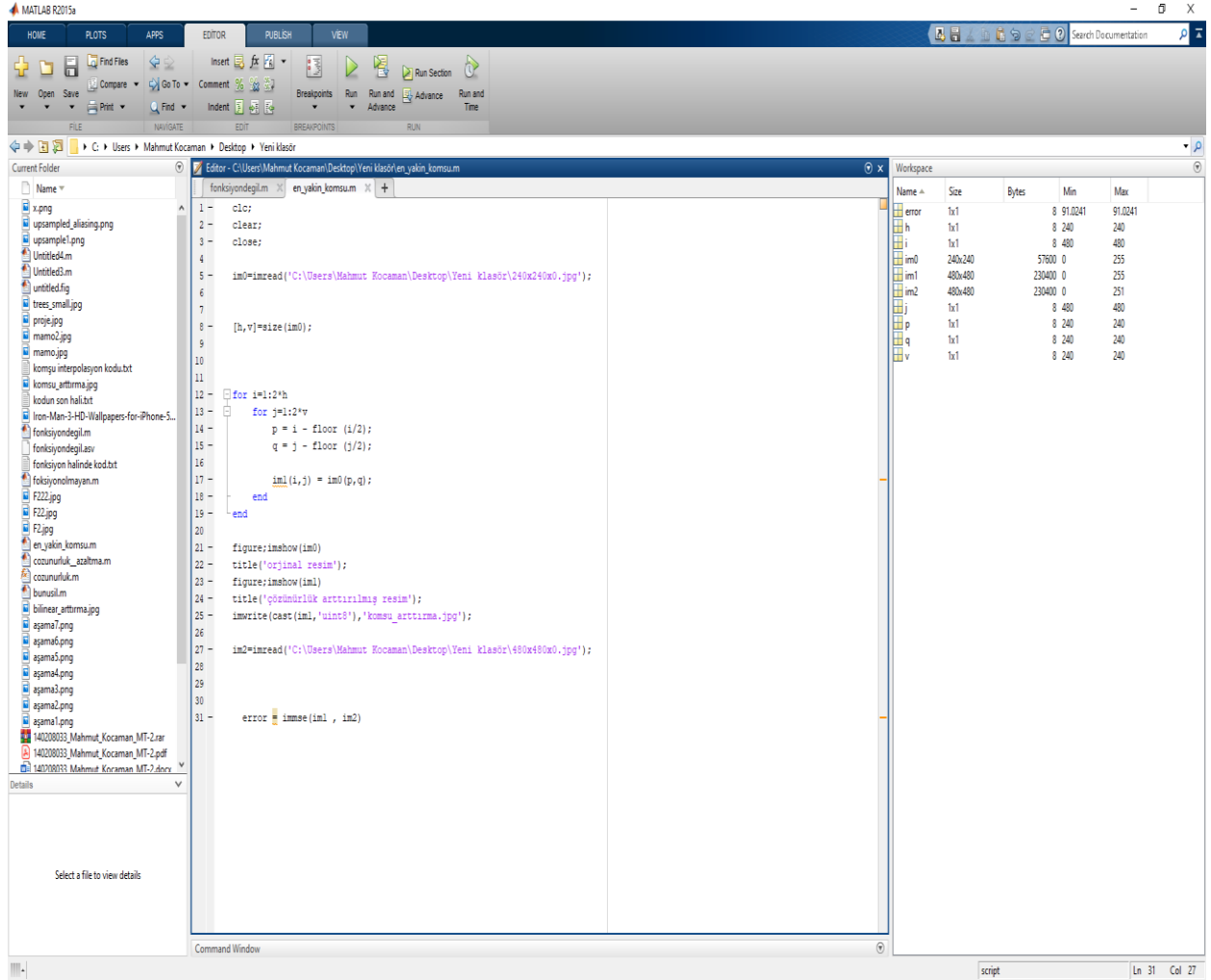
Şekil 2.3.6. Orijinal görüntü ve iki oranında ölçeklenmiş çözünürlüklü görüntü arasındaki ölçek ve kalite farkı.



Şekil 2.3.7. İki oranında ölçeklenmiş ve üç oranında ölçeklenmiş çözünürlüklü görüntü arasındaki ölçek ve kalite farkı.

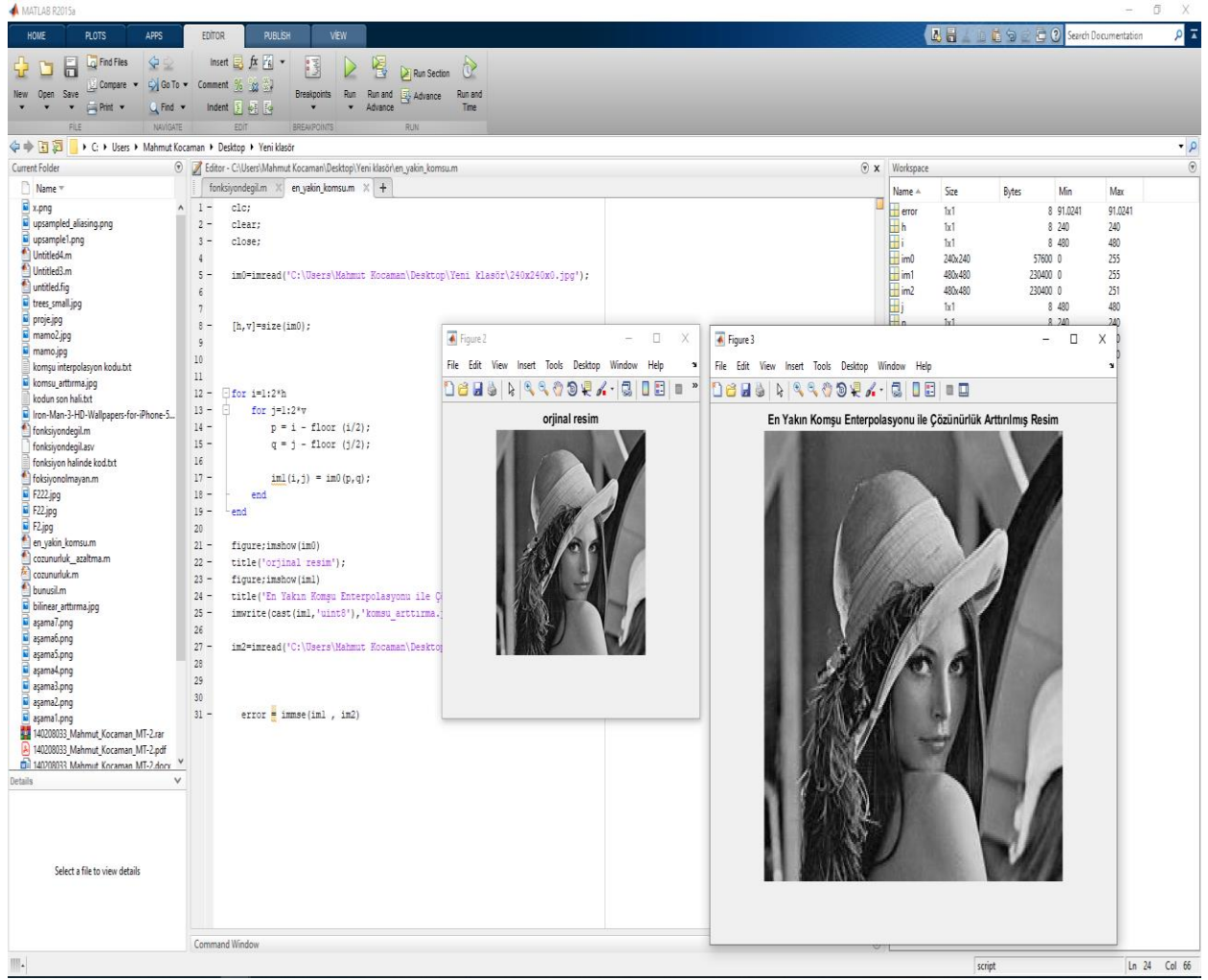
2.4. Deneysel Çalışmalar (En Yakın Komşu Enterpolasyon Yöntemi İçin)

Kod, matlabda yazıldı ve gerçekleştirildi. Burada gerçekleştirilen aşama ve geline son nokta, elimizde bulunan görüntünün çözünürlüğünü, En Yakın Komşu Enterpolasyon yöntemiyle belirlenen bir oranda arttırılmasıdır.



Şekil 2.4.1. Kodun Matlab programında yazılmış hali.

Burada kod matlab programına yazıldı ve çalıştırılmayı bekliyor. Kod parçalarının açıklamaları bölüm sonunda paylaşılan kodda yazmaktadır.



Şekil 2.4.2. Kodun çalıştırılmasından sonra enterpolasyonlu görüntü ve orijinal görüntü birlikte gösteriliyor.

En Yakın Komşu Enterolasyonu bilgilerini içeren bölümde de anlatıldığı üzere bu yöntem basit algoritma ve boyutlandırma işleminin çok az zamanda tamamlanması gibi avantajlar içeriyordu. Şekil 2.4.2 de görüldüğü üzere kod gayet basit ve işlem tamamlanma hızı gayet başarılı. Ayrıca ara işleme gerek duymadan, diğer yönteme göre daha hızlı çıktıyı veriyor. Çözünürlüğü 480x480 ve orijinal görüntünün çözünürlüğünün 2 katına çıkmış durumda. Eşlenmiş pikseller, en yakın belirlenmiş piksellerine göre yerleştirilmiş olduğu için resmin genel orijinallliği bozulmuyor. Fakat ard arda tekrar eden fazla pikseller olduğu için diğer yönteme göre görüntü kalitesinde daha çok bozulma oluyor.

2.5. Proje İçin Yazılan Matlab Kodu (Bilinear Enterpolasyon Yöntemi İçin)

```
clc;
clear;
close;

im0=imread('C:\Users\Mahmut Kocaman\Desktop\Yeni klasör\proje.jpg');

im=cast(im0,'int16'); %im0 bir uint8 türü olduğundan, im0 negatif
değerleri ve 255'ten daha büyük olmayan tamsayıları desteklemez o
yüzden int16 ya çeviriyoruz.

imshow(cast(im,'uint8')); % imshow nedeniyle int16 desteklemiyor. Bu
kısım orjinal görüntüyü gösteriyor o yüzden orjinal görüntüyü
gösterirken uint8 şeklinde gösteriyoruz.

pause; %görüntünün orjinal hali gösterildikten sonra kod duruyor
sonrasında tekrardan Command Window kısmında enter a bastığımızda
kod devam edecektir.

[h,v,d]=size(im); %yükseklik , genişlik ve derinlik

oran = 2; %bu kısımda görüntümüzün kaç kat arttırarak
ölçeklendireceğimizi belirtiyoruz.

for i=1:h

    for j=1:v

        im1(1+(i-1)*oran,1+(j-1)*oran,:)=im(i,j,:); %ölçekleme
oranının yerine getirilmesi için verilen görüntünün piksellerini
yeni bir diziye eşleme gerçekleştiriliyor.

    end

        imshow(cast(im1,'uint8')); %orijinal görüntüden eşlenen
görüntü gösteriliyor.

end

%bilinear enterpolasyon

for i=1:1+(h-2)*oran %sıra numarası

    for j=1:1+(v-2)*oran %sütun numarası

        if ((rem(i-1,oran)==0) && (rem(j-1,oran)==0)) % Orijinal
resimden eşlenen değerler yeniden hesaplanmamalı.! rem(x,y) şeklinde
kullanır. Burada x bölünen , y ise bölen sayıdır. Program x'in y'ye
bölünmesinde kalan değerini verir.
```

```

else %hesaplanması gereken pikseller için.

    h00=im1( ceil(i/oran)*oran-oran+1,ceil(j/oran)*oran-
oran+1,:); %hesaplanan piksel için en yakın dört bilinen piksel. ceil
komutu + yöne doğru en yakın tamsayıya yuvarlatma yapar. Örneğin
ceil(5.3) komutunun cevabı 6'dır.

    h10=im1( ceil(i/oran)*oran-oran+1+oran,ceil(j/oran)*oran-
oran+1,:);

    h01=im1( ceil(i/oran)*oran-oran+1,ceil(j/oran)*oran-
oran+1+oran,:);

    h11=im1( ceil(i/oran)*oran-oran+1+oran,ceil(j/oran)*oran-
oran+1+oran,:);

    x=rem(i-1,oran); %hesaplanan pikselin koordinatları.
    y=rem(j-1,oran);

    dx=x/oran; %hesaplanan pikseli en yakın dört bilinen
piksele yerleştiriyoruz.
    dy=y/oran;

    b1=h00; %bilinear enterpolasyon sabitleri.
    b2=h10-h00;
    b3=h01-h00;
    b4=h00-h10-h01+h11;

    im1(i,j,:)=b1+b2*dx+b3*dy+b4*dx*dy; %bilinear
enterpolasyon denklemi.
end
end
imshow(cast(im1,'uint8')); %enterpolasyonlu görüntü gösteriliyor.
end

figure;imshow(cast(im1,'uint8')); %görüntünün son hali gösteriliyor.
title('Çözünürlüğü Arttırılmış Resim');
figure;imshow(im0); %görüntünün orijinali gösteriliyor
title('Orjinal Resim');

imwrite(cast(im1,'uint8'),'bilinear_arttırma.jpg'); %görüntünün son
hali kaydediliyor.

im2=imread('C:\Users\Mahmut Kocaman\Desktop\Yeni
klasör\bilinear_arttırma.jpg');
im3=imread('C:\Users\Mahmut Kocaman\Desktop\Yeni
klasör\1023x1023.jpg');

error = immse(im2 , im3) %enterpolasyonlu görüntü ve küçültülmemiş
orjinal görüntünün benzerlik oranı belirleniyor

```

2.6. Proje İçin Yazılan Matlab Kodu (En Yakın Komşu Enterpolasyon Yöntemi İçin)

```
clc;
clear;
close;

im0=imread('C:\Users\Mahmut Kocaman\Desktop\Yeni
klasör\240x240x0.jpg');

[h,v]=size(im0); %orjinal resmin boyutu ölçülüyor. Burada size
komutu matrisin boyutlarını değişkenlere atamaya yarıyor.

oran = 2; %görüntünün hangi oranda boyutlandırılacağı belirleniyor

for i=1:oran*h %ölçekleme oranının yerine getirilmesi için verilen
görüntünün piksellerini yeni bir diziye eşleme gerçekleştiriliyor.
    for j=1:oran*v
        p = i - floor (i/oran); %seçilecek en yakın komşu piksel
        %belirleniyor. Floor komutu - yöne doğru en yakın tamsayıya
        %yuvarlatma yapar. Örneğin floor(2.3) komutunun cevabı 2'dir.
        q = j - floor (j/oran);

        im1(i,j) = im0(p,q); %belirlenen pikseller boş piksellere
        %aynen yerleştiriliyor
    end
end

figure;imshow(im0)
title('orjinal resim');
figure;imshow(im1)
title('En Yakın Komşu Enterpolasyonu ile Çözünürlük Arttırılmış
Resim');
imwrite(cast(im1,'uint8'),'komsu_arttırma.jpg');

im2=imread('C:\Users\Mahmut Kocaman\Desktop\Yeni
klasör\480x480x0.jpg');

error = immse(im1 , im2) %enterpolasyonlu görüntü ve
küçültülmemiş orjinal görüntünün benzerlik oranı belirleniyor. err =
immse(X,Y)diziler arasındaki ortalama-kare hatasını hesaplar X ve Y
herhangi bir boyutun dizileri olabilir, ancak aynı boyutta ve
sınıfta olmalıdırlar.
```

3. SONUÇLAR

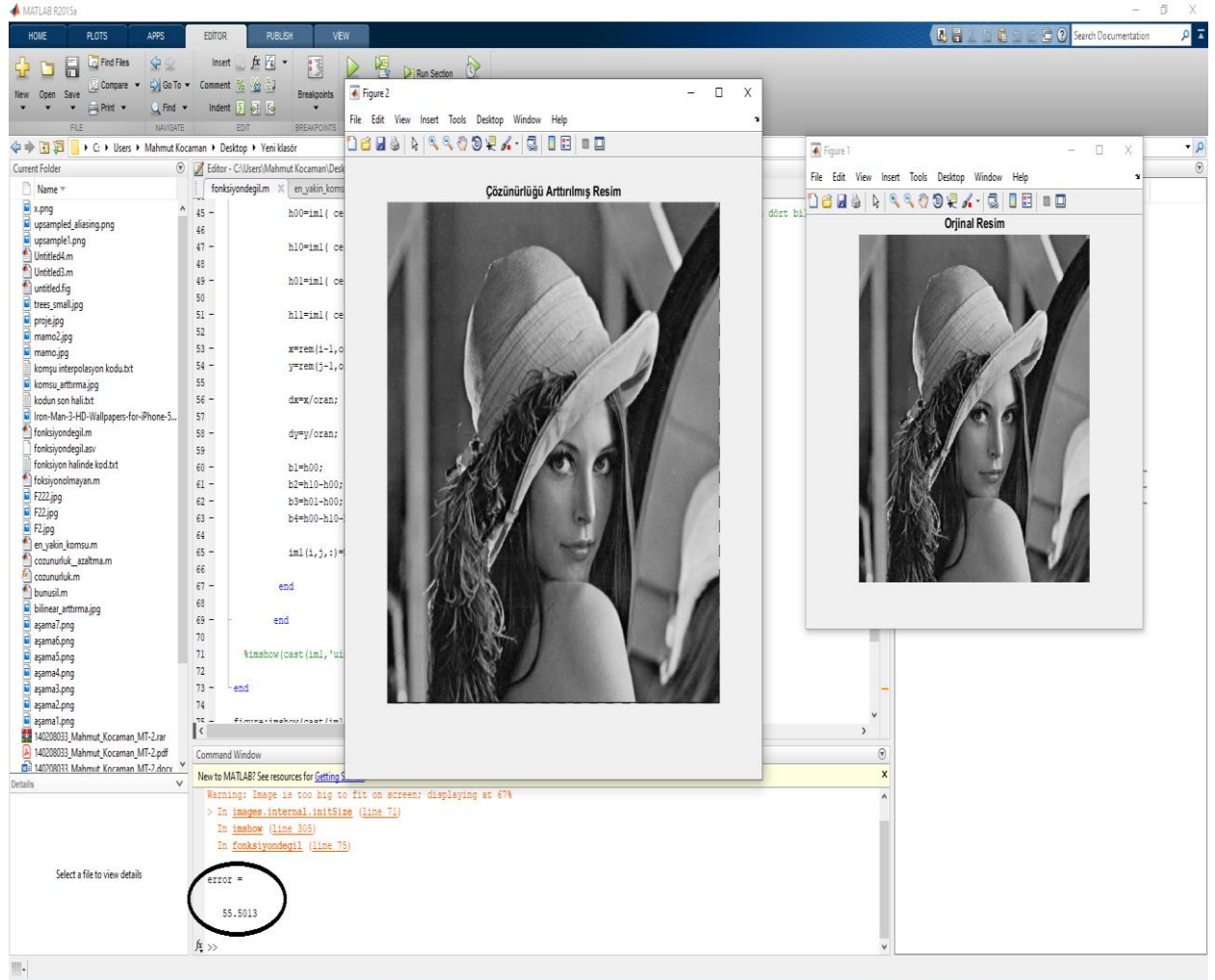
Final raporunda ara rapora ek olarak bir diğer enterpolasyon yöntemi olan En Yakın Komşu Enterpolasyonu yöntemi ile ilgili bilgiler, algoritması, avantajları ve dezavantajlarına yer verildi. Ayrıca kullanılan kod paylaşıldı. Bilinear Enterpolasyonu kodu güncellendi. Ara raporda Bilinear Enterpolasyon Yöntemi kodu fonksiyon olarak paylaşılmıştı, final raporuna boyutlandırma oranının kodun içinde gömülü olan versiyonu ile değiştirildi. Eski koda ek olarak görüntü çıktıları matlabda alınan kod güncellemesi eklendi. Her iki yöntem için hata oranını gösteren kod eklendi.

Bu iki yöntemi kullanarak alınan çıktılar ve verilere dayanarak elde edilen sonuçlar şunlardır:

Öncelikle her iki yönteminde kendine göre avantajları ve dezavantajları vardır. Bilinear Enterpolasyon Yöntemi daha karmaşık bir algoritmaya sahiptir ve kullanılan kod daha uzun ve anlaşılması zordur. Ayrıca boyutlandırma işlemini diğer yönteme göre daha uzun bir sürede yapar. Ancak elde edilen çıktı, kalite olarak En Yakın Komşu Enterpolasyonuna göre daha iyidir. Bunu Bilinear Enterpolasyon Yönteminin hata oranına bakarak rahatça söyleyebiliriz. En Yakın Komşu Enterpolasyonu ise daha basit bir algoritmaya sahip, kısa ve anlaşılması kolay bir koda sahiptir. Bu yüzden boyutlandırma işlemini daha kısa sürede tamamlar. Ancak ayrıntılı değil de yüzeysel bir enterpolasyon yaptığı için kalite olarak daha geride kalmaktadır. Bunu da En Yakın Komşu Enterpolasyonu Yöntemi için hata oranına bakarak rahatça söyleyebiliriz.

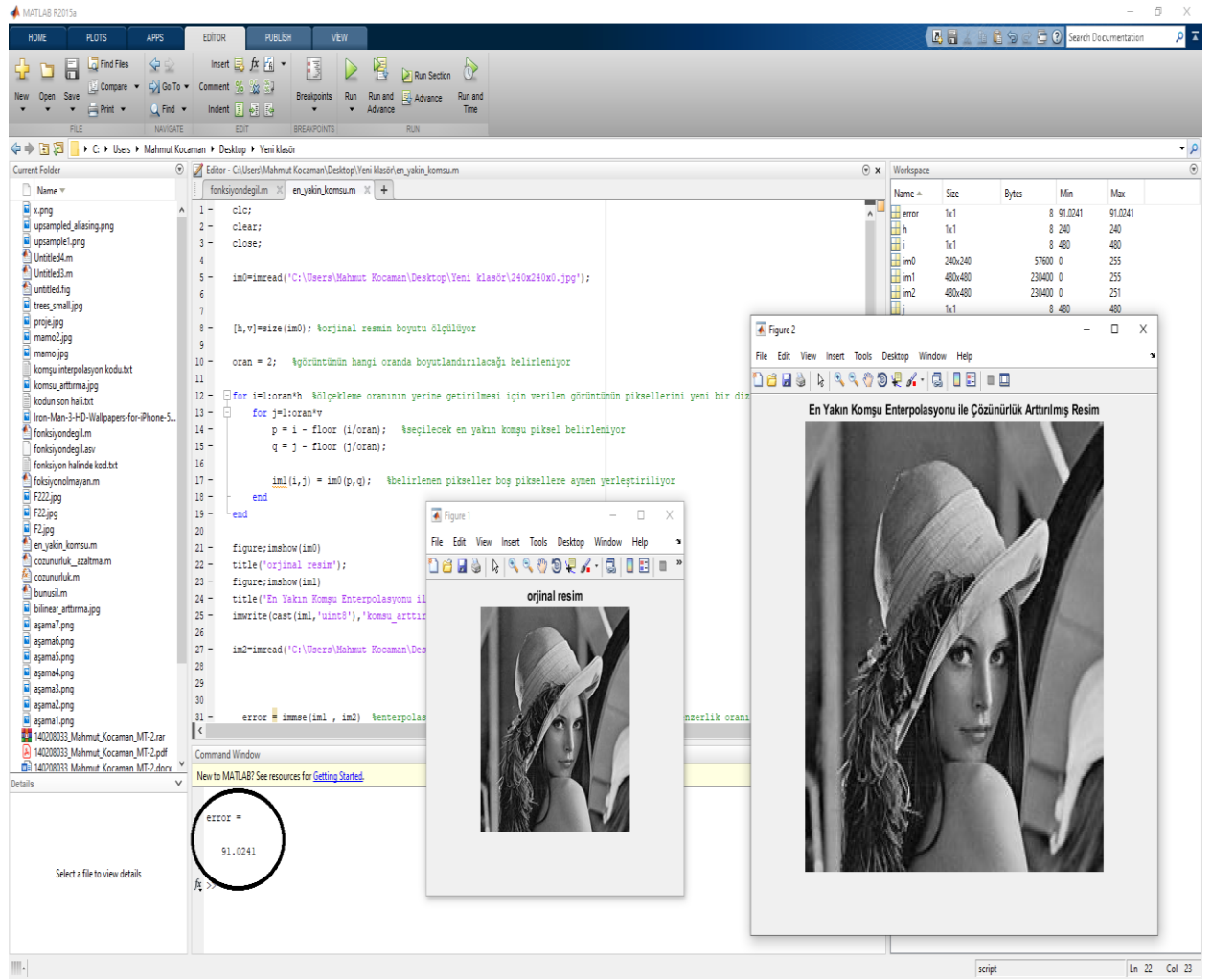
Projeyi ileriye taşıyabilmek adına yapılabilecek şeylerin başında daha dinamik ve daha kaliteli sonuçlar verecek yöntemlere başvurmak gelmektedir. Böylelikle daha iyi sonuçlar elde edilerek kritik derecede önemli ve düşük çözünürlüklü görüntüleri daha sağlıklı ve hata oranının sıfıra yakın derecede olacağı çıktılar alınabilecektir. Örneğin güvenlik kameralarında belirlenen nesneyi en iyi şekilde tanımlamak, uzak mesafelerdeki görüntüleri en iyi şekilde görebilmek için daha iyi yöntemler kullanılmalıdır.

3.1. Hata – Benzerlik Oranı (Bilinear Enterpolasyon Yöntemi İçin)



Şekil 3.1.1. Bilinear Enterpolasyon Yöntemi kullanılarak boyutlandırılan görüntünün hata oran

3.2. Hata – Benzerlik Oranı (En Yakın Komşu Enterpolasyonu Yöntemi İçin)



Şekil 4.2.1. En Yakın Komşu Enterpolasyonu Yöntemi kullanılarak boyutlandırılan görüntünün hata oran

KAYNAKLAR

- [1] <http://blog.udentify.co/04/2017/goruntu-isleme-nedir/>
- [2] https://en.wikipedia.org/wiki/Image_scaling
- [3] https://en.wikipedia.org/wiki/Bilinear_interpolation
- [4] Digital Image Processing Rafael C. Gonzalez, Richard E. Woods
- [5] https://www.giassa.net/?page_id=207
- [6] <https://www.mathworks.com/help/vision/ug/interpolation-methods.html>
- [7] <https://www.mathworks.com/help/images/ref/immse.html>
- [8] <https://www.mekatronikmuhendisligi.com/matlab-temel-islemler.html>