2022

MATLAB ile Görüntü İşleme Kullanarak Filtre Uygulaması

SİSTEM PROGRAMLAMA DERSİ FİNAL ÖDEVİ

BURAK KAAN ŞAHİN

201513152099

İçindekiler

MATLAB'DA GÖRÜNTÜ İŞLEME KULLANARAK FİLTRE UYGULAMASI TASARIMI

FILTER APPLICATION DESIGN USING IMAGE PROCESSING IN MATLAB

Burak Kaan ŞAHİN
201513152099

ABSTRACT

In this study, a program and its interface were prepared by using the MATLAB program, which reads on the picture and applies filters according to the values we read. First of all, the interface template for the program was designed via APP Designer, which is a MATLAB plugin. Function codes were written for the buttons and tools added later. Callback functions of the function codes are made to the buttons. The program has been tested both as ".exe" and ".m" and ".mlapp" files over MATLAB.
Keywords: MATLAB, APPDesigner, Photo Editor, Image Editing, Applying Effects, Interface.

ÖZET

Bu çalışmada MATLAB programı kullanılarak resim üzerinde okuma yapıp okuduğumuz değerlere göre filtre uygulayan bir program ve arayüzü hazırlanmıştır. Program için öncelikle arayüz şablonu MATLAB eklentisi olan APPdesigner üzerinden tasarlanmıştır. Daha sonrasında eklenen buton ve araçlara fonksiyon kodları yazılmıştır. Fonksiyon kodlarının butonlara geri çağırma fonksiyonları yapılmıştır. Program hem ".exe" hali hem de MATLAB üzerinden ".m" ve ".mlapp" dosyası test edilmiştir.
Anahtar Kelimeler: MATLAB, APPDesigner, Fotoğraf Düzenleyici, Resim Düzenleme, Efekt Uygulama, Arayüz

1. GİRİŞ

Görüntü işleme, bilgisayarların görüntüler ve videolardaki nesneleri ve kişileri tanımlamasını ve anlamasını sağlamaya odaklanan bir bilgisayar bilimi alanıdır. Diğer yapay zeka türleri gibi görüntü işleme de insan yeteneklerini kopyalayan görevleri gerçekleştirmeyi ve otomatikleştirmeyi amaçlar. Bu durumda, görüntü işleme hem insanların görme biçimini hem de gördüklerini anlamlandırma biçimini kopyalamaya çalışır. Görüntü işleme, CAD programları, çizim ve tasarım programları gibi bir çok uygulama da kas gücünden kurtulup daha az hata payıyla veriyi görselleştirip görseli de veri olarak okumamıza olanak sağlıyor.

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, video karesi veya fotoğraf gibi girdinin görüntü olduğu ve

çıktının da görüntü veya o görüntüyle ilişkili özellikler olabileceği bir tür sinyal dağıtımıdır.

Görüntü işlemenin amacı 5 gruba ayrılmıştır. Onlar:

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.

Bu bildiride, projede kullanılan uygulamaların açıklanması, uygulamayı oluşturan kodun hazırlanması, uygulamanın kurulumunun nasıl yapılacağı ve uygulamanın bilgisayar ortamında test edilmesi anlatılmaktadır. İkinci bölümde MATLAB ile görüntü işleme başlığı altında APP designer uygulamasının ne işe yaradığını ve şablonun hazırlanmasına ve uygulama koduna yer verilmektedir.Üçüncü bölümde ihtiyaç dahilinde uygulamanın kurulumu gerekebileceğinden dolayı kurulumu ve nasıl çalıştığı anlatılmaktadır. Dördüncü bölümde uygulamanın sonuçları verilmektedir.

2. MATLAB İLE GÖRÜNTÜ İŞLEME

MATLAB (matrix laboratory), çok paradigmalı sayısal hesaplama yazılımı ve dördüncü nesil programlama dilidir. Sahipli bir programlama dili olan MATLAB, MathWorks tarafından geliştirilmektedir. MATLAB kullanıcıya, matris işleme, fonksiyon ve veri çizme, algoritma uygulama, kullanıcı arayüzü oluşturma, C, C++, Java, ve Fortran gibi diğer dillerde yazılmış programlarla arabağlama imkânı tanır.

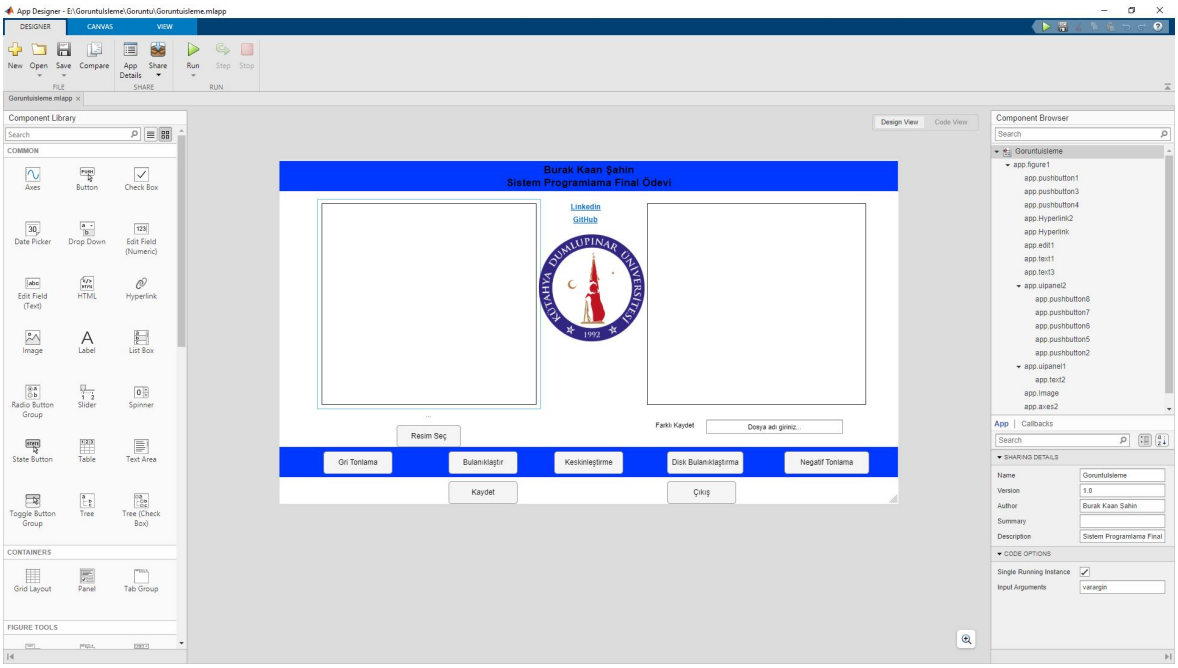
MATLAB, öncelikli olarak sayısal işleme yönelik üretilmiş olmasına rağmen, isteğe bağlı olarak sembolik hesaplama yapabilen MuPAD sembolik motorunu kullanır. Ek paket, dinamik ve gömülü sistemler için Simulink'i, grafiksel çoklu alan simülasyonunu ve model tabanlı tasarımı ekler.

2.1. App Designer

App Designer, bir uygulama düzeni tasarlamak ve davranışını programlamak için etkileşimli bir geliştirme ortamıdır. MATLAB Editor'ün tam entegre bir sürümünü ve çok sayıda etkileşimli arayüz bileşeni sağlar. Ayrıca, kullanıcı arayüzünüzü düzenlemek için bir ızgara düzeni yöneticisi ve uygulamanızın ekran boyutundaki değişiklikleri algılamasını ve bunlara yanıt vermesini sağlamak için otomatik yeniden akış seçenekleri sunar. Uygulamaları doğrudan App Designer araç çubuğundan paketleyerek veya bağımsız bir masaüstü veya web uygulaması oluşturarak (MATLAB Compiler gerektirir) dağıtmanıza olanak tanır.

App Designer MATLAB'da uygulama oluşturmak için önerilen ortamdır. App Designer, profesyonel bir yazılım geliştiricisi olmanıza gerek kalmadan MATLAB'da profesyonel

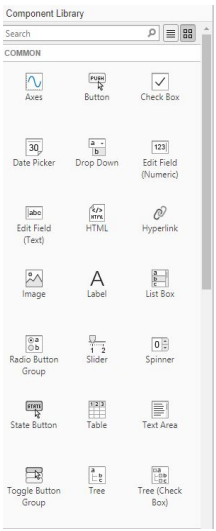
uygulamalar oluşturmanıza olanak tanır. Grafik kullanıcı arayüzü (Graphical user interface- GUI) tasarımını düzenlemek için görsel bileşenleri sürükleyip bırakın ve davranışını hızlı bir şekilde programlamak için entegre düzenleyiciyi kullanabilirsiniz.



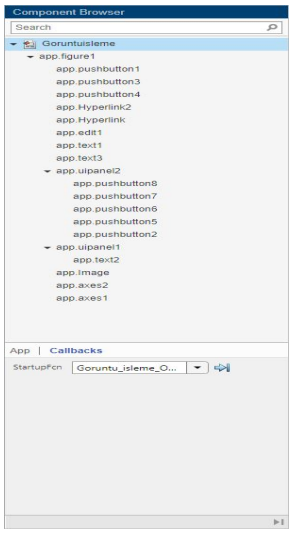
Şekil 1 App Designer Arayüzü

2.2. Şablonun Hazırlanması

Programın varsayılan görünümünde sol tarafta "Component Library" bileşen kütüphanesi bulunmaktadır. Programımda bulunmasını istediğim bileşenler sırasıyla komutları gerçekleştirmek için "buton/button", bilgilendirme ve açıklamalar için "yazı/text", görüntüyü gösterebilmek için "eksen/axes", göze hoş görünmesi için vektörel çizimler ve görseller eklemek için "resim/image" ve farklı kaydetmek için yeni ismini girebileceğimiz "düzenlenebilir yazı kutusu/Edit Field(Text)" bileşenleridir.



Şekil 2 Component Library



Şekil 3Kullandığım Componentler

Bileşen kütüphanesinden kullanacağım bileşenleri istediğim düzende ekrana sürükleyerek konumlandırdım. İsimlendirmelerini yaptıktan sonra callback yani kod içerisinde gerçekleştrimesini istediğimiz fonksiyonlara geri çağırma bağlantılarını yaptım.

2.3. Fonksiyon Kodlarının Yazılması

AppDesigner içerisinden "Code View" görünümüne geçerek uygulamanın otomatik hazırladığı bileşen özellikleri satırları haricinde geri çağrımları başlatma fonksiyonuyla programı başlatma komutunu yazdım ve butonların fonksiyonlarını tanımladım. Resim efekti olarak "fspecial" ve "imfilter" fonksiyonlarını kullandım.

```
%Programın başlangıcı

classdef Goruntuisleme < matlab.apps.AppBase


    % Properties that correspond to app components

    % Uygulama bileşenlerine karşılık gelen özellikler

    properties (Access = public)

        figure1      matlab.ui.Figure

        pushbutton1  matlab.ui.control.Button

        pushbutton3  matlab.ui.control.Button

        pushbutton4  matlab.ui.control.Button
```

```
    Hyperlink2   matlab.ui.control.Hyperlink
    Hyperlink    matlab.ui.control.Hyperlink
    edit1        matlab.ui.control.EditField
    text1        matlab.ui.control.Label
    text3        matlab.ui.control.Label
    uipanel2     matlab.ui.container.Panel
    pushbutton8  matlab.ui.control.Button
    pushbutton7  matlab.ui.control.Button
    pushbutton6  matlab.ui.control.Button
    pushbutton5  matlab.ui.control.Button
    pushbutton2  matlab.ui.control.Button
    uipanel1     matlab.ui.container.Panel
    text2        matlab.ui.control.Label
    Image        matlab.ui.control.Image
    axes2        matlab.ui.control.UIAxes
    axes1        matlab.ui.control.UIAxes
end

% Callbacks that handle component events
% Bileşen fonksiyonlarını işleyen geriçağırmalar
methods (Access = private)

    % Code that executes after component creation
    % Bileşen oluşturulduktan sonra yürütülen kod
    function Goruntu_isleme_OpeningFcn(app, varargin)

        movegui(app.figure1, 'onscreen');
```

```
    [hObject, eventdata, handles] = convertToGUIDECallbackArguments(app);


    handles.output = hObject;


    guidata(hObject, handles);
end

% Button pushed function: pushbutton1
% pushbutton1 isimli butonun fonksiyon kodu
% düzenlenecek resim seçilir soldaki çerçevede önizlemesi verilir
function pushbutton1_Callback(app, event)
    %
    [hObject, eventdata, handles] = convertToGUIDECallbackArguments(app, event);

    [filename pathname]=uigetfile('*.jpg;*.bmp;*.jpeg;*.png;','Resim seç');
    imgname=[pathname filename];
    axes(handles.axes1)
    imshow(imgname);title('Orjinal Resim')
    set(handles.text3,'string',imgname)
end

% Button pushed function: pushbutton2
% pushbutton2 isimli butonun fonksiyon kodu
```

```
% seçilen resime gri tonlama efekt uygulanır sağdaki çerçevede önizlemesi verilir
function pushbutton2_Callback(app, event)
    %
    [hObject, eventdata, handles] = convertToGUIDECallbackArguments(app, event);

    oimg=getimage(handles.axes1);
    grayimg=rgb2gray(oimg);
    axes(handles.axes2)
    imshow(grayimg);title('Gri Tonlama')
end

% Button pushed function: pushbutton3
% pushbutton3 isimli butonun fonksiyon kodu
% kullanıcı için kapat butonu ve yanlışlıkla basılması durumunda teyit penceresi

function pushbutton3_Callback(app, event)
    %
    [hObject, eventdata, handles] = convertToGUIDECallbackArguments(app, event);

    conf=questdlg('Çıkmak istediğine emin misin?','Kapat','Evet','Hayır','Hayır');
    switch conf
        case 'Evet'
            close(gcf)
        case 'Hayır'
            return
    end
end
```

```
% Button pushed function: pushbutton4
% pushbutton4 isimli butonun fonksiyon kodu
% sağ çerçevede önizlemesi yapılan resim aynı isimde jpg olarak kaydedilir

function pushbutton4_Callback(app, event)
    %
    [hObject, eventdata, handles] = convertToGUIDECallbackArguments(app, event);

    img_name=get(handles.edit1,'string');
    pro_img=getimage(handles.axes2);
    imwrite(pro_img,[img_name,'.jpg'],'jpg')
    msgbox('Başarıyla Kaydedildi')
end

% Button pushed function: pushbutton5
% pushbutton5 isimli butonun fonksiyon kodu
% seçilen resime motion blur efekti uygulanır sağdaki çerçevede önizlemesi verilir
%motion komutu bir kameranın doğrusal hareketini tahmin eder

function pushbutton5_Callback(app, event)
    %
    [hObject, eventdata, handles] = convertToGUIDECallbackArguments(app, event);

    oimg=getimage(handles.axes1);
    H=fspecial('motion',20,45);  % h = fspecial('motion',len,theta)
    motion_img=imfilter(oimg,H,'replicate');
```

```
    axes(handles.axes2)
    imshow(motion_img);title('Bulanıklaştırma')
end

% Button pushed function: pushbutton6
% pushbutton6 isimli butonun fonksiyon kodu
% seçilen resime sharpened efekti uygulanır sağdaki çerçevede önizlemesi verilir
% unsharp bir görüntüyü keskinleştirmek için kullanılan bir komuttur.

function pushbutton6_Callback(app, event)
    %
    [hObject, eventdata, handles] = convertToGUIDECallbackArguments(app, event);

    oimg=getimage(handles.axes1);
    H=fspecial('unsharp');
    sharpd_img=imfilter(oimg,H,'replicate');
    axes(handles.axes2);
    imshow(sharpd_img);title('Keskinleştirme')
end

% Button pushed function: pushbutton7
% pushbutton7 isimli butonun fonksiyon kodu
% seçilen resime dairesel bulanıklılık efekti uygulanır
% sağdaki çerçevede önizlemesi verilir
% disk Dairesel ortalama filtresi (ilaç kutusu) komutudur

function pushbutton7_Callback(app, event)
```

```
        %
        [hObject, eventdata, handles] = convertToGUIDECallbackArguments(app, event);



        oimg=getimage(handles.axes1);
        H = fspecial('disk',10);
        blurred = imfilter(oimg,H,'replicate');
        axes(handles.axes2)
        imshow(blurred);title('Disk Blur')
    end

    % Button pushed function: pushbutton8
    % pushbutton8 isimli butonun fonksiyon kodu
    % seçilen resime negatif efekti uygulanır sağdaki çerçevede önizlemesi verilir
    % sobel  yatay kenar vurgulama filtresi olarak kullanılır.

    function pushbutton8_Callback(app, event)
        %
        [hObject, eventdata, handles] = convertToGUIDECallbackArguments(app, event);


        oimg=getimage(handles.axes1);
        H = fspecial('sobel')
        log_img=imfilter(oimg,H,'replicate');
        axes(handles.axes2)
        imshow(log_img);title('Negatif')
    end
end
```

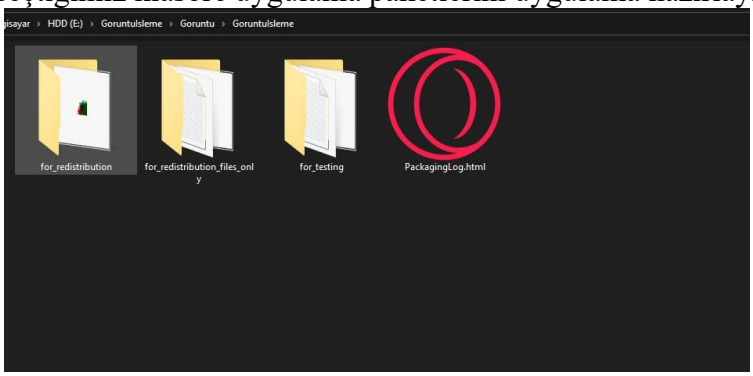
% ANA PROGRAMIN SONU

Kodun devamında oluşturulan bileşenlerin renk boyut bağlantı gibi özellikleri yer almaktadır. AppDesigner tarafından otomatik olarak oluşturulur.

3. UYGULAMANIN KURULUM VE KULLANIMI

3.1. Kurulum

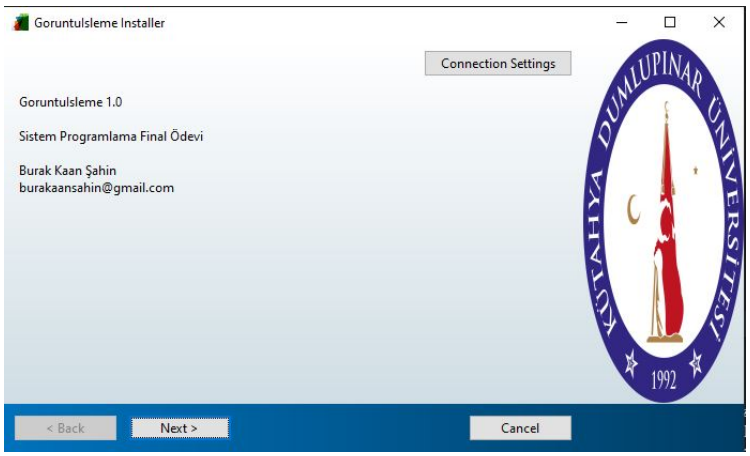
MATLAB App Designer üzerinden hazırladığımız ve kodunu yazdığımız uygulamanın yayınlanması için "Share=>Standalone Desktop APP" seçeneklerini seçip ayarlamaları yaptıktan sonra seçtiğimiz klasöre uygulama paketlerini uygulama hazırlayıp çıkarıyor.



Şekil 4App Designer Tarafından Oluşturulan Uygulama Kurulum Paketi

Şekil 4. Te görünen klasörler içinden "for_redistribution" klasörüne girerek uygulamamızın yükleyicisini çalıştırıyoruz.

Yükleyici ekranında(Şekil 5.) programla ilgili bilgiler görünmektedir. Direktifleri uygulayarak kurulumu yapılır. Bu aşamalarda uygulamayı kurarak ya da kaynak kodu üzerinden çalıştırmak fark etmeksizin MATLAB derleyicisine ihtiyaç duyacağız.

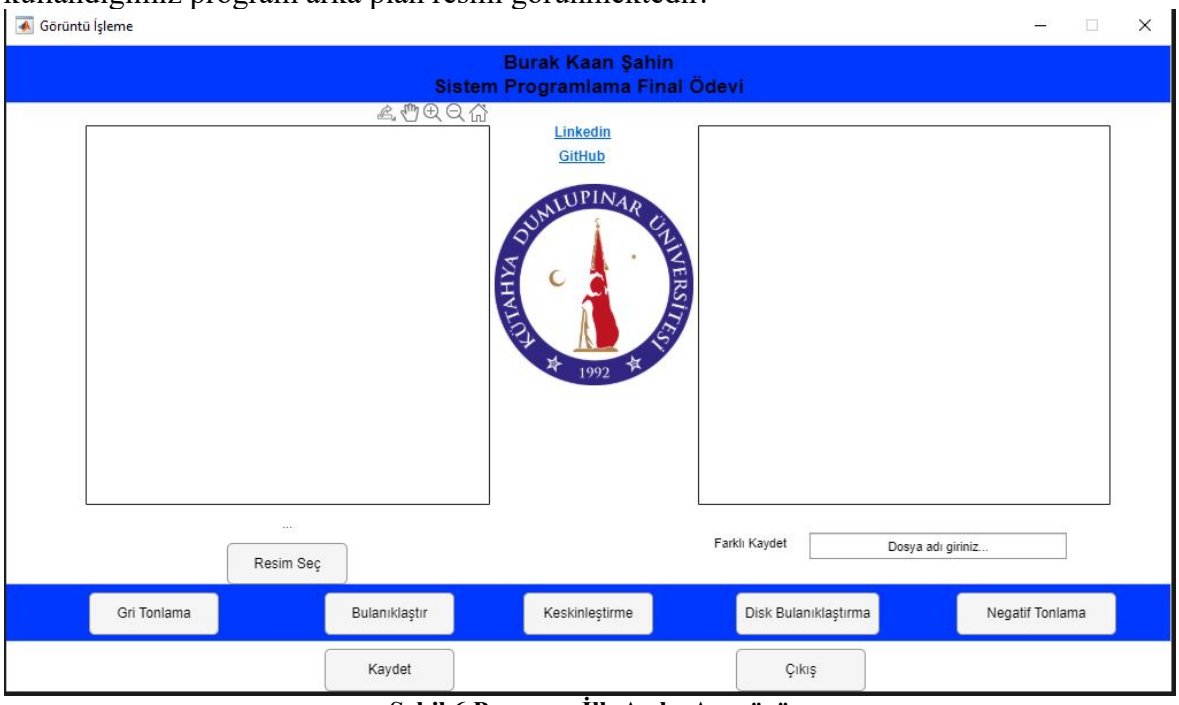


Şekil 5 Yükleyici Ekranı

3.2. Programın Kullanımı

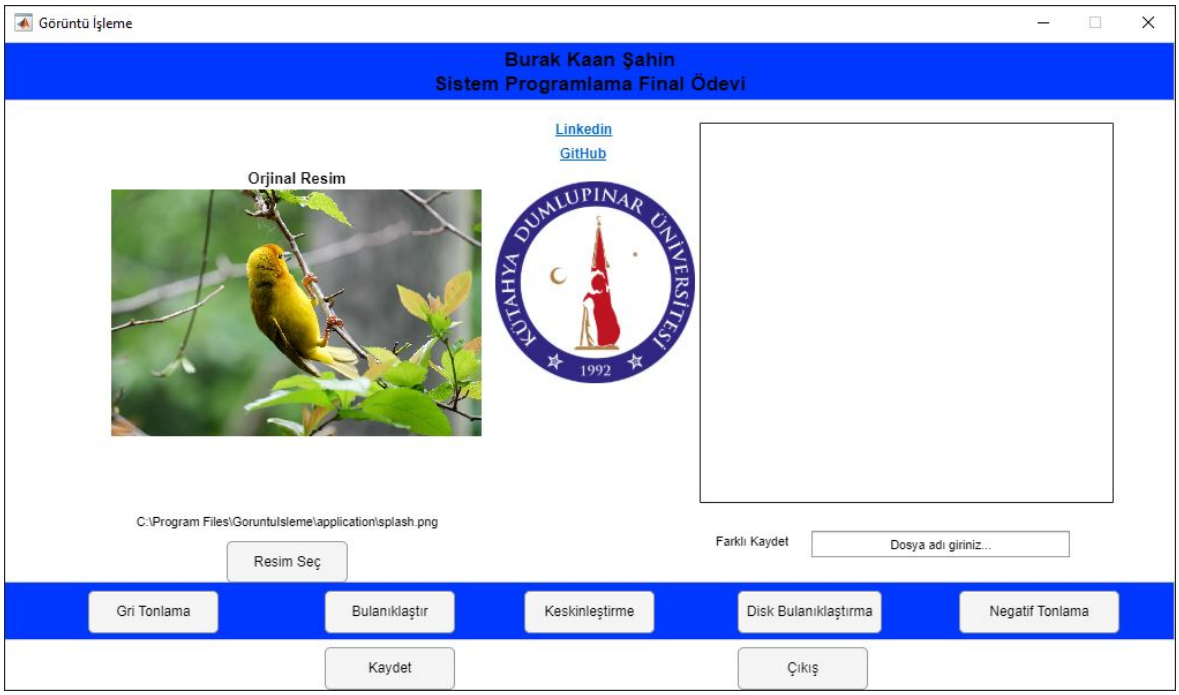
Uygulamanın simgesine çift tıklayarak ya da “.mlapp” ve “.m” dosyalarını aynı klasörde bulundurup “goruntuisleme.m” uzantılı dosyayı çalıştırıp “run” komutunu ya da MATLAB üzerinden bulundukları konumu seçip MATLAB komut penceresine “goruntuisleme” yazarak programı çalıştırabiliriz.

Programı ilk çalıştırdığımızda Şekil 6.’da göründüğü üzere ekranda 2 adet içi boş çerçeve, bilgilendirme yazıları, iletişim linkleri, efekt butonları, komut butonları ve tasarımda kullandığımız program arka plan resmi görünmektedir.



Şekil 6 Program İlk Açılış Arayüzü

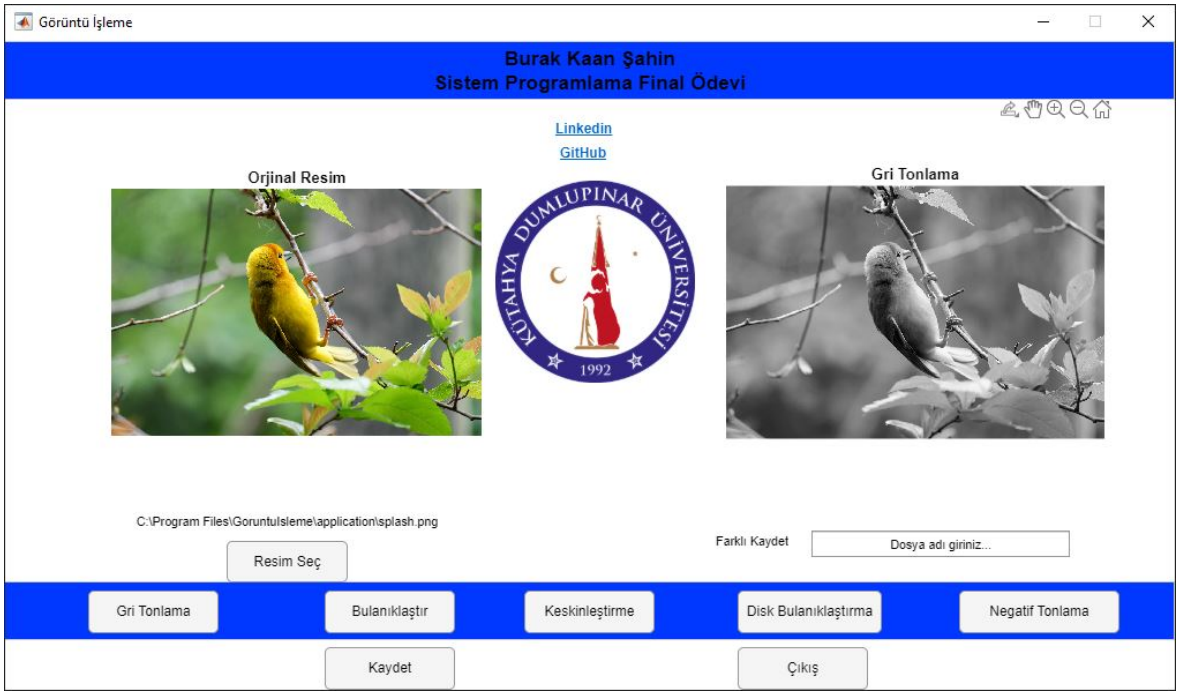
Sol çerçevenin altında bulunan “Resim Seç” yazılı butona tıkladığımızda dosya sistemimizden bir dosya seçmemizi ister. Uygulamamızda “.jpg, .png*, .jpeg*, .bmp*” dosya türlerini düzenleyebileceğimiz için bu dosya türlerinden birini seçeriz. Sol pencerede orjinal(efekt uygulanmamış) resim dosyamız önizlenir. (Şekil 7)



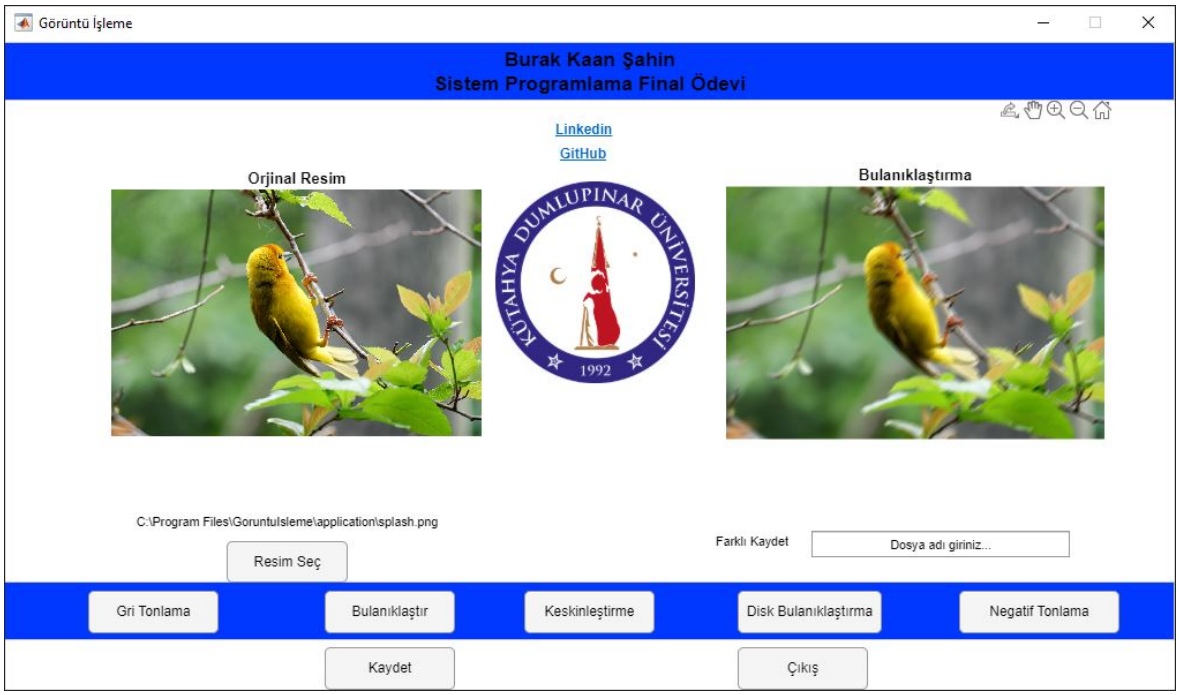
Şekil 7 Resim Seçimi

Efektleri uygulamak için mavi satır üzerinde bulunan butonları kullanarak aynı anda yalnızca bir filtre uygulayabiliriz. Resim dosyasının uygulanan filtreli önizlemesi sağ çerçeve içerisinde görünür.(Şekil 8.)

4. SONUÇ

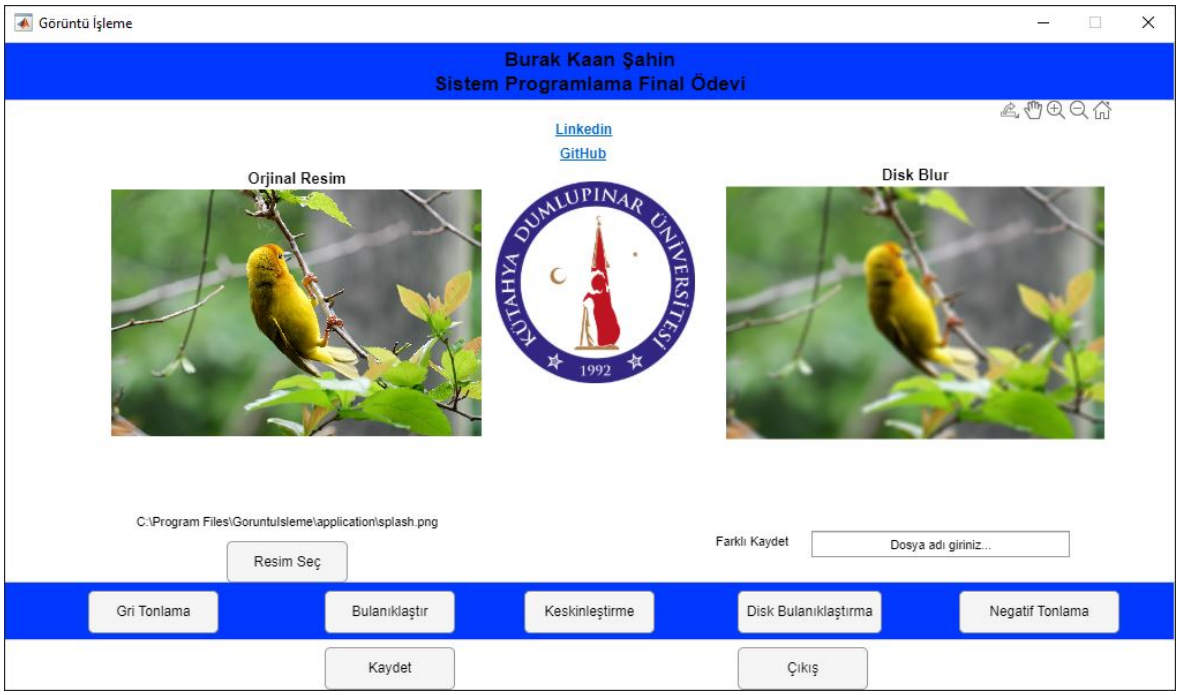


Şekil 8 Gri Tonlama Efektinin Uygulanması

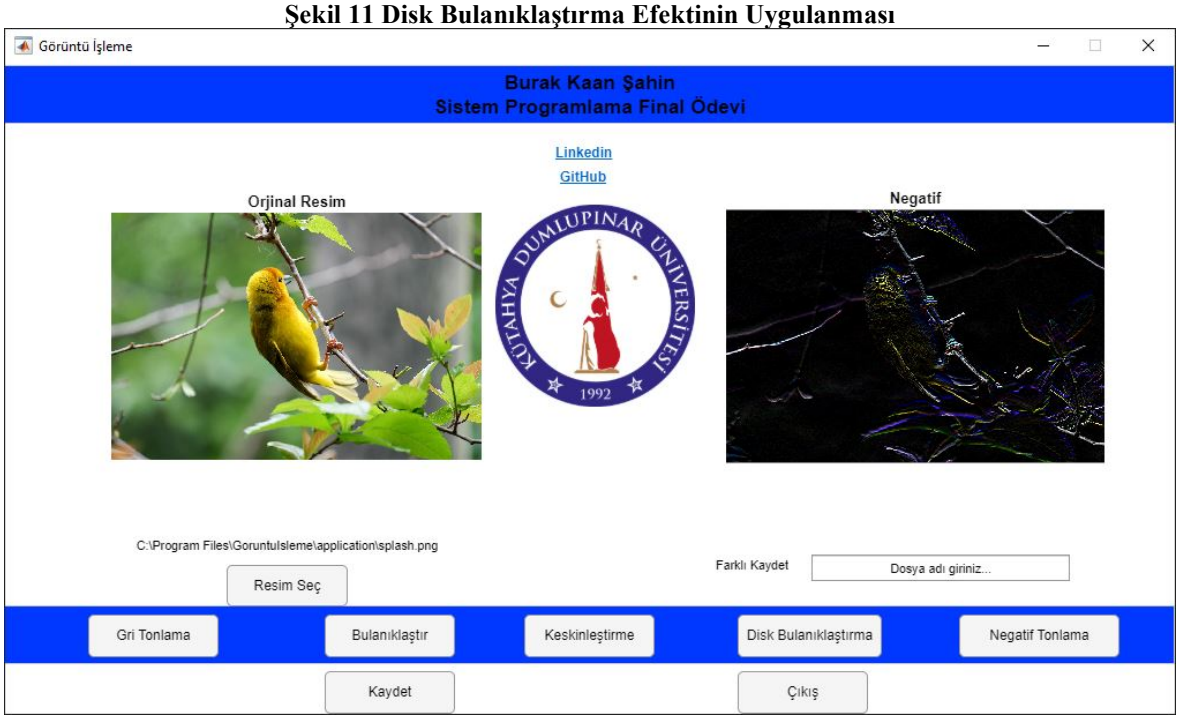


Şekil 9 Bulanıklaştırma Efektinin Uygulanması

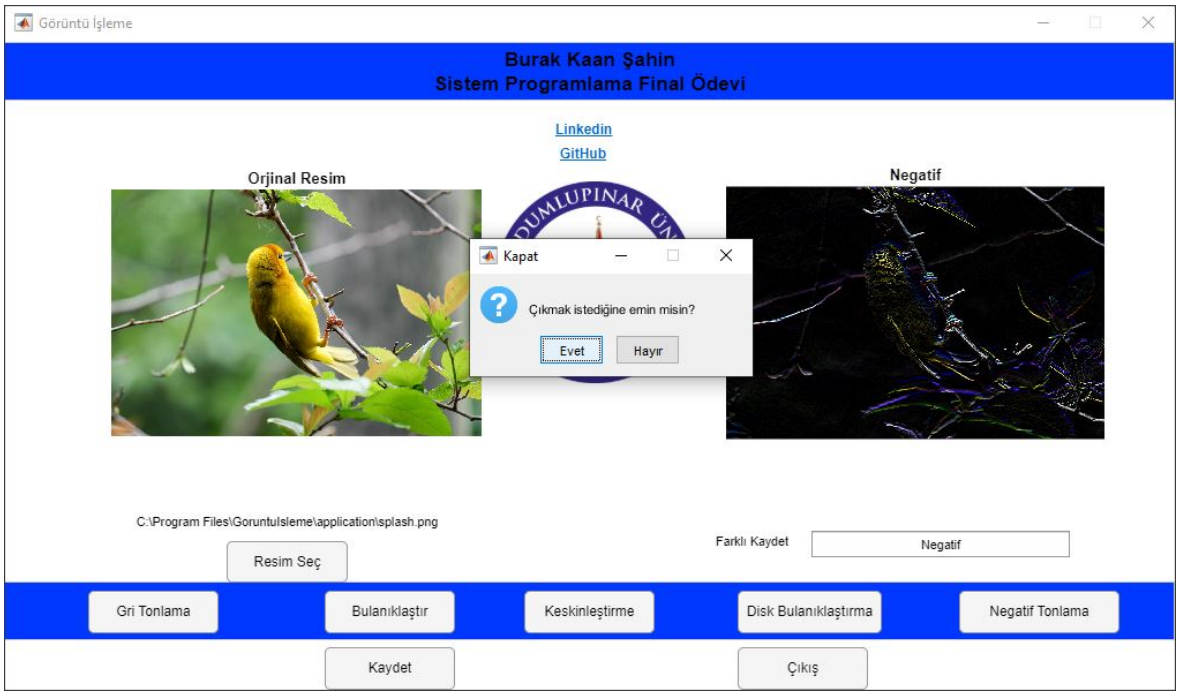
Şekil 10 Keskinleştirme Efektinin Uygulanması



Şekil 11 Disk Bulanıklaştırma Efektinin Uygulanması



Şekil 12 Negatif Efektinin Uygulanması



Şekil 13 "Çıkış" Butonunun Fonksiyonu

MATLAB üzerinden App Designer eklentisini kullanarak görüntü işleme uygulaması ve ara yüzü hazırladık. Bu uygulamanın öncelikle ara yüzünde bulunmasını istediğimiz ve gerçekleştirmesini istediğimiz komutları belirledik ve şablonu oluşturduk. Programlama dili olarak MATLAB kullanarak butonların fonksiyonlarını ve fonksiyonlar içerisinde uyguladığımız filtrelerin kodlarını hazırladık. Uygulamamızın masaüstü yükleyicisini oluşturarak yayınlamaya hazır hale getirdik. Uygulamamızın yükleyicisinden uygulamayı yükledik ve filtrelerimiz sonuçlarını kullanıcı gözüyle gördük.
Bu çalışmada görüntü işleme kullanılarak filtre uygulama uygulaması ve ara yüzü sunulmuştur.

KAYNAKLAR

MATLAB overview, MathWorks internet sitesi