KARADENİZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

HARİTA MÜHENDİSLİĞİ ANABİLİM DALI

PYTHON PROGRAMLAMA DİLİ KULLANILARAK UZAKTAN ALGILAMA AMAÇLI ARAYÜZ TASARIMI

YÜKSEK LİSANS TEZİ

Harita Müh. Ekrem SARALIOĞLU

MAYIS 2015
TRABZON

KARADENİZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

HARİTA MÜHENDİSLİĞİ ANABİLİM DALI

PYTHON PROGRAMLAMA DİLİ KULLANILARAK UZAKTAN ALGILAMA AMAÇLI ARAYÜZ TASARIMI

Ekrem SARALIOĞLU

Karadeniz Teknik Üniversitesi Fen Bilimleri Enstitüsünce
"HARİTA YÜKSEK MÜHENDİSİ"
Unvanı Verilmesi İçin Kabul Edilen Tezdir.

Tezin Enstitüye Verildiği Tarih : 04 / 05 /2015
Tezin Savunma Tarihi : 25 / 05 /2015

Tez Danışmanı : Doç. Dr. Oğuz GÜNGÖR

Trabzon 2015

KARADENİZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

HARİTA MÜHENDİSLİĞİ ANABİLİM DALINDA
Ekrem SARALIOĞLU tarafından hazırlanan

PYTHON PROGRAMLAMA DİLİ KULLANILARAK UZAKTAN ALGILAMA AMAÇLI ARAYÜZ TASARIMI

başlıklı bu çalışma, Enstitü Yönetim Kurulunun 05 / 05 / 2015 gün ve 1601 sayılı kararıyla oluşturulan jüri tarafından yapılan sınavda YÜKSEK LİSANS TEZİ olarak kabul edilmiştir.

Jüri Üyeleri

Başkan : Prof. Dr. Çetin CÖMERT .....

Üye : Doç. Dr. Oğuz GÜNGÖR .....

Üye : Doç. Dr. Levent TAŞÇI .....

Prof. Dr. Sadettin KORKMAZ

Enstitü Müdürü

ÖNSÖZ

"PYTHON PROGRAMLAMA DİLİ KULLANILARAK UZAKTAN ALGILAMA AMAÇLI ARAYÜZ TASARIMI" adlı yüksek lisans çalışmam boyunca danışmanım olarak araştırmalarımı yönlendiren, fikirleri ile vizyonumu geliştiren, çok değerli saygıdeğer hocam Doç. Dr. Oğuz GÜNGÖR' e teşekkür ederim.

Yüksek lisans çalışmam süresince desteklerini benden esirgemeyen saygıdeğer hocalarım Doç. Dr. Recep NİŞANCI, Doç. Dr. Osman DEMİR, Doç. Dr. Volkan YILDIRIM ve Yrd. Doç. Dr. Ebru ÇOLAK hocalarıma teşekkür ederim.

Tez çalışmalarım boyunca manevi desteklerini benden esirgemeyen KTÜ mesai arkadaşlarım, Arş. Gör. Ziya USTA, Arş. Gör. Volkan YILMAZ, Arş. Gör. Yaşar Selçuk ERBAŞ, Arş. Gör. Şevket BEDİROĞLU, Arş. Gör. Tuğba MEMİŞOĞLU arkadaşlarıma ve tez çalışmalarım boyunca bana yardımcı olan ve bilgi, birikimini benimle paylaşan Arş. Gör. Deniz YILDIRIM' a ayrıca teşekkürü bir borç bilirim.

Son olarak hayatımın her anında sevgi ve destekleriyle yanımda olan, üzüntü ve sevinçlerimi paylaşan anneme, babama, kardeşime ve eşim Arş. Gör. Merve MURAT'a en içten teşekkürlerimi bir borç bilirim.

Ekrem SARALIOĞLU

Trabzon 2015

TEZ ETİK BEYANNAMESİ

Yüksek Lisans Tezi olarak sunduğum “PYTHON PROGRAMLAMA DİLİ KULLANILARAK UZAKTAN ALGILAMA AMAÇLI ARAYÜZ TASARIMI” başlıklı bu çalışmayı baştan sona kadar danışmanım Doç. Dr. Oğuz GÜNGÖR’ ün sorumluluğunda tamamladığımı, verileri/örnekleri kendim topladığımı, deneyleri/analizleri ilgili laboratuarlarda yaptığımı/yaptırdığımı, başka kaynaklardan aldığım bilgileri metinde ve kaynakçada eksiksiz olarak gösterdiğimi, çalışma sürecinde bilimsel araştırma ve etik kurallara uygun olarak davrandığımı ve aksinin ortaya çıkması durumunda her türlü yasal sonucu kabul ettiğimi beyan ederim. 25/05/2015

Ekrem SARALIOĞLU

İÇİNDEKİLER

Sayfa No

Yüksek Lisans Tezi

ÖZET

PYTHON PROGRAMLAMA DİLİ KULLANILARAK UZAKTAN ALGILAMA AMAÇLI ARAYÜZ TASARIMI

Ekrem SARALIOĞLU

Karadeniz Teknik Üniversitesi
Fen Bilimleri Enstitüsü
Harita Mühendisliği Anabilim Dalı
Danışman: Doç. Dr. Oğuz GÜNGÖR
2015, 59 Sayfa, 27 Sayfa Ek

Bu çalışmada, Uzaktan Algılama da kullanılan görüntü kaynaştırma yöntemleri (Brovey, IHS ve Dalgacık dönüşümü), kaynaştırılmış görüntülerin kaynaştırma doğruluklarının belirlenmesi için kullanılan kalite değerlendirme metrikleri (ERGAS, SSIM ve CC) ve görüntü sınıflandırma yöntemlerinden ISODATA ve K-means yöntemleri, Python programlama dili 2.7.3 sürümü kullanılarak kodlanmıştır. Daha sonra Ot grafiksel arayüz geliştirme ortamı ile oluşturulan görüntü kaynaştırma, kalite değerlendirme ve sınıflandırma adlı arayüzler oluşturulmuştur. Bu arayüzler sayesinde yukarıda ismi geçen yöntemler kolaylıkla kullanılabilmektedir.

Daha sonra oluşturulan bu arayüzler Coğrafi Bilgi Sistemi içerikli uygulamaların kullanımında lider olan ve uzaktan algılamada kullanılan yöntemlerden bazılarını içeren ArcGIS yazılımına eklenmiştir. Bu sayede ArcGIS yazılımındaki görüntü kaynaştırma, kalite değerlendirme ve sınıflandırmadaki bazı eksiklikler giderilmiş olmaktadır.

Anahtar Kelimeler: Uzaktan Algılama, Python Programlama dili, ArcGIS, Grafik Kullanıcı Arayüzü

Master Thesis

SUMMARY

DESIGNING GRAPHICAL USER INTERFACES FOR REMOTE SENSING APPLICATIONS USING PYTHON PROGRAMMING LANGUAGE

Ekrem SARALIOĞLU

Karadeniz Technical University
The Graduate School of Natural and Applied Sciences
Geomatics Engineering Graduate Programme
Supervisor: Assoc. Prof. Oğuz GÜNGÖR
2015, 59 Pages, 27 Pages Appendix

In this study, we have devised and implemented Graphical User Interfaces (GUI) for remote sensing applications using Qt library and Python programming language. Using these GUI, a researcher can apply fusion with popular image fusion techniques, including IHS, Brovey, and wavelet-based techniques. One can also assess the quality of the fused images using fusion quality assessment metrics ERGAS, SSIM and CC. Furthermore, with these GUI, one can classify images using ISODATA and K-means classification methods. In addition to the standalone versions, we have implemented the GUI as ArcGIS addins, thereby extending ArcGIS, the leading software used for Geographic Information Systems applications, for remote sensing researchers. These GUI can be extended with other fusion techniques, quality assessment metrics and classification methods.

Key Words: Remote Sensing, Python Programming Language, ArcGIS, Graphical User Interface

ŞEKİLLER DİZİNİ

Sayfa No

TABLOLAR DİZİNİ

Sayfa No

SEMBOLLER DİZİNİ

AKK	: Açık kaynak kod
API	: Application Programming Interface
CBS	: Coğrafi Bilgi Sistemleri
CC	: Correlation Coefficient
ERGAS	: Erreur Relative Globale Adimensionnelle de Synthése
ESRI	: Environmental Systems Research Institute Inc.
GDAL	: Geospatial Data Abstraction Library
GKA	: Grafik Kullanıcı Arayüzü
IBM	: International Business Machines
IDLE	: Integrated DeveLopment Environment
IHS	: Intensity-Hue-Saturation
ISODATA	: The Iterative Self-Organizing Data Analysis Technique
NTP	: Nesne Tabanlı Programlama
NET	: Network
NSA	: National Security Agency
SSIM	: Structure Similarity Index
VBA	: Visual Basic For Application
XML	: Extensible Markup Language

1. GENEL BİLGİLER

1.1. Giriş

Uzaktan algılama yeryüzündeki nesnelere ait bilgilerin fiziksel temas olmaksızın yansıyan ve yayılan enerjinin kaydı ve analizi ile elde edilmesini konu alan bir bilim dalıdır. Uydular ve uçaklar yeryüzüne ait bilgilerin uzaktan algılanarak elde edilmesinde kullanılan ortak platformlardır (Sanderson, 2010). Günümüzde uzaktan algılama verilerinin elde edilmesinde kamera ve sensörlerle donatılmış insansız hava araçlarının da kullanımı yaygınlaşmaktadır.

"Coğrafi Bilgi Sistemleri (CBS); konuma dayalı işlemlerle elde edilen grafik ve grafik-olmayan verilerin toplanması, saklanması, analizi ve kullanıcıya sunulması işlevlerini bir bütünlük içerisinde gerçekleştiren bir bilgi sistemidir" (Yomralıoğlu, 2009). CBS teknolojisi, sorgulama ve istatistiksel analiz gibi klasik veritabanı işlemlerini görselleştirme ve haritalar tarafından sağlanan coğrafi analizlerle birleştirmektedir. Bu yeteneği CBS' yi diğer bilgi sistemlerine göre farklı kılmaktadır. Bunun bir sonucu olarak, CBS, hizmet alanındaki olayların tanımlanmasında ve ileriye dönük tahminlerde bulunarak stratejik planların yapılmasında kamu ve özel sektör tarafından oldukça yoğun bir şekilde kullanılmaktadır.

Uzaktan algılama yöntemleriyle elde edilmiş çeşitli verilerin Coğrafi Bilgi Sistemleri'nin (CBS) etkileşimi ile alışılagelmiş yöntemlere kıyasla hedefe daha hızlı, daha ucuz, daha az insan gücüyle ulaşılabilmektedir. Bu da uzaktan algılama yöntemlerinin yaygın bir kullanım alanının oluşmasını sağlamıştır (Kavzoğlu ve Çölkesen). Ormancılık, tarım, jeoloji, doğal kaynak, arazi örtüsü tespiti, arazi yönetimi, bölge ve şehir planlama, kirlilik analizleri, kent bilgi sistemleri, değişim analizleri, sürdürülebilir çevre uygulamaları, çeşitli amaçlar için en uygun alanların belirlenmesi, sayısal arazi modeli üretimi, tematik ve topografik haritaların üretimi, 3 boyutlu şehir haritaları üretimi, yeryüzü deformasyonlarının izlenmesi bunlardan bazılarıdır.

Bilgi çağı olarak adlandırılan günümüzde; Coğrafi Bilgi Sistemleri ve Uzaktan algılama teknolojileri, karmaşık planlama ve yönetim sorunlarının çözülebilmesi için etkin araçlar sağlamaktadır. Özellikle Avrupa Birliği'ndeki coğrafi veri ve yönetimi ile ilgili

gelişmeler ülkemizdeki çalışmaları da hızlandırmış, verilerin ulusal ve uluslararası kullanım ihtiyacı, yeni teknikler, yeni standartların oluşmasını zorunlu kılmıştır (URL-1, 2015).

Coğrafi Bilgi Sistemleri ve uzaktan algılama teknolojilerinin gelişmesine paralel olarak ArcGIS yazılımını kullanan kuruluş ve uzman sayısı da gün geçtikçe artmaktadır. ESRI firması tarafından geliştirilen ArcGIS yazılımı; kamu, özel sektör ve akademik alanda yaygın kullanımda olan bir CBS yazılımıdır. Uzaktan algılama ve görüntü işleme uygulamalarında yapılabildiği ArcGIS yazılımı C, Java, Python, Android, IOS, Silverlight gibi farklı yazılım geliştirme dillerini desteklemektedir. Ayrıca dünya genelinde geliştiriciler için geniş bir kütüphaneye ve destek sistemine sahiptir. Günümüzde CBS yazılım piyasasının %50'sinden fazlasını elinde bulunduran ArcGIS yazılımı üzerinden geliştirilmiş sayısız arayüz, eklenti vb. uygulama bulunmaktadır. Dünya çapında konumsal bilimler hakkında bilimsel araştırma yapan birçok geliştirici bu yazılımı kullanmakta, masaüstü CBS, mobil CBS ve Web CBS uygulama araçları geliştirmektedir. 40' dan fazla raster veri yapısı destekleyen ArcGIS yazılımı geliştiricilere imkan sağlayan açık yapısı ile CBS uygulamalarının yanında uzaktan algılama ve görüntü işleme uygulamalarını da desteklemektedir **(**Xua ve Gaob, 2008).

ArcGIS 9.0 sürümü ile Python programlama dilini tanıtmıştır. Sonraki yayınlanan her sürümünde de Python'u kullanmaya devam etmiştir. Python programlama dili etkin, çoğu işletim sisteminde kolaylıkla kullanılabilen, okunabilirliği açık, kod yazılımı sade, yüksek seviyeli, nesne tabanlı bir dildir. Standart kütüphaneleri ve modülleri ile birlikte özellikle bilimsel alanda yaygın olarak kullanılan diller arasına girmiştir. Kolay öğrenilen ve güçlü bir programlama dili olan Python verimli yüksek seviyeli veri tiplerine sahiptir. Python yorumlayıcısı C veya C++ (ya da C dilinden çağırılabilen başka bir dil) ile yazılmış veri tipleri ve fonksiyonlar ile genişletilebilir. Diğer dillerde yazılan programlara da Python yorumlayıcısı bağlanabilir ve Python ile ek özellikler eklenebilen programlar yazılabilir (Rossum, 2003). Ayrıca Python programlama dili etkin ve kullanışlı grafik kullanıcı arayüzlerinin (GKA) tasarlanabileceği çok geniş araçlar içermektedir. GKA bilgisayarlarda işletilen komutlar ve bunların çıktıları yerine simgeler, pencereler, düğmeler ve panellerin tümünü ifade etmek için kullanılan genel addır. Grafiksel kullanıcı arayüzü, bilgisayar kullanıcılarının komut satırı kodlarını ezberlemeden fare, klavye gibi araçlar sayesinde bilgisayarları kontrol etmelerini sağlar. Günümüzdeki programların birçoğu GKA ile birlikte piyasaya sürülmektedir (Martinez, 2011).

Bu çalışmada Python programlama dili kullanılarak çeşitli uzaktan algılama uygulamalarını içeren bir arayüz oluşturulmuştur. Ayrıca oluşturulan bu arayüz CBS piyasasında ve akademik alanda yaygın kullanımda olan ArcGIS yazılımının 10.2 sürümüne eklenmiştir.

1.2. Problemin Tanımı

Uzaktan algılama ve CBS uygulamalarının otomatik olarak gerçekleştirildiği pek çok ticari yazılım mevcuttur. Yazılım birçok araştırma ve ticari kuruluş tarafından kullanıcı taleplerini karşılamak amacıyla uzun uğraşlar neticesinde hazırlanan bir dizi algoritmanın programlama dilleri ile kodlanıp kullanıcıların rahat bir şekilde kullanabilmesi için hazırlanan arayüzlerle desteklenerek oluşturulur (Yomralıoğlu, 2009). Temel olarak literatürde yaygın olarak kullanılan yöntemleri içeren bu yazılımlar, programlama teknikleri, içerdikleri fonksiyonlar, çalışabildikleri platformlar, içerdikleri arayüz tasarımları ile birbirlerinden farklılıklar gösterirler. Ayrıca çeşitli uygulamalarda bu yazılımlar eksiklikler içermektedir. Literatüre yeni eklenen yöntemlerin çoğu ticari yazılımlara eklenmemektedir. Ayrıca yazılımların yapamadığı ama gerekli olan bazı durumlar da yapılan çalışmalarda ortaya çıkmaktadır. Örneğin CBS alanında oldukça etkin bir yazılım olan ArcGIS yazılımı uzaktan algılama uygulamalarına da yer vermektedir. Fakat ArcGIS yazılımında yapılan bu uygulamalar yetersiz kalmaktadır. Bu tarz durumlar yazılımlara ek programlama dillerinin de kullanılmasıyla çözülebilir. Bu nedenle bu çalışmada Python programlama dili kullanılıp çeşitli uzaktan algılama uygulamaları ile ilgili betikler yazılmıştır. Yazılan bu betikler ile bir arayüz oluşturulmuştur. Daha sonra oluşturulan bu arayüz CBS alanında çokça tercih edilen bir yazılım olan ArcGIS 10.2 yazılımına entegre edilmiştir.

1.3. Çalışmanın Amacı

Bu çalışmanın amacı, uzaktan algılamada görüntü işleme, sınıflandırma, kaynaştırma gibi uygulamaların Python Programlama dili ile kodlanarak bir arayüz oluşturulmasıdır. Oluşturulacak bu arayüz güncellenebilecek ve yeni bulunan yöntemler bu arayüze eklenebilecektir. Oluşturulan bu arayüz ile CBS, görüntü işleme konuları ile ilgilenen

uzaktan algılama araştırmacıları için popüler CBS yazılımı olan ArcGIS yazılımının güncel sürümüne (10. 2) bir eklenti (“Add-in”) oluşturulacaktır.

Bu amaçla, bu eksikliği giderebilmek için programlama dili olarak Python (2. 7. 3) sürümü seçilmiştir. Çünkü Python’un bu sürümü ArcGIS 10.2 yazılımına entegre olarak gelmektedir. Kullanılan modüller arasında yer alan PyQt4 modülü ile Qt kütüphanesine bağlanılabilmektedir.

Bu dille bir görsel arayüz ve ArcGIS 10. 2 eklentisi oluşturulmuştur. Oluşturulan arayüzlerle, temel uygulamalar olan görüntü kaynaştırma (Dalgacık dönüşümü, Brovey, IHS), görüntü sınıflandırma (k-means ve ISODATA), kalite değerlendirme (kaynaştırma için ERGAS, SSIM, CC,) metrikleri Python programında arayüz olarak hazırlanmış ve ArcGIS yazılımına hazırlanan bu arayüz entegre edilmiştir.

1.4. Metodoloji

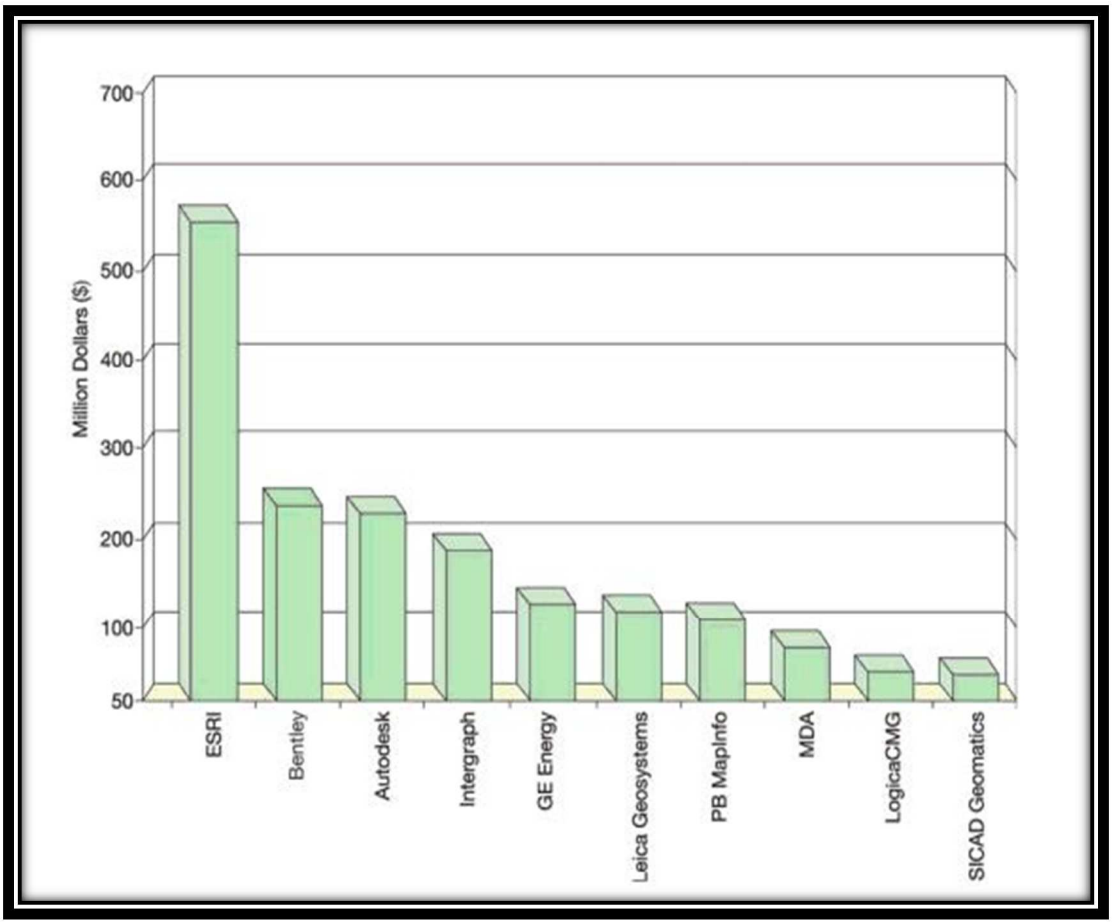
Bu yüksek lisans tezi kapsamında yapılan çalışmalar aşağıdaki gibidir:

- Python programlama dilinde betik yazma için temeller.
- Grafik arayüz tasarımı için uygun kütüphanenin belirlenmesi ve arayüz hazırlama.
- Görüntü kaynaştırma, görüntü kaynaştırmada kalite değerlendirme metrikleri ve sınıflandırma uygulamalarından bazıları Python programlama dili ile kodlanıp bu kodların hazırlanan arayüze eklenmesi,
- Oluşturulan arayüzün ArcGIS yazılımına entegre edilmesi işlem adımlarını içermektedir.

1.5. ArcGIS Yazılımı

ArcGIS yazılımı Coğrafi Bilgi Sistemleri piyasasının en önemli çözüm üreticilerinden biri olan 1969 yılında California-ABD’ de kurulan ESRI (Environmental Systems Research Institute Inc.) firması tarafından geliştirilmekte olan bir CBS yazılımdır. 1982 yılında IBM sistemleri için geliştirilen Arc/Info ile yazılım sektörüne girmiştir. 1991 yılında masaüstü kullanıcıları için MS Windows üzerinde çalışan ArcGIS geliştirildi. ESRI ürünleri (özellikle ArcGIS Masaüstü) ARC Advisory Group tarafından yapılan araştırmalara göre hazırlanan 2010 raporunda, Konumsal Bilgi Sistemleri alanında dünya piyasasının % 40,7’sini oluşturmaktadır (Francica, 2011). Dünya genelinde 9000 civarı

kişinin çalıştığı ESRI firması yıllık geliri 2013 istatistiklerine göre 900 milyon doların üzerindedir. Günümüzde 350.000 den daha fazla kurum ve kuruluşa hizmet etmekte olan ESRI 1 milyonu aşkın kullanıcı kitlesine sahiptir (Dangermond, 2015).



Şekil 1. CBS yazılım üreticilerinin pazar payı (Longley, 2012)

ESRI firması CBS pazarında yazılım geliştirmeye odaklı bir firmadır fakat yazılım geliştirmenin ötesinde de CBS hizmetleri ve danışmanlık da yapmaktadır. Günümüzde ESRI firmasının odağı bütünleşik yazılımlardan oluşturduğu ArcGIS yazılımı üzerine odaklıdır. ArcGIS ailesini kullanıcılar (özellikle profesyonel ve teknik), geliştiriciler, el cihazları, kişisel bilgisayarlar, sunucular ve web hizmetleri oluşturur (Longley, Goodchild, Maguire ve Rhind, 2010). ArcGIS coğrafi bilgilerin toplanması, organize edilmesi, yönetimi ve dağıtılmasına izin veren kapsamlı bir sistemdir. Dünyanın önde gelen CBS platformu olan ArcGIS, dünya çapında insanlar tarafından kamu, bilim, eğitim, medya gibi alanlarda kullanılmaktadır. ArcGIS coğrafi bilgilerin yayınlanmasını mümkün kılar bu

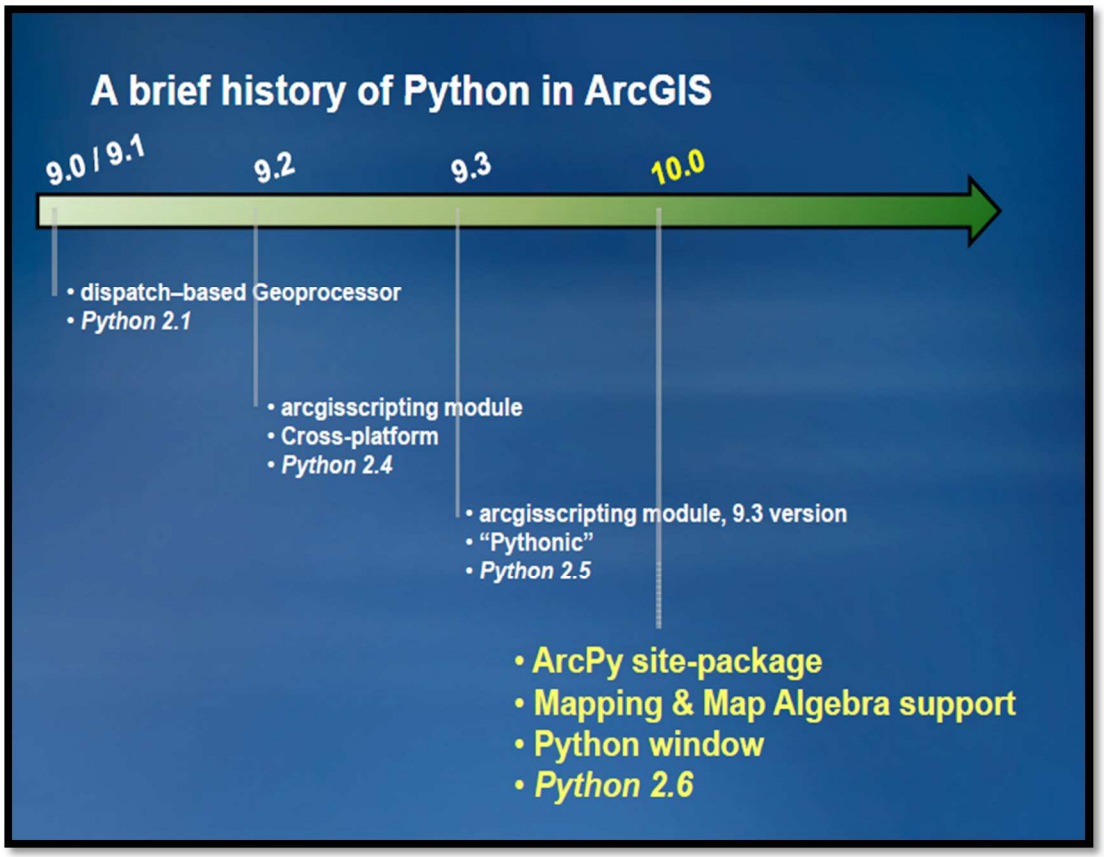
sayede bu yazılımı kullanan herkes bu bilgilere ulaşabilir. Bu sistem her türlü bilgisayarlarda, internet tarayıcıda ve akıllı telefonlarda, kullanılabilir (URL-2, 2015). Aşağıda Şekil 2' de ArcGIS yazılımı kullanım platformları gösterilmektedir.

Şekil 2. ArcGIS yazılımı kullanım platformları

1.5.1. ArcGIS Yazılımı ve Python Programlama Dili Desteği

ArcGIS 9.0 sürümü ile Python yazılımı ArcGIS kullanıcılarına tanıtılmıştır. O zamandan beri Python yazılımı betik dili olarak kullanılmaktadır. Yayınlanan her bir ArcGIS sürümünde Python kullanım alanı genişletilmekte ve daha kullanıcı dostu bir hale getirilmektedir. Python Arcgis10.0 sürümü ile ArcGIS yazılımında çok yaygın bir rol oynamaya başlamıştır. Esri böylece Python programlama dilini ArcGIS kullanıcıları için tamamen benimsemiş olmuştur. (Esri Online Help). ArcGIS, Python'u benimsemesi sayesinde iş akışlarını otomatikleştirmek için birçok fırsata sahip olmuştur. Bu süreçte Python, Field Calculator bölümünde Visual Basic For Application (VBA) yazılımının yerini almıştır. Ayrıca ArcGIS yazılımına yönetici penceresi bölümüne Python penceresi eklenmiştir.

ArcMap ve ArcCatalog için VBA'nın ArcGIS 10' da ve sonraki sürümlerde kullanımı desteklenmemektedir. VBA programlama dili ArcGIS Desktop 10.0 sürümünde olduğu gibi yeni yayımlanacak sürümlerde de ArcGIS ile birlikte yüklenemeyecek. Bu nedenle VBA makrolarıyla çalışmaya devam edecek geliştiriciler ESRI'den VBA lisansı talep etmek zorunda kalacaklar. Hatta ArcGIS yazılımının ötesinde ESRI artık bir geliştirme platformu olarak VBA' yı desteklememektedir. Betik yazmada VBA'yı kullanan kullanıcıların, Python, VB .NET veya C # gibi ArcGIS tarafından desteklenen bir geliştirme dilini kullanarak kendi uygulamalarını yeniden yazmak için stratejiler planlaması uygun olacaktır (ArcGIS Resorces, 2015). Böylece kodlanan uygulamalar ArcGIS' in sonraki yayımlanan sürümleri için de kullanılabilir olmuş olacaktır.



Şekil 3. ArcGIS yazılımı Python desteği (Honeycutt ve Wynne,2010)

1.5.2. ArcGIS Add-in Teknolojisi

ArcGIS Add-in, ArcGIS masaüstü uygulamaları (ArcMap, ArcCatalog, ArcGlobe ve ArcScene) için araç çubuğu, menü, buton, açılan kutu gibi elemanlar içeren bir eklentidir. ArcGIS 10.0 sürümünden beri, kullanıcılara, kullanıcı ara yüzünü değiştirebilmeleri için kolay bir yol sağlamaktadır. En başta NET ve Java kullanılarak oluşturulurken, ilk sürüm değişikliğiyle (ArcGIS 10.1 sürümü), Python, XML çifti tercih edilmeye başlanmıştır (Pimpler, E., 2013). ArcGIS 10.1 sürümü ile masaüstü işlevselliğini genişletmek ve kolay bir çözüm sunmak amacıyla, masaüstü eklentileri için Python programını tanıtmıştır. Eklentilerin yüklenebilmesi için Python Eklentisi Sihirbazının kullanılması gerekmektedir. Bir Python eklentisi olan ".esriaddin" uzantılı dosya tek bir sıkıştırılmış dosyadan oluşmaktadır. Bu dosya aşağıdaki öğeleri içerir:

- config.xml — yazar, sürüm, başlık, vb kategori için -Bir Genişletilebilir İşaretleme Dili olan (XML) dosyası.
- Python script — Python kodlarını içeren (.py file) çalışma dosyası.
- Kaynak dosyası — eklentiyi desteklemek için kullanılan, ek veri dosyasıdır.

Add-in yapısı ArcGIS yazılımına Python programlama dili ile yazılan betiklerin eklenmesini sağlar (Esri Online Help, 2015).

1.6. Python Programlama Dili

1.6.1. Python Programı

Python 1991 yılında Guido Van Rossum tarafından geliştirilen çok güçlü bir yüksek-seviyeli, dinamik nesne yönelimli programlama dilidir. (Harwani, B. M., 2012) Python platformdan bağımsız bir programlama dilidir. Bu sayede Linux, Windows, Mac OS X, BSD, Solaris, AIX, AROS, AS/400, BeOS, MorphOS, S60, iPOD, iPhone, Android ve Macintosh dahil tüm büyük donanım platformu ve işletim sistemleri üzerinde çalışabilmektedir. Ayrıca Python programlama dilinin basit ve temiz söz dizimi, onu Eric S. Raymond, diğer pek çok programcı ve Google tarafından tercih edilen bir dil haline getirmiştir, Ruby ve Perl gibi alternatiflerin önüne geçmiştir (Raymond, E., 2000; Swaroop, C. H., 2003). Python dilinin söz diziliminin basit olması sayesinde hem program

yazmak, hem de başkası tarafından yazılmış bir programı okumak, başka dillere kıyasla daha kolaydır (Özgül, F., 2013).

1.6.2. Günümüzde Python Programı Kullanımı

Dünya çapında Python kayıtlı kullanıcı veri tabanlarına göre yaklaşık 1 milyon Python kullanıcısı vardır. Bu tahmin, indirme oranları, web istatistiklerine ve geliştiricilerin yaptığı anketlere dayalıdır. Genel olarak, Python geniş bir kullanıcı kitlesine ve çok aktif bir geliştirici topluluğuna sahiptir. Dünyanın en iyi ve en yaygın kullanılan ilk on programlama dillerinden biri olarak kabul edilir. Çünkü Python yaklaşık yirmi yıldır geniş çapta kullanılmakta ve geliştirilmektedir. Bu da onu oldukça stabil ve kuvvetli bir program yapmaktadır. Bireysel kullanıcılar tarafından kullanılmasının yanı sıra Python şirketler tarafından gelir sağlayıcı uygulamalar oluşturmak için de yoğun olarak kullanılmaktadır. Aşağıda Python programı kullanan ünlü firmalar yer almaktadır.

- Google gibi web arama motorları Python programını geniş çapta kullanmaktadır.
- Populer vidyo paylaşım sitesi olan YouTube büyük ölçüde Python ile yazılmıştır.
- Dropbox depolama servisinde birincil yazılım olarak Python kullanılmıştır.
- EVE Online internet üzerinde oynanan oyunlarda yaygın bir şekilde Python kullanır.
- Yaygın olarak kulanılan BitTorent paylaşım sitesi Python ile geliştirilmektedir.
- Industrial Light & Magic, Pixar ve diğerleri animasyon film üretiminde Python programlama dilini kullanmaktadır.
- ESRI ve onun popüler CBS harita ürünleri, kullanıcıların geliştirebileceği araçlarda Python programlama dilini tercih ediyor.
- The IronPort eposta sunucusu yaptığı iş için bir milyondan fazla Python kod satırı kullandı.
- Maya, güçlü entegre 3D modelleme ve animasyon sistemi, Python betik API'sini kullanmaktadır.
- NSA (National Security Agency) kriptografi ve istihbarat analizi için Python kullanmaktadır.
- iRobot ticari ve askeri robotik aygıtların geliştirilmesinde Python kullanmaktadır.

- Intel, Cisco, Hewlett-Packard, Seagate, Qualcomm, ve IBM donanım testi için Python kullanmaktadır (Lutz, 2013).

1.6.3. Python Programlama Dili Özellikleri

Python programı yüksek seviye dilli bir programdır. Ayrıca düşük seviyeli diller de vardır. Bunlar bazı durumlarda "makine dili" veya "birleştirici dili" şeklinde isimlendirilir. Bilgisayarlar sadece düşük seviyeli dillerde yazılmış programları çalıştırabilir. Bu yüzden, yüksek seviyeli dillerde yazılmış programlar çalıştırılmadan önce bir işlemden geçmelidir. Yüksek seviyeli dilleri düşük seviyeli dillere çevirmek için yorumlayıcılar ve derleyiciler kullanılır. Yorumlayıcılar yüksek seviyeli programı okur ve işletir. Derleyiciler programı okur ve programı çalıştırmadan önce farklı bir kod haline dönüştürür. Bu durumda yüksek seviyeli programa kaynak kod, çevrildiği programa ise hedef kod denir. Program derlendikten sonra, daha fazla çevirme yapmadan tekrar tekrar çalıştırılabilir. Bu durum yüksek seviyeli dillerin dezavantajıdır. Ancak yüksek seviyeli dillerin avantajları da oldukça fazladır. İlk olarak yüksek seviyeli diller, düşük seviye dillere göre makineye daha az bağımlıdır ve düşük seviyeli bir dilde yazılmış birçok satır yüksek seviyeli bir dilde, tek bir komutla gösterilebilir. Bu nedenle yüksek seviyeli dilde yazılmış programlar daha az sürede yazılır, daha kısadır, okuması daha kolay ve doğru olma ihtimalleri daha yüksektir. Yüksek seviyeli diller bilgisayarlarda değişiklik yapmadan çalıştırılabilirler. Düşük seviyeli programlar ise sadece tek bir bilgisayar çeşidinde çalışabilirler ve başka bir bilgisayarda çalışabilmesi için tekrar yazılmalıdır (Elkner vd, 2008).

Python programı nesne tabanlı bir programlama dilidir. Nesne tabanlı programlamada (NTP) temel amaç, yapılacak uygulamanın parçalara bölünerek yapılmasıdır. Program yazılırken sınıflar ve bu sınıfların altında nesneler kullanılır. NTP ile yazılan bir programda yeni bir nesne oluşturulurken bu nesne mutlaka bir sınıftan üretilmek zorundadır. Bu yüzden yapılan çalışmalar daha düzenli olur. NTP ile yazılmış olan bir kod bloğu istenildiği zaman tekrar çağırılıp kullanılabilir ve farklı projelere aktarılabilir. NTP' nin temel özelliklerinden biri olan kalıtım sayesinde bir sınıftan başka sınıflar türetilebilir bu sayede her sınıf için ortak olan özelliklerin bir daha kodlanmasına gerek kalmaz ve bu programlamada oldukça kolaylık sağlar.

Python çok geniş kütüphaneleri içeren bir programlama dilidir. Çeşitli alanlardaki bu kütüphaneler; belge oluşturma, veri tabanları, web tarayıcıları, kartografya, grafiksel

kullanıcı arayüzü (GKA) , konsol uygulamaları, görüntü işleme, bilimsel hesaplamalar gibi pek çok alanda uygulamaların yapılmasına yardımcı olmaktadır.

Python ücretsiz ve açık kaynak kodlu bir programlama dilidir. Python programlama dili, standart sürümünde, C dili kullanılarak gerçekleştirilmiştir. Bu kodlar ile Python'un standart kütüphanesi, geliştirme araçları ve diğer birçok kütüphane, açık kaynak kod (AKK) olarak İnternet'ten ücretsiz ve herhangi bir lisans sorunu yaşanmadan indirilebilmektedir (Malkoç, 2010). Bu sayede program ücretsiz olarak dağıtılabilir ve kullanılabilir. Açık kaynak kodlu yazılımların en büyük ayırt edici özelliği kullanıcıya yazılımı değiştirme imkanını sağlamasıdır. Bu avantalarının yanı sıra Python programının ücretsiz olması, onun herhangi bir sürümümün indirilip ticari veya ticari olmayan uygulamalar için para ödemeden kullanılması anlamına gelmektedir.

Python platformdan bağımsız bir programlama dilidir. Bu nedenle herhangi bir sistemde (Windows, Linux, Macintosh, Solaris, AIX, AROS, AS/400, BeOS, MorphOS, S60, iPOD, iPhone, Android) programda değişiklik yapılmadan yazılabilir ve taşınabilir. Bu nedenle dosya oluşturmaya ve programı farklı platformlara uyarlamaya gerek kalmaz (Malkoç, 2010).

Python hafıza yönetimi konusunda etkilidir. Hafıza azlığı konusunda endişelenmeye gerek yoktur çünkü Python çöp toplama kullanır. Eğer bir sınıf örneklenmezse o sınıf program tarafından otomatik olarak çöp toplama işlemine tabi olur. Bu işlem ile kullanılmayan veriler hafızadan silinir. Bu sayede etkili bir hafıza yönetimi uygulanmış olup programların performansı artmış olur.

Python programının yaygın olarak kullanılmasının en önemli sebeplerinden biri de hesaplamalı bilimlerdeki işlevselliğidir. Python programının bu fonksiyonelliğini modül, paket veya kütüphane olarak adlandırılan eklentiler artırmaktadır. Bu paketlerin en bilinenlerinden biri NumPy ve SciPy kütüphaneleridir. NumPy genel olarak dizi/vektör/matris hesaplamaları için özelleşmiş bir kütüphane olup, bu kütüphane ile matematiksel ve mantıksal işlemler, sıralama, seçme, ayrık Fourier dönüşümleri, temel doğrusal cebir, temel istatistiksel işlemler, rastgele simülasyonlar ve daha fazlası yapılabilmektedir. SciPy ise NumPy veri yapıları bünyesinde, sık kullanılan matematiksel rutinleri (diferansiyel denklem çözümü, istatiksel hesaplamalar) ve fiziksel problemlerinin bilgisayar ortamında ifade edilmesine yönelik fonksiyonları barındırmaktadır (URL-3, 2014).

Aşağıda Şekil 4’ de Python programının tercih edilmesinin nedenleri gösterilmiştir. “Thinking in Java” ve “Thinking in C++” kitaplarının yazarı Bruce Eckel’ e göre hiçbir dil Python kadar üretken değildir ve Python hariç diğer diller programcıların işini kolaylaştırma gibi bir amaca yönelmemiştir (Venners, B., 2003).

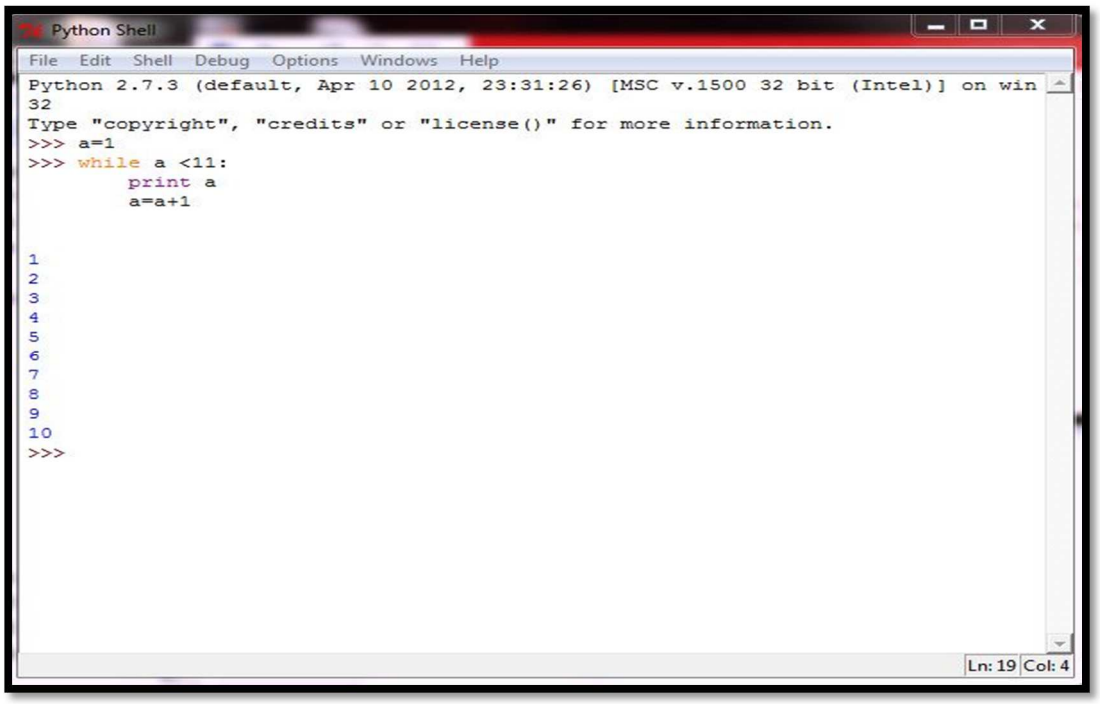
Şekil 4. Python programlama dili tercih nedenleri

1.6.4. Python Programı Temelleri

Python programını kullanabilmek için Python programı temel veri yapılarını bilmek gerekir. Python programı diğer çoğu programlama diline göre kolay öğrenilebilmesine rağmen etkili olarak kullanılabilmesi zaman almaktadır. ArcGIS 10.2 yazılımı Python 2.7 sürümü ile bütünleşik olarak gelmektedir.

1.6.4.1. Python Programı Etkileşimli Kip Ekranı

Başlat/ Tüm Programlar/ ArcGIS/ Python 2.7 dizini içinde IDLE (Integrated DeveLopment Environment) ile Python Shell kısmından etkileşimli olarak kod yazılabilmektedir. Bu kısımda yazılan kodlar ile anlık sonuçlar elde edilir. Bu nedenle Python Shell kodları test etmek için uygun bir ortamdır. Bu ortamda kodların çalışıp çalışmadığı test edilebilir. Bu nedenle programı yeni öğrenmeye başlayacaklar için Python Shell kısmı oldukça uygun bir çalışma ortamıdır. Dikkat edilmesi gereken bir nokta da bu kısımda yazılan kodlar bir betik oluşturmamaktadır. Betik yazmak için Python Shell ekranından dosya/yeni pencere bölümüne girilerek kodların bu ekrana yazılması gerekmektedir. Yazılan betik ".py" uzantılı şekilde kaydedilir.

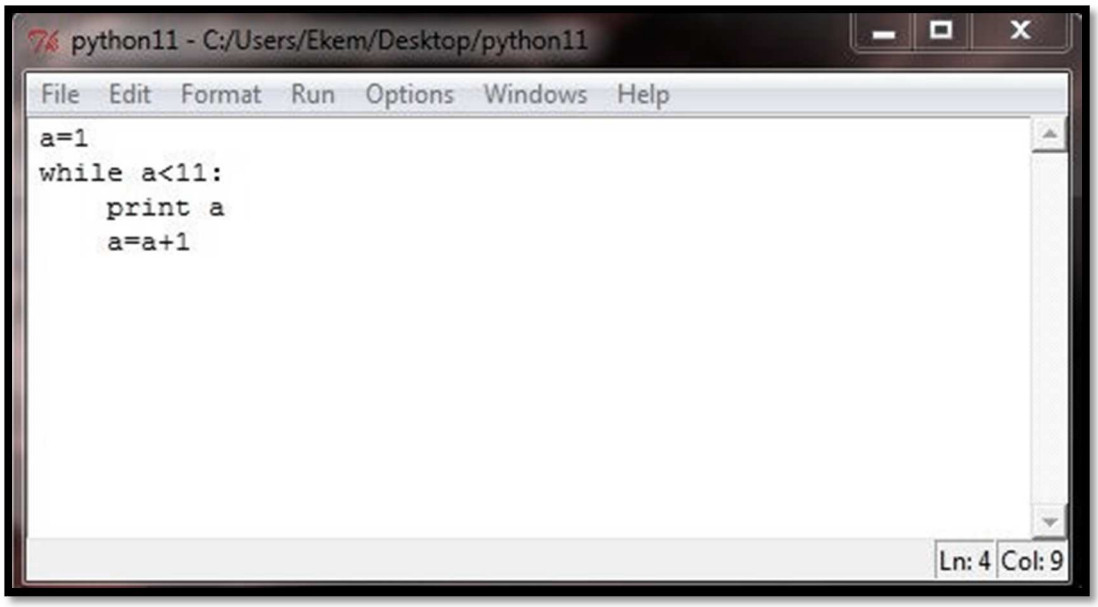


Şekil 5. Python Shell penceresi

Şekil 5'de Python Shell ekranı görülmektedir. Bu ekranda ilk satır, Python program sürümü gibi açıklamaları ikinci satır da lisans bilgilerini göstermektedir. Kod yazım işlemi üç tane büyük işaretinin (>>>) olduğu satırdan hemen sonra başlar.

Python betik yazma penceresi ise Şekil 6' da görüldüğü gibidir. Bu ekrana kodlar yazılıktan sonra Run/Run Module kısmı seçilir ve dosya adı ".py" uzantılı olacak şekilde girilip kayıt edilir. Sonuçlar Python Shell kısmının otomatik olarak açılması ile gösterilir.

Dosya kaydedildikten sonra ".py "uzantılı bir Python dosyasını açmak için dosyanın üzerine farenin sağ tuşu ile tıklanarak Edit with IDLE komutu seçilerek dosya açılır.



Şekil 6. Python betik yazma penceresi

1.6.4.2. Python'da Değişkenler

Python' da değişkenler, değişken ismi ve değeri olarak tanımlanan ve bellekte yer tutan veri yapısıdır. Aşağıda Tablo 1' de değişken ataması gösterilmektedir. Değişken isimleri isteğe bağlı olarak uzun olabilir. Hem harf hem de rakam içerebilir, ancak mutlaka bir harfle başlamaları gerekir. Her ne kadar büyük harf kullanmak geçerli olsa da, geleneksel olarak büyük harfler tercih edilmez. Önemli bir nokta da Python harfe duyarlıdır. Yani büyük harfle yazılan bir değişken ismi ile küçük harf ile yazılan aynı manaya gelen değişken isimleri farklıdır.

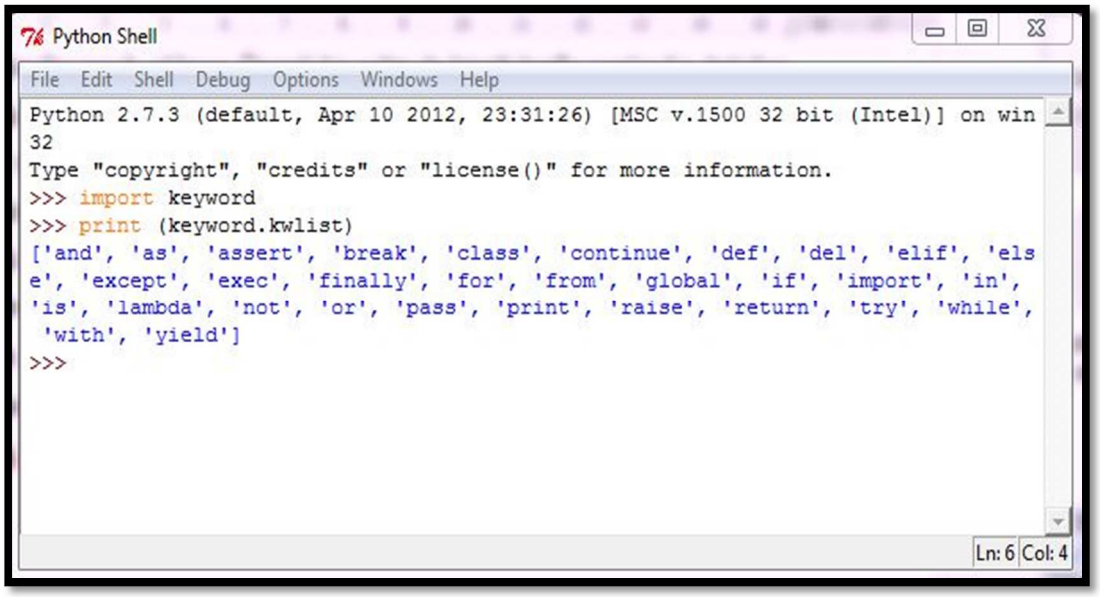
Altçizgi karakteri (_) bir isimde yer alabilir. Genellikle birden fazla harf içeren kelimelerde kullanılmaktadır (Elkner, J). Bunun gibi değişken isimleri yazılırken belli kurallara dikkat etmek gereklidir.

Değişken isimleri tanımlanırken;

- Harfler, rakamlar ve alt çizgi içerebilir fakat rakam veya alt çizgi ile başlayamaz.
- İlk karakter harf olmak zorundadır.

Python’ un kullandığı temel isimler değişken adı olarak atanamaz. Şekil 7’de Python anahtar kelimelerin listesi gösterilmektedir. Bu listedeki isimler değişken ismi olarak programa verilemez. Verildiği durumda program hata verir.

- İsim içinde özel karakterler veya işaretler kullanılamaz (%, ?...)



Şekil 7. Python anahtar kelimelerin listesi

Tablo 1. Pythonda Değişken Ataması

Değişken	
Değişken İsmi	Değişken Değeri
Ada_No	5
Parsel_No	12

Python’da değişkenler dinamik yapıdadır. Dinamik yapıda olması veri tipini tanımaya gerek olmadan değişkenlerin oluşturulabilmesi anlamına gelir.

1.6.4.3. Python’da Karakter Dizileri

Karakter Dizileri (Stringler) metin bilgilerini depolamak için oluşturulmuş karakterlerden oluşan veri tipidir. Karakter Dizileri tek tırnak (‘), çift tırnak (“), ya da üç tırnak (’’’) ile ifade edilir. Type() fonksiyonu ile veri tipi sorgulandığında <class ‘str’> çıktısı alınıyorsa bu karakter dizisi olduğunun göstergesidir. Genel olarak, karakter dizileri üzerinde matematiksel işlemler uygulanamaz. Bu nedenle, “Meyve”+10, “yeni”*"Merhaba", "20"+2 ifadeleri hata üretir. Ancak “+” oparatörü karakter dizilerini birleştirme amacıyla, “*” oparatörü de karakter dizilerini tekrarlama amacıyla kullanılabilir.

```
>>> meyve1="armut"
>>> meyve2="kiraz"
>>> print meyve1+meyve2
armutkiraz
>>> meyve1*5
'armutarmutarmutarmutarmut'
```

1.6.4.4. Python’da Veri Tipleri

1.6.4.4.1. Sayı Veri Tipi

Yaygın kullanılan veri tipi olan sayılar Python programında üç şekilde temsil edilir. Bunlar;

1. Tam Sayılar (integers)
2. Kayan Noktalı Sayılar (floating point numbers veya kısaca floats)
3. Karmaşık Sayılar (complex numbers) (Özgül,2013)

Bir verinin hangi sayı formatında olduğunu type() fonksiyonu ile sorgulandığında int çıktısı üretiyorsa tam sayı, float çıktısı üretiyorsa kayan noktlalı sayıyı ve complex çıktısı üretiyorsa da bu verinin karmaşık sayı olduğunu gösterir.

Tablo 2. Python’da Sayı Formatları

Fonksiyon	Görevi	Örnek	Sonuç
int()	Bir veriyi tam sayıya dönüştürür.	int('2.7')	2
float()	Bir veriyi kayan noktalı sayıya dönüştürür.	float(2)	2.0
complex()	Bir veriyi karmaşık sayıya dönüştürür.	complex(2)	(2+0j)

1.6.4.4.2. Liste Veri Tipi

Liste veri tipi içerisinde aynı anda farklı türden verileri (sayı, karakter, diğer listeler, sözlük…) sıralı bir biçimde barındırabilen kapsayıcı bir veri tipidir. Python’da listelerde, her bir eleman bir indis (index) numarasına sahiptir ve bir listenin başlangıç indisi 0 (sıfır)'dır. Liste tanımlamak için köşeli parantez içerisine ([ ]) öğeler virgül ile ayrılarak yazılır.

```
LBos = []                          # Boş bir liste oluştur
liste = [5,5.37 "meyve"]           # Boş olamayan karma veri tipli liste
LBos.append("armut")               # Listenin sonuna tek bir öğe ekleme
LBos.extend(["kitap", "defter"]) # Listeye birden fazla öğe ekleme
if "5" in liste:                   # Listedeki bir öğeyi sorgulama
liste.remove("meyve")              # Listeden bir eleman siler
```

1.6.4.4.3. Demet(Tuple) Veri Tipi

Python’da bir diğer veri tipi olan demet (Tuple) veri yapısı listeye benzer fakat liste veri tipinde değişiklik yapılabilmesine rağmen demet veri tipinde değişiklik yapılamaz. Yani listelerdeki gibi yeni öğeler eklenip çıkartılamaz. Demet oluşturulurken parantez işareti kullanılır veya hiçbir işaret kullanılmazsa da Python bunu demet olarak algılar.

Örnek demet oluşturma;

```
Demet1 = ('kimya', 'matematik', 1, 2);
Demet2 = (1, 2, 3, 4, 5 );
Demet3 = "a", "b", "c", ;
```

1.6.4.4.4. Sözlük Veri Tipi

Python'da önemli veri tiplerinden birisi de sözlüklerdir (dictionary). Sözlükler, anahtar ve değer çiftlerinin birbirleriyle eşleştirildiği bir veri tipidir. Dolayısıyla sözlükler bu anahtar ve değer çiftleri arasında birebir ilişki kurar. Sözlükler değer olarak her türlü veri tipini kabul eder. Fakat sözlüklere anahtar olarak her veri tipi tanımlanamaz. Bir değerin 'anahtar' olabilmesi için, o öğenin değiştirilemeyen (immutable) bir veri tipi olması gerekir (Özgül, 2013).

```
söz1 = {}                                  # Boş bir sözlük oluşturur
söz2 = {"İstanbul": 34, "Trabzon": 61}     # Anahtar ve değer çiftinden oluşan bir sözlük
```

1.6.4.4.5. Küme Veri Tipi

Python' da veri tiplerinden bir diğeri de kümelerdir (set). Kümeler; liste ve demet veri tiplerinin aksine sözlükler gibi öğelerin sırasız bir şekilde toplandığı bir veri tipidir. Kümeler, matematikte geçerli olan "küme" kavramının sahip olduğu kesişim, birleşim ve fark gibi bütün özellikleri taşır (URL-4, 2014).

```
set1 = set()                        # Bir küme oluşturur.
set1.add("cat")                     # Kümeye bir eleman ekler.
set1.update(["dog", "mouse"])       # Kümeye birden fazla eleman ekler.
for item in set1:                   # Her bir elemana kontrol eder.
  print item                        # kesişim yapar.
set4 = set1 | set2                  # Birleşim yapar.
set5 = set1 - set3                  # Fark alır    (The Python Documantation, 2015).
```

1.6.4.5. Python’da İşleçler

İngilizce’de operatör adı verilen işleçler, sağında ve solunda bulunan değerler arasında bir ilişki kuran işaretlerdir. Bir işlecin sağında ve solunda bulunan değerlere ise işlenen (operand) adı verilir (URL-5, 2014).

Tablo 3. Python’da İşleçler

Aritmetik İşleçler		
+	Toplama	Toplama işlemi yapar
-	Çıkarma	Çıkarma işlemi yapar
*	Çarpma	Çarpma işlemi yapar
/	Bölme	Bölme işlemi yapar
%	Modülüs	Bölme işleminde kalan değeri gösterir
**	Üstel İşlemler	Üstel işlemleri yapar
//	Taban Bölme	Bölme işlemi sonucunu tam sayı verir
Karşılaştırma İşleçleri		
==	Eşittir	İki işlenenin birbirine eşitliğini kontrol eder. Eşitse True değerini verir
!=	Eşit Değildir	İki işlenenin birbirine eşit olmama durumunu kontrol eder. Eşit değilse True değerini verir
<	Küçüktür	Soldaki oparatörün sağdakinden küçük olma durumunu kontrol eder ve küçükse True değerini verir.
>	Büyüktür	Soldaki oparatörün sağdakinden büyük olma durumunu kontrol eder ve büyükse True değerini verir.
<=	Küçük eşittir	Soldaki oparatörün sağdakinden küçük veya eşit olmasını kontrol eder. Bu koşul sağlandığında True değerini verir.
>=	Büyük eşittir	Soldaki oparatörün sağdakinden büyük veya eşit olmasını kontrol eder. Bu koşul sağlandığında True değerini verir

Tablo 3’ ün devamı

Atama İşleçleri		
=		Sağ taraftaki değeri sol taraftaki değişkene atar
+=		Değişken değerine verilen değer eklenir ve sonuç tekrar değişkene atanır.
-=		Değişkenden verilen değer çıkarılır ve sonuç tekrar değişkene atanır.
*=		Değişken değeri ile verilen değer çarpılır ve sonuç tekrar değişkene atanır.
/=		Değişken değeri verilen değere bölünür ve sonuç tekrar değişkene atanır.
%=		Bölme işleminden kalan sayıyı aynı değişkene atar.
**=		Sayının kuvvetini hesaplayıp değişkene atar.
//=		Taban bölme işleminin sonucunu aynı değişkene atar.

1.6.4.6. Sınıflar ve Fonksiyonlar

Sınıflar ve nesneler nesne yönelimli programlamanın temel kavramlarıdır. Python programı nesne tabanlı bir programlama dilidir. Yani nesne yönelimli programlamanın desteklediği özellikleri sağlar. Nesne yönelimli programlama 1960’larda ortaya çıkmasına rağmen 1980’lerin ortasına kadar yeni yazılım üretmede ana programlama paradigması haline gelmedi. Nesne tabanlı programlama hızlı bir şekilde büyüyen ve karmaşıklaşan yazılım sistemlerini kotarma ve bu büyük ve karmaşık sistemleri zaman içerisinde daha kolay değiştirmek için geliştirilmiştir (Elkner, 2008). Sınıflar ve nesneler, nesne yönelimli programlamanın iki ana yönünü oluşturur. Python'un sınıf mekanizması dile minimum yeni söz dizim ile sınıflar ekler. Bu C++ ve Modula-3 sınıf mekanizmalarının bir karışımıdır. Modüllerde de olduğu gibi Python’ da sınıflar tanım ile kullanıcı arasına mutlak bir sınır koymaz. Miras mekanizması birden fazla temel sınıf kullanabilir. Türetilmiş sınıflar temel sınıfın metodlarını geçeriz kılabilir ve kendi sınıf metodlarından temel sınıf metodlarını çağırabilirler. Sınıflar çalışma zamanında oluşturulur ve oluşturulduktan sonra bunlar

üzerinde değişiklik yapılabilir (The Python Documantation). NTP'de en önemli kavram "sınıflar"dır. Sınıflar yapı olarak fonksiyonlara benzer, fonksiyonların görevi, karmaşık işlemleri bir araya toplayarak, bu işlemlerin tek adımda yapılmasını sağlamaktır. Fonksiyonlar kullanılarak, bir veya birkaç adımdan oluşan işlemler tek bir isim altında toplanabilir. Python'daki 'fonksiyon' kavramı başka programlama dillerinde 'rutin' veya 'prosedür' olarak adlandırılır. Gerçekten de fonksiyonlar rutin olarak tekrar edilen görevleri veya prosedürleri tek bir çatı altında toplayan araçlardır. Fonksiyonlar yardımıyla farklı değişkenler ve veri tipleri, tekrar kullanılmak üzere bir yerde toplanır. Sınıflar yardımıyla da farklı fonksiyonlar, veri tipleri ve değişkenler gruplandırılabilir (Özgül, 2013).

Pythonda bir fonksinyon oluşturmak için;

```
def fonksiyon_adi(parametre):
    Yürütülecek kod
```

Fonksiyon tanımlanırken ilk olarak def parçacığını, sonra da fonksiyonun adı yazılır. Fonksiyon adlarında anahtar kelimeler, Türkçe karakterler kullanılmaz ve fonksiyon adları kolaylık olması açısından fonksiyonun kısaca ne işe yaradığını gösterecek şekilde yazılır. Fonksiyon adından sonra parantez ve iki nokta üst üste konularak fonksiyon tanımlanmış olur. Parantez kısmının içine bir veya daha fazla parametre konulabilir veya boş olarak bırakılabilir. Bundan sonra fonksiyonun çalışmasını altına yazılacak kodlar belirler. Fonksiyonun altına yazılacak kodlar def yapısından dört boşluk girintili olacak şekilde yazılır. Yazılan bir fonksiyonu çalıştırabilmek için ise fonksiyonu çağırmak gerekir. Bu işlem fonksiyon adı ve onu takip eden parantez içerisindeki fonksiyona aktarılacak olan değer listesin yazılımı ile olur, değer yoksa boş parantez kullanılır.

Örnek fonksiyon;

```
def ilk_fonksiyon():
    print "fonksiyon"
```

Sınıfların yapısı da fonksiyonlara benzer. Sınıflar veriler ve yapılan işlemler için bir şablon niteliği taşır (Harwani, 2012).

Sınıf tanımları programın içerisinde herhangi bir yerde bulunabilir, ancak genellikle başlangıçta kullanılırlar (import cümlelerinden sonra) (Elkner, 2008). Fonksiyonlar tanımlanırken kullanılan def parçacığı yerine sınıfları tanımlarken class parçacığı kullanılır. Daha sonra Türkçe karakter ve anahtar isim içermeyecek şekilde sınıfa bir ad verilir ve iki nokta üst üste işareti konulur.

Örnek bir sınıf tanımı aşağıdaki gibidir;

class sinif:

<deyim

1.6.4.7. Kontrol Komutları (Döngüler)

Kontrol komutları programlama kısmının önemli parçalarından biridir. Bir bilgisayar programında komutlar normal olarak yazılış sırasına göre çalıştırılır. Örneğin herhangi bir programlama dilinde yazılmış olan

Komut1

Komut2

Komut3

…..

$Komut_{n-1}$

$Komut_{n}$

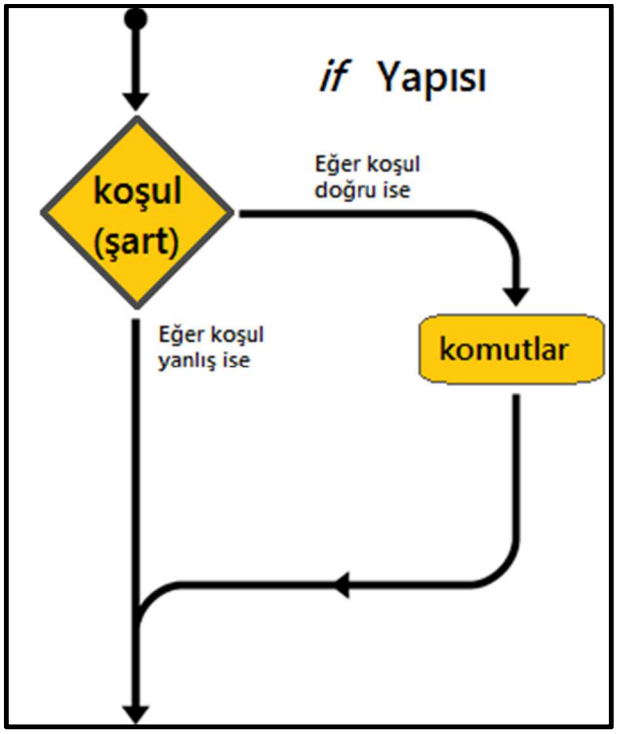
şeklindeki bir programda, önce Komut1 sonra Komut2 ve daha sonra da bu sıra ile komutlar çalıştırılır. Fakat birçok programlama dili bu kadar basit değildir. Programlama dillerinin büyük gücü aynı işi tekrar tekrar yapabilmelerinden ya da değişik parametre değerleri için değişik sonuçlar bulabilmelerinden kaynaklanmaktadır. Kontrol komutları ile yukarıda gösterilen doğrusal akış sırası değiştirilebilir. Örneğin Komut1' den Komut2' ye geçmeden belli bir şartın sağlanması durumunda Komut 3'e geçilebilir. Bu duruma dallanma adı verilir. Kontrol komutları dallanmayı gerçekleştiren komutlardır. Temel olarak iki tipte kontrol komutu vardır (Uysal, 2013).

1.6.4.7.1. İf Deyimi

"if" Türkçe' de "eğer" anlamına gelmektedir. Anlamından da anlaşıldığı gibi bu deyimle bir koşul belirtilir (URL-6, 2014). Bu deyimden sonraki satırda gelen ifade 4 boşluk içerden başlar. Bu satıra yazılan ifade doğru mu koşulu aranacaktır. Koşul yanlış ise program sonraki kodlara geçecektir. Aşağıdaki küçük kod parçasında if yapısının kullanımı gösterilmektedir. Eğer x değeri 0' dan büyükse ekrana x pozitiftir ifadesi yazılacaktır.

```
if x > 0:
    print "x pozitiftir"
```

Şekil 8’ de if yapısı gösterilmektedir. Koşul sağlanması durumunda if bloğu içindeki kod çalıştırılır. Koşul sağlanmadığı durumda ise program devam eder.



Şekil 8. if yapısı (URL-7, 2014)

1.6.4.7.2. Elif Deyimi

Karşılaştırmalarda koşulların artması durumunda elif komutu kullanır. Elif komutu da if komutu gibi koşul bildirir ve if’ den sonra yazılınca if 'in oluşturduğu bloğun haricinde tutulmalıdır. İf ile arasındaki fark ise elif ile belirtilen birden fazla koşuldan istenileni veren ilk koşul sağlandığında program dururken, if ile belirtilen birden fazla koşulda bu böyle olmaz, program geri kalan koşulları da inceler ve öyle durur.

Örnek verilecek olursa;

```
# !/usr/bin/env python
# -*- coding: utf-8 -*-
```

```
sayi = input ("Bir sayı giriniz:")
if sayi < 0:
   print ("Sayı 0'dan küçüktür.")
elif sayi == 10:
   print ("Sayı 10'dur.")
elif sayi > 5:
   print ("Sayı 5'ten büyüktür.")
elif sayi > 10:
   print ("Sayı 10'dan büyüktür.")
```

Bu kod çalıştırıldığında program koşulu sağlayan ilk ifadeyi bulduktan sonra blok kısmındaki işlemleri gerçekleştirecektir ve geriye kalan koşulları dikkate almaz. Örneğin kullanıcı girdi verisi olarak 10 girmişse çıktı şu şekilde olacaktır:

Sayı 10' dur.

Aynı kod aşağıdaki şekilde sadece if kullanarak yazılsaydı:

```
# !/usr/bin/env python
# -*- coding: utf-8 -*-
sayi = input ("Bir sayı giriniz:")
if sayi < 0:
   print ("Sayı 0'dan küçüktür.")
if sayi == 10:
   print ("Sayı 10'dur.")
if sayi > 5:
   print ("Sayı 5'ten büyüktür.")
if sayi > 10:
   print ("Sayı 10'dan büyüktür.")
```

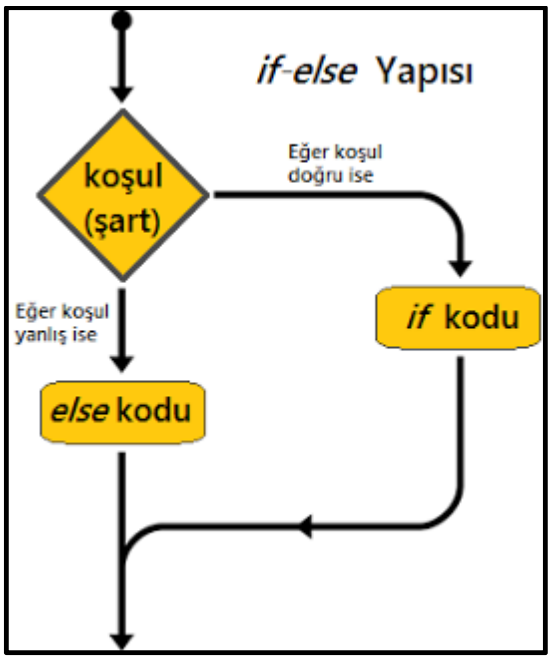
Bu durumda program bütün koşulları tek tek sorgulayacaktır. Kullanıcının yine 10 girdiğini varsayılırsa bu durumda da ekran çıktısı aşağıdaki şekilde olur.

Sayı 10'dur.

Sayı 5'ten büyüktür.

1.6.4.7.3. Else Deyimi

Else deyimi, if ve elif deyimlerindeki koşulların sağlanmadığı bütün durumları kapsar. Şekil 9' da else deyiminin akış diyagramı gösterilmektedir.



Şekil 9. else yapısı (URL-8)

else deyimi isteğe bağlı bir ifadedir ve if-elif bloklarıyla birlikte sadece bir kez kullanılır. Python dilinde else deyiminin yazılışı aşağıdaki gibidir;

```
if koşul:
    ifadeler
else:
    ifadeler
```

Yukarıdaki kullanım şekline sahip "if" deyimi istenen koşula göre istenen kod parçasının çalıştırılmasını sağlar.

if karşılaştırma:

```
  doğruysa çalıştırılacak blok
elif karşılaştırma:
   yürütülecek blok
else:
 yürütülecek blok
```

1.6.4.7.4. While Döngüsü

While döngüsü, programın tamamlanana kadar programın kapanmasına engel olur ve kod dizisinin döngünün başındaki koşul sağlanana kadar dönmesini sağlar.

Örnek verilecek olursa,

```
Sayi=1
while Sayi<100:
        Sayi=Sayi+1
        print Sayi
```

Bu küçük kod parçacığı 1’ den 100’ e kadar olan sayıları ekrana yazdırır.

1.6.4.7.5. For Döngüsü

For döngüsünün görevi bir dizi içindeki öğeleri denetlemektir.
Örnek verilecek olursa,

```
for harf in “Damla-Umut”:
   print harf
```

Bu programda öncelikle bir karakter dizisine harf adını bütün öğelerine ad olarak atanmıştır. Daha sonra print komutu ile bu harfler ekrana tek tek yazdırılmış olur.

1.7. Grafiksel Kullanıcı Arayüzü (GKA)

Grafiksel Kullanıcı Arayüzü (ing. Graphical User Interface; GUI, GKA), bilgisayarlarda işletilen komutlar ve bunların çıktıları yerine; simgeler, pencereler, düğmeler ve panellerin oluşturduğu bir sistemdir. Grafiksel kullanıcı arayüzü, bilgisayar kullanıcılarının komut satırı kodlarını ezberlemeden fare, klavye gibi araçlar sayesinde bilgisayarları kontrol etmelerini sağlamıştır. Günümüzdeki programların birçoğu GKA ile birlikte gelse de, birçok bilgisayar kullanıcısı (özellikle programcılar) daha hızlı olduğu gerekçesiyle komut satırını GKA'larla birlikte kullanmaya devam etmektedirler (URL-9, 2015). Arayüz tasarım kavramı, her çeşit ortamdaki teknolojik ürünlerin insanlar tarafından kullanılmasıyla ilgilidir. Arayüz tasarımında amaç, bu ürünleri kullanan kişilere başarılı deneyimler yaşatmak ve memnuniyetlerini sağlamaktır (Boling ve Sousa,1993). Grafiksel kullanıcı arayüzü (GUI) kullanarak program geliştirmek oldukça yaygındır. Pencere, buton, listeleme kutuları, menüler modern programların ayrılmaz bileşenlerini oluşturmaktadır (Uysal, 2012).

Günümüzde kullanıcı arayüz tasarımı için kullanılan en popüler tasarım kılavuzlarından birisi Nielsen tarafından verilmektedir (Nielsen's Ten Usability Heuristics).

Nielsen'e göre kullanıcı arayüzlerinde şunlara dikkat edilmelidir.

- Basit ve doğal diyalog kullanımı
- Hafıza yükünün en aza indirilmesi
- Tutarlılık
- Geribildirim sağlanması
- Açıkça gösterilmiş çıkış ve işlem sonlandırma
- Kısayolların önerilmesi
- Uygun hata mesajlarının tasarlanması
- Hataların engellenmesi
- Yardım ve belgeleme sağlanması

Genel olarak programlama dilleri, yazılan programla kullanıcı arasındaki iletişimi iki farklı yoldan sağlar:

Komut satırı arayüzü (command-line interface)

Grafik kullanıcı arayüzü (grapfical user interface)

Bu durum Python programlama dili için de geçerlidir. Python programlama dilinin sadece kendi imkanlarını kullanarak yalnızca komut satırı üzerinden çalışan programlar yazılabilir. Fakat Python'la uyumlu harici kütüphaneleri kullanarak grafik arayüze sahip programlar yazma imkanı da mevcuttur.(Özgül, 2013)

Python'da grafik arayüz tasarlayabilmek için pek çok ek kütüphane seçeneği mevcuttur. Bu seçeneklerin en önemlilerini şöyle sıralanabilir:

Tkinter

PyGObject + GTK3

PyQt

FOX

OpenGL

Bunların dışında başka seçenekler de bulunur. Qt kütüphanesi; pek çok farklı işletim sistemi üzerinde çalışabilen, oldukça profesyonel ve son derece modern bir arayüz geliştirme kütüphanesidir. Bu arayüz kütüphanesi Nokia şirketi tarafından C++ adlı programlama dili ile geliştirilmektedir. Qt, C++ adlı programlama dili ile geliştirildiğinden C++ ile birlikte kullanılabilir. Qt adlı bu arayüz kütüphanesini Python ile birlikte de kullanabilmek için, PyQt adlı bir ara katmandan yararlanmak gereklidir. Herhangi bir dille yazılmış bir kütüphanenin başka bir dille birlikte de kullanılabilmesini sağlayan bu tür ara katmanlara 'bağlayıcı' (binding) adı verilmektedir. PyQt4; Python ile Qt adlı arayüz kütüphanesi arasında bağ kuran PyQt adlı bağlayıcının 4 numaralı sürümüdür.

Bu çalışmada Arayüz hazırlamada PyQt4 kullanılmıştır. PyQt4 Standart GKA araçlarından daha fazlasını sunar. Unicode, düzenli ifadeler, SQL veritabanları, SVG, OpenGL, XML, işlevsel bir web tarayıcısı, yardım sistemi ve de zengin GKA araçlarını sunar. Qt sınıfları güvenli ve kolay bir şekilde yeniden kullanılabilir yazılım bileşenleri oluşturur. Nesneler arasındaki iletişim için bir sinyal / slot mekanizması kullanır. Qt' nin en önemli avantajlarından birisi de, bir grafik kullanıcı arayüzü tasarımcısı (Qt tasarımcısı) içermesidir. Bu sayade PyQt, Qt tasarımcısı ile Python kodu üretebilir. Qt tasarımcısı Sadece ekran (form) tasarlamakta kullanılmaz, herhangi bir özel görsel öğe (widget) geliştiriminde kullanılabilir. Tasarımlarda kullanılan resim ve ikon dosyaları Qt Resource adı verilen dosyalarda tanımlanabilir ve Qt'nin temelini oluşturan sinyal ve slot bağlantıları yapılabilir. PyQt, Python' un ve Qt' nin bütün avantajlarını birleştirir.

1.7.1. Grafiksel Kullanıcı Arayüzü İçin PyQt4

GKA (Grafiksel Kullanıcı Arayüzü, GUI) programları, bir pencere üzerindeki çeşitli elemanlardan oluşur ve kullanıcının tepkisine göre program davranışı belirlenir. Örneğin kullanıcının fareyi hareket ettirmesi, farenin tuşlarına bastırması, klavye tuşlarına bastırma veya diğer çevre elemanlardan bir uyarı gelmesi durumuna göre program ne yapacağına karar verir. Python' da grafik arayüz tasarımında Qt' nin seçilmesinin nedeni hem tam teşekküllü olması hem de bizzat geliştirici firma tarafından Python eklentisinin desteklenmesidir.

1.7.1.1. PyQt4 Temel İşlemler

Arayüz tasarlarken vazgeçilmez öğelerin başında pencere oluşturma yer almaktadır. Daha sonra pencereye menüler, butonlar ve araç çubukları eklenir. PyQt4 ile geliştirme yaparken temel olarak iki modülü mutlaka içe aktarmak gerekmektedir. Bunlar PyQt4 modülü içindeki QtGui sınıfı ve gömülü modüllerden biri olan sys modülüdür. Bu modüllerin sunduğu işlevlerden program yazma sırasında yararlanılmaktadır. Bu modüller içe aktarıldıktan sonra zorunlu olarak bir QApplication nesnesi oluşturulur. QApplication nesnesi, programın çalıştığı masaüstü ortamı ile yazılan program arasındaki ilişkiyi düzenlediği için, grafik arayüze dair başka herhangi bir kod yazmadan önce bu satırı yazmış olmak gerekmektedir. QApplication() nesnesine sys.argv parametresi verilir. Bu sayade program çalıştırılırken komut satırında verilen parametrelerin tutulması sağlanmış olur. QApplication(sys.argv) nesnesi yazılan programın başlangıcından sonuna kadar olan bütün süreci takip ederek programın düzgün bir şekilde çalışıp sona ermesini sağlar. Bu kod yazılmadığı durumda program çalışmaz ve hata verir. Daha sonra pencere oluşturmak için QtGui modülünün QWidget() sınıfı QtGui.QWidget() şeklinde kullanılır. Oluşturulan pencerenin ekranda görülebilmesi için show() adlı bir metottan yararlanır. Aksi taktirde oluşturulan pencere ekranda görülmez. Daha sonra QApplication nesnesi exec_() adlı metod ile açılan pencerenin hemen kapanmaması sağlanmış olur. Yani bu komut pencerenin kullanıcının verdiği komut ile kapanması sağlamaktadır. Bu satır uygulama

döngüsünü oluştur. Aksi durumda pencere hemen açılıp kapanacaktır. Aşağıda kod boş bir pencere tasarlamak için yeterlidir. Şekil 10’ da açılan boş pencere görülmektedir. Bu pencerede olması gereken simge durumuna küçültme, tam ekran yapma ve çıkış tuşları otamatik olarak gelmektedir.

```
from PyQt4.QtGui import *
import sys
temelfonk = QApplication(sys.argv)
pencere = QWidget()
pencere.show()
temelfonk.exec_()
```

Şekil 10. PyQt ile hazırlanan boş bir pencere.

Yukarıda açıklanan kodlar ile, PyQt4 ile prosedür tabanlı (yordamsal) programlama ilkelerine uygun bir geliştirme süreci takip edilebilir. Fakat hem kullanışlılık açısından, açısından nesneye yönelik programlama tarzı daha çok tercih edilmektedir. Yukarıdaki kodlar ile oluşturulan boş bir pencere nesneye yönelik kodlama ile oluşturulması için aşağıdaki şekilde kodlanmalıdır.

```
from PyQt4.QtGui import *
import sys
class YeniPencere(QWidget):
    def __init__(self):
        QWidget.__init__(self)
        self.show()

uygulama = QApplication(sys. argv)
pencere = YeniPencere()
uygulama. exec_()
```

Buradaki class kısmı ile bir sınıf oluşturulmaktadır. Python' da bir programın ilk kez çalıştırıldığı anda işlemesi istenilen kısımlar __init__ fonksiyonu içine yazılır. Self kısmı oluşturulan sınıfın örneğini temsil eder.

Pencerenin başlığını değiştirmek için setWindowTitle() metodu kullanılır. self. setWindowTitle("Pencere Başlığı") şeklinde olması istenen başlık bu metodun parantezleri arasına yazılır. Pencere simgesini değiştirmek içinse QIcon() ve setWindowIcon() metodları kullanılır. QIcon() metodu QIcon('iconresim.jpg) şeklinde kullanılarak simge dosyasını tanımlamamızı sağlamaktadır. Ardından setWindowIcon() metodu kullanılarak istenilen simge eklenmiş olur.

```
penceresimgesi = QIcon("iconresim. jpg")
self. setWindowIcon(penceresimgesi)
```

Pencere boyutlarını ayarlamak için ise resize() adlı metod resize(100,100) şeklinde kullanılır. Parantez içindeki ilk parametre pencerenin enini, ikinci parametre ise boyunu göstermektedir. Aksi durumda pencere oluşturulduğunda ön tanımlı bir pencere açılmış olacaktır. Pencerenin ekrandaki konumunu ayarlamak içinse move() metodu kullanılır. Örneğin, move(100,200) şeklinde kullanılır. Burada 100 değeri pencerenin soldan sağa konumunu verirken 200 değeri de yukardan aşağı konumunu göstermektedir.

Pencere oluşturulduktan sonra grafik arayüz tasarımı için düğmeler, menüler, kutucular, etiketler ve buna benzer başka araçların oluşturulması gerekmektedir. Bir grafik arayüz penceresinde bulunan bütün bu araçlara 'pencere aracı' (*widget*) adı verilir. En temel pencere araçlarından biri olan QLabel() adlı pencere aracı ile program penceresi üzerinden kullanıcılara program ile ilgili mesajlar gösterilebilir. Bu etiketin içerisine yazılacak metin ekranda gösterilir. QPushButton() metodu ise buton oluşturmayı sağlar. Bu metot üzerine bastırılabilen bir buton oluşturulmasını sağlar. QSpinBox() sayısal değerlerin kullanıcı hatasına imkan vermeden girilmesini sağlayan bir metotdur. QComboBox ise kullanıcıya verilen seçeneklerden birisini seçmesinin sağlayabileceği bir açılır liste oluştur. QCheckButton ()' da işaretleme düğmesini oluşturan bir metottur.

Bunların dışında program kullanıcılarının penceredeki herhangi bir nesne üzerinde fare ile gezindiklerinde, farenin bir tuşuna tıkladıklarında veya klavyeden bir tuşa bastıklarında sinyaller oluşturulur. Örnek olarak QPushButton() ile oluşturulmuş bir butonun üzerine basıldığında buradan bir pressed() sinyali yayımlanır. Sinyaller bir işleve bağlanmadığı sürece iş yapılmaz dolayısıyla sinyallerin bir işleve yönlendirilmesiyle hazırlanan grafik arayüzler iş yapmış olacaktır. Bir butona tıklandığında pencerenin kapanması işlevi buna örnektir (Beşer, 2013).

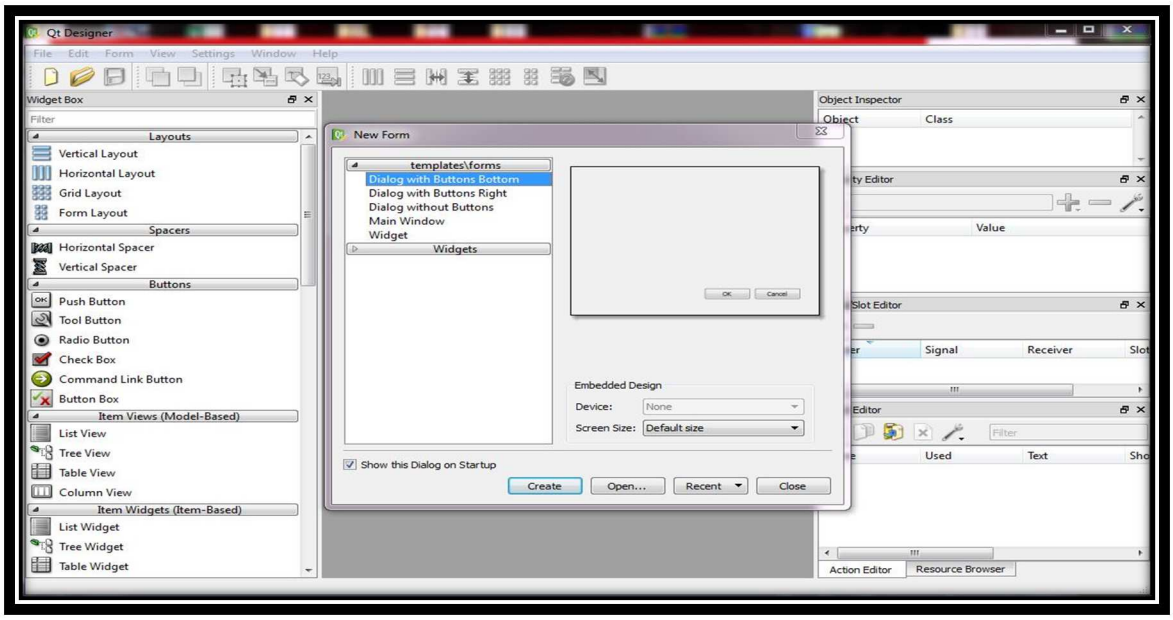
1.7.2. Qt Tasarımcısı (Designer)

Qt, birden çok platformu destekleyen bir grafiksel kullanıcı arayüzü geliştirme araç takımıdır. Genellikle GKA programları geliştirmek için kullanılsa da gelişmiş kütüphanesi GKA bileşenlerinin dışında birçok araç içermektedir. Qt, C++ kullansa da, Python, Ruby, PHP, Perl, Pascal, C# ve Java gibi programlama dillerine olan bağlantıları sayesinde bu dillerle de rahatlıkla kullanılabilir (URL-10, 2015). Qt ile birlikte arayüz tasarlamak için kullanılabilecek Designer programı, Qt'nin gelişmiş ve eksiksiz belgelerine bakabilmek için bir belge gezgini olan Assistant programı ve Qt uygulamalarını farklı dillere çevirebilmek için Linguist programları otomatik olarak gelmektedir.

Qt Designer, grafik arayüz hazırlamada kullanılan bir pencere tasarım aracıdır. Pencere tasarım araçları ile pencere, form ve diyaloglar kod yazma ile uğraşılmadan tasarım aracının üzerinde bulunan parçacıkların sürükle bırak yöntemi ile kolayca tasarlanmasını sağlar ve bu parçacıkların özelliklerinin değiştirilebilmesine imkan verir. Kod yazılarak yapılan bu işler bir robot tarafından otomatik olarak yapılır ve hazırlanan

arayüzün kodlarını verir. Python programlama dili ön tanımlı olarak bu tarz bir bütünleşik pencere tasarım aracı sunmamaktadır. Fakat çok kapsamlı bir GKA aracı olan Qt ile birlikte Qt Designer(Tasarımcı) otomatik olarak gelmektedir. Yani Python ile kullanılabilen PyQt sürümü Qt designer tasarımcısı içermektedir. Qt ile tüm diller kullanılabilir bu nedenle Qt herhangi bir dile özgün kod üretmez. Bunun yerine Qt de oluşturulan pencereler XML biçiminde kaydedilir. Üretilen XML dosyası uygun bir araçla istenilen dile dönüştürülür. Ayrıca Python ya da bir başka programlama dilindeki sürüm değişikliğinden etkilenmez.

Qt tasarımcısı, Şekil 11' de görüldüğü gibi üç ana panelden oluşmaktadır. Sol taraftaki kısım arayüzde kullanılacak araçların bulunduğu kısımdır. Orta kısım da pencerelerin tasarlanacağı çalışma alanını oluşturur. Son olarak sağ taraf da pencere araçlarının özelliklerinin ayarlanabildiği, sinyallerin oluşturulabildiği bir panel yer almaktadır. Qt tasarımcısı başlatıldığında ilk olarak açılan diyalogdan ne tarz bir pencere oluşturulacağı sorulmaktadır.



Şekil 11. Qt tasarımcı ekranı

1.7.3. Qt Tasarımcısında Hazırlanan Arayüzü Python Koduna Çevirme

Qt tasarımcısında sürükle bırak yöntemiyle arayüz tasarlandığında oluşan dosya ".ui" formatında oluşur. Bu dosyaya Python programlama dili ile kod ekliyebilmek için bu ".ui "

dosyasının python kodlarına çevrilmesi gerekmektedir. Örneğin x.ui şeklindeki bir Qt Tasarımcısı' nda oluşturuşmuş bir dosyayı Python dosyasına çevirmek için öncelikle yönetici penceresi açılır (command window) sonra Qt Tasarımcısı ile hazırlanan arayüzün olduğu dizine gidilir ve aşağıdaki kod yazılır.

```
C:\Python32\Lib\site-packages\PyQt4\pyuic4.bat x.ui-o x.py
```

Bu kod ile x isminde kaydedilen ui dosyasından yine x adında bir Python dosyası oluşturulmuş olur.

2. YAPILAN ÇALIŞMALAR

2.1. Kullanılan Yazılımlar ve Modüller

Bu çalışmada kodların yazımında Python 2.7 programlama dili kullanılmıştır. Python programlama dili, resmi internet sitesi olan https://www.python.org/downloads/ adresinden çalışmaların yapılacağı Windows 7 64 bit işletim sistemli bilgisayara uyumlu olan sürümü ücretsiz olarak indirilip kurulmuştur. Daha sonra Python programlama dili ile ilgili ek modüller indirilip kurulmuştur. Bu modüller Python geliştiricilerinin hazırladığı modüller ve Python ile ilgilenen programcıların hazırlayıp internet ortamında paylaştığı modülleri içermektedir. Kodların yazımında kullanılan bazı modüller numpy, os, sys, pywt, pillow, GDAL, imread, scipy, pywt, PyQt4 modülleridir. Arayüz hazırlama aşaması için ise Qt kütüphanesinin Python ile uyumlu hale getirilmiş olan PyQt4 kütüphanesi kullanılmıştır. Bu tez çalışmasında ESRI firması tarafından geliştirilen ArcGIS yazılımının 10.2 sürümü kullanılmıştır.

2.2. Gerçekleştirilen İşlemler

Python programlama dili ile görüntü kaynaştırma yöntemlerinden IHS, Brovey ve dalgacık dönüşümü yöntemleri kodlanmıştır. Görüntü kaynaştırmanın doğruluğunun belirlenebilmesi için ERGAS, SSIM, SAM ve CC metrikleri kodlanmıştır. PyQt4 ile görüntü kaynaştırma ile ilgili bir arayüz oluşturulmuştur. Görüntü kaynaştırma için kalite değerlendirme metriklerini içeren başka bir arayüz hazırlanmıştır. Sınıflandırma yöntemlerinden ISODATA ve K-means yöntemleri için bir arayüz oluşturulmuştur. Oluşturulan bu üç arayüz için ArcGIS yazılımının desteklediği Python add-in ile bir eklenti oluşturulup bu eklenti sayesinde bu arayüzler ArcGIS yazılımına eklenmiştir.

2.2.1. Görüntü Kaynaştırma Yöntemlerinin Kodlanması

2.2.1.1. Brovey Yönteminin Kodlanması

Brovey yönteminde görüntüdeki bantların toplamınından “intensity” bileşeni hesaplanır ve bu bileşen pankromatik görüntüye çözünürlüğü hariç eşit sayılır. Her bir bant, renkli görüntüdeki bantların toplamından oluşan intensity bileşenine bölünür ve pankromatik görüntü ile çarpılır.

R_cikti= (R_girdi*pan)/(R_girdi+Y_girdi+B_girdi)
G_cikti= (G_girdi*pan)/(R_girdi+Y_girdi+B_girdi)
B_cikti= (B_girdi*pan)/(R_girdi+Y_girdi+B_girdi)

Sonuç kaynaşmış görüntü R_cikti, G_cikti ve B_cikti bantlarının birleştirilmesinden oluşur.

Brovey yöntemi kodlanırken GDAL (Geospatial Data Abstraction Library) ve numpy modülleri kullanılmıştır. Görüntülerin açılmasında ve kaydedilmesinde GDAL modülü kullanılırken, piksel düzeyinde işlemlerde numpy modülü kullanılmıştır. Tanımlanan fuse brovey adında bir fonksiyon ile kullanıcılardan girdi parametreleri olarak cokbantli_image, pan_image, sondosyaad ve dosya tipi parametrelerinin verildiği durumda Brovey görüntü kaynaştırma yöntemi uygulanmış olmaktadır. Brovey yönteminde her bir bant, renkli görüntüdeki bantların toplamından oluşan intensity bileşenine bölündüğünden tüm bantların sıfır olma durumunda yani siyah bir yüzeyde sayı bölü sıfır belirsizliği oluşacağından bantların toplamına gri değerleri çok etkilemeyecek olan bir sayı (0,01) eklenmiştir.

```
def fusebrovey(multiad, panad, sondosyaad, dosyatipi='GTiff'):
    pan = LoadFile(panad)
    multi = LoadFile(multiad)
    sinif = multi.dtype
    multi = multi.astype('float')
    pan = pan.astype('float')
    bantsay = multi.shape[0]
    sizp = pan.shape
```

```
    sonsiz = (bantsay, sizp[0], sizp[1])
    toplam=numpy.sum(multi, axis=0)
    toplam = cokbantboyut(toplam,sizp)+0.01
    oran = pan/toplam
    yenires = numpy.zeros(sonsiz)
    for i in range(bantsay):
        yenires[i] = cokbantboyut(multi[i],sizp)*oran
   return sonislemler(yenires, sinif, sondosyaad, dosyatipi)
```

2.2.1.2. IHS (Intensity-Hue-Saturation) Yönteminin Kodlanması

IHS görüntü kaynaştırma yönteminde fuseihse adında bir fonksiyon oluşturulup kullanıcılardan girdi olarak da multiad, panad, sondosyaada, parametrelerini istenmektedir. Burada hızlı IHS yöntemi kullanılmıştır. İlk olarak multispektral görüntü oluşturulan cokbantboyut fonksiyonu ile yeniden örnekleme yapılır. Yeniden örneklenmiş multispektral görüntünün bantlarının ortalaması hesaplanır ve "ort" değişkenine atanır. "fark" değişkenine pan ile ort değişkenin farkı atanır. Bu fark da her bir multispektral banda eklenir ve kaynaşmış görüntünün son bantları ortaya çıkartılır.

```
def fuseihs(multiad, panad, sondosyaad, dosyatipi='GTiff'):
    pan = LoadFile(panad).astype(float)
    multi = LoadFile(multiad)
    sizp = pan.shape
    sinif = multi.dtype
    boyut = multi.shape
    bantsay = boyut[0]
    multi = cokbantboyut(multi.astype(float), sizp)
    ort = multi.sum(axis=0)
    ort /= float(bantsay)
    fark = pan-ort
    son = numpy.zeros([bantsay, sizp[0], sizp[1]])
    for i in range(bantsay):
```

```
    son[i] = multi[i]+fark
return sonislemler(son, sinif, sondosyaad, dosyatipi)
```

2.2.1.3. Dalgıcık Tabanlı Yöntemin Kodlanması

Dalgacık tabanlı yöntem için Python programla dili modüllerinden WaveletPacket2D modülü kullanılmıştır. Burada “ sh ” değişkenine çokbantlı görüntünün boyutları yani bant sayısı, sütun ve satır sayısı atanmıştır. Aynı şekilde “sizp” değişkenine pankromatik görüntünün boyutları atanmıştır. Daha sonra “son” değişkenine çokbantlı görüntünün bantsayı ve pankromatik görüntünün satır ve sutun sayıları kadar bir 0 matrisi oluşturulup atanmıştır. Sonra WaveletPacket2D fonksiyonu kullanılmıştır.

```
def fusewave(multiad, panad, sondosyaad, dosyatipi='GTiff'):
from pywt import WaveletPacket2D
pan = LoadFile(panad).astype(float)
multi = LoadFile(multiad)
sh = multi.shape
sizp = pan.shape
bantsay = sh[0]
son = numpy.zeros((bantsay, sizp[0], sizp[1]))
wp= WaveletPacket2D(data=pan, wavelet='db1', mode='sym')
for n in range(bantsay):
   wp['a'] = cokbantboyut (multi[n]), wp['a'].data.shape)
   wp.reconstruct(update=True)
   son[n] = wp.data
return son
```

2.2.2. Kalite Değerlendirme Metrikleri'nin Kodlanması

Kaynaştırma sonucu oluşan görüntülerde kaynaştırma sonuçlarının doğruluğunun değerlendirilmesi için çeşitli metrikler kullanılmaktadır. Bu çalışmada ERGAS, SSIM ve CC metriklerine yer verilmiştir.

2.2.2.1. ERGAS ((Erreur Relative Globale Adimensionnelle de Synthése)

ERGAS metriği karesel ortalama hatadan ve kaynaştırılacak olan görüntülerin konumsal çözünürlüklerinden faydalanır. Keskinleştirilmiş görüntünün genel kalitesi hakkında bir fikir vermektedir. Bu ölçüt keskinleştirilmiş görüntünün bozulma miktarı ile alakalı olduğu için mümkün olduğunca küçük olması beklenir (Alparone vd., 2008). Çok bantlı görüntülerin birleştirilmesinde kullanılan yöntemlerin spektral kalite metriğini ölçmede kullanılır. (Kösesoy, 2012)

ERGAS metriğinin formülü aşağıdaki gibidir.

$$E = 100\frac{h}{l}\sqrt{\frac{1}{K}\sum_{k=1}^{K}\left(\frac{RMSE(k)}{\mu(k)}\right)^2} \quad (1)$$

Bu formüldeki h ve l sırasıyla pankromatik görüntü ve çok bantlı görüntünün konumsal çözünürlüklerini, $\mu(k)$ k bandının ortalamasını, K toplam bant sayısını, $RMSE(k)$ ise çok bantlı görüntünün k' inci bandı ile kaynaştırılmış görüntünün k' inci bandı arasındaki karesel ortalama hatayı göstermektedir (Gungor, 2008).

Aşağıdaki kodda imulti çok bantlı görüntüyü, ifused da görüntü kaynaştırma yöntemleriyle kaynaştırılmış görüntüyü temsil etmektedir. Burada "msh" değişkenine çok bantlı görüntünün bant sayısı, satır sayısı ve sütun sayısı atanmaktadır. Aynı şekilde "fsh" değişkeninde kaynaşmış görüntünün bant sayısı, satır sayısı ve sutun sayısı atanır. "fshl" ve "mshl", değişkenleri de bant sayısının olmadığı hali, yani sadece satır ve sütun sayısını içermektedir. "downs" değişkeni false olarak ayarlanırsa çok bantlı görüntüde yeniden örnekleme yapılır, true olduğu durumda da kaynaştırılmış görüntü de yeniden örnekleme yapılmaktadır. Her iki durumda da boyutlar eşitlenmiş olmaktadır. Daha sonra ERGAS metriği formülü uygulanmaktadır. Her bant için kaynaşmış görüntü bantları ile çok bantlı

görüntü bantları arası karesel ortalama hata (RMSE) hesaplanıp, çok bantlı görüntünün bandının ortalamasına bölünür. Burada hl=1/4 kaynaştırılmış görüntünün çözünürlüğü 1m, çok bantlı görüntünün çözünürlüğünün ise 4 m olduğu durumu temsil etmektedir.

```
def ergas(imulti,ifused,downs=True,hl=None):
    msh=imulti.shape
    fsh=ifused.shape
    fshl=list(fsh)
    mshl=list(msh)
    fshl.pop(0)
    mshl.pop(0)

    if hl is None:
        hl=float(mshl[0])/float(fshl[0])

    if hl==1:
        hl=1/4

    if hl>1:
        hl=1/hl


    bantsay=min(fsh[0],msh[0])
    if downs:
        ifused=cokbantboyut(floati(ifused),tuple(mshl))
    else:
        imulti=cokbantboyut(floati(imulti),tuple(fshl))

    top=float(0)
    for n in range(bantsay):
        ff=ifused[n].reshape(-1)
        mm=imulti[n].reshape(-1)
```

```
    rmoo=rmse(ff,mm)
    omoo=mm.mean()
    top=(rmoo/omoo)**2+top
top=numpy. sqrt(top/bantsay)
return top*100*hl
```

2.2.2.2. SSIM (Structural Similarity Index Method)

Konumsal bir metrik olan SSIM metriği görüntüler arası konumsal detayların aktarımındaki kaliteyi ölçer. Sonuçlar -1 ile +1 arasında değer alır. Sonucun +1'e yaklaşması benzerliğin arttığı anlamına gelir (Yıldırım ve Güngör, 2012).

SSIM yöntemi için Python geliştiricileri tarafından hazırlanan pyssim modülü kullanılmıştır. Bu modül SSIM yöntemini içermektedir. Orijinal kodlar Antoine Vacavant tarafından yazılmıştır ve Christopher Godfrey ve Jeff Terrace tarafından modifikasyonlar gerçekleştirilmiştir. Bu modül Python 2.7 sürümü ile kullanılabilmektedir. Kaynak kod Ek 1' de verilmektedir.

2.2.2.3. CC (Collerattion Coefficent)

Korelasyon Katsayısı Görüntülerin her bandı arasındaki korelasyon kolayca hesaplanabilir ve genel olarak görüntü için korelasyon değeri istenir ise bantların korelasyonun ortalaması kullanılabilir.

$$CC(Bi,Bj) = \frac{\sum_{m=1}^{M}\sum_{n=1}^{N}(Bi_{(m,n)} - \bar{B}i)(Bj_{(m,n)} - \bar{B}j)}{\sqrt{\sum_{m=1}^{M}\sum_{n=1}^{N}(Bi_{(m,n)} - \bar{B}i)^2 \sum_{m=1}^{M}\sum_{n=1}^{N}(Bj_{(m,n)} - \bar{B}j)^2}} \quad (2)$$

Yukarıdaki formülde B_i ve B_j karşılaştırılacak bantları $\bar{B}i$ ve $\bar{B}j$ bantların ortalama gri değerlerini, *M* ve *N* satır ve sütun sayısını temsil etmektedir. Korelasyon katsayısının değeri [-1,+1] aralığındadır. Bu değer +1'e yaklaştıkça bantlar istatistiki olarak birbirine çok yakın -1'e yaklaştıkça bantlar birbirine çok zıt demektir (Özendi, 2014).

Kolerasyon katsayısı için kodlanan korell fonksiyonu kullanılmıştır.

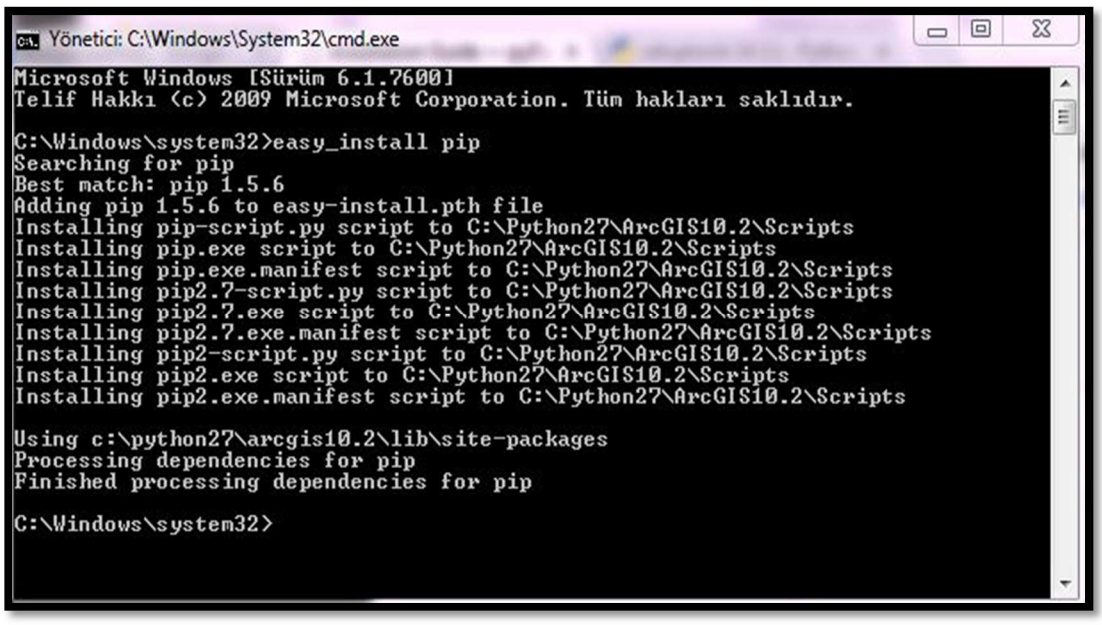
```
def cc(imulti,ifused=None):
    if ifused is None:
        return 0
    else:
        son= korell(imulti,ifused,True)
        n,m=son.shape
        return rmse(son,numpy.eye(n,m))
```

2.2.3. Kontrolsüz Sınıflandırma Yöntemlerinin Kodlanması

2.2.3.1. K-means Sınıflandırması

Kontrolsüz sınıflandırmada kümeleme yöntemini kullanan K-means yöntemi için pyradar modülü kullanılmıştır. İlk olarak Pyradar modülü kurulmuştur. Pyradar modülü K-means ve ISODATA algoritmalarının uygulanabildiği bir modüldür. Modül kurulumunda yönetici penceresine aşağıdaki kodlar yazılıp kurulum gerçekleştirilmiştir.

```
> easy_install pip
> pip install pyradar
```



Şekil 12. Pyradar modülünün kurulması

Aşağıdaki kodda pyradar.classifiers.kmeans' den, kmeans_classification modülü import edilmiştir. Burada k değeri sınıf sayısını göstermektedir. iter_max ise yapılacak maksimum iterasyon sayısını belirtir. Daha sonra k-means fonksiyonu ile (kmeans_classification(image, k, iter_max)) görüntü, sınıf sayısı ve maksimum iterasyon sayısı ile k-means yöntemi uygulanmaktadır. equalization_using_histogram fonksiyonu ile görüntüye histogram eşitleme uygulanmaktadır bu sayade sınıflandırılmış görüntü piksel değerleri 0-255 arasına gerilir. Daha sonrada görüntü kaydedilir.

```
from pyradar.classifiers.kmeans import kmeans_classification
k= 4
iter_max = 1000
class_image = kmeans_classification(image, k, iter_max)
class_image_eq = equalization_using_histogram(class_image)
save_image(IMG_DEST_DIR, "class_image_eq", class_image_eq)
image_eq = equalization_using_histogram(image)
save_image(IMG_DEST_DIR, "image_eq", image_eq)
```

2.2.3.2. ISODATA Yöntemine Göre Kontrolsüz Sınıflandırma

ISODATA yöntemi için Python modüllerinden pyradar kullanılmıştır. Burada ilk olarak import ile isodata_classification modülü programa import edilmektedir. ISODATA fonksiyonuna verilen parametrelere göre görüntü sınıflandırılmaktadır.

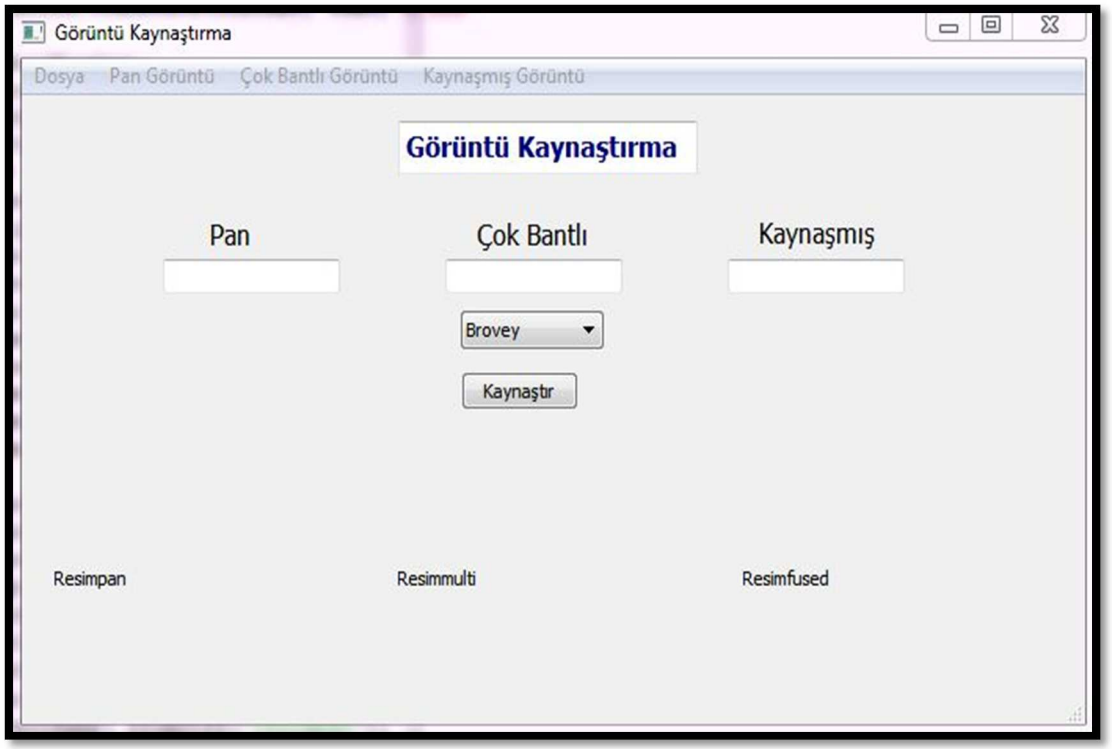
K = sınıf sayısını göstermektedir.
I=Yapılacak maksimum iterasyon sayısını gösterir.
P=Birleşmiş olabilecek maksimum küme sayısı
THETA_M=Herbir küme için eşik değeri
THETA_S =Standart sapma eşik değeri
THETA_C = Eş mesafeler için eşik değerini göstermektedir.

```
from pyradar.classifiers.isodata import isodata_classification
params = {"K": 15, "I" : 100, "P" : 2, "THETA_M" : 10, "THETA_S" : 0.1,
        "THETA_C" : 2, "THETA_O" : 0.01}
class_image = isodata_classification(img, parameters=params)
    class_image_eq = equalization_using_histogram(class_image)
save_image(IMG_DEST_DIR, "class_image_eq", class_image_eq)
image_eq = equalization_using_histogram(image)
    save_image(IMG_DEST_DIR, "image_eq", image_eq)
```

2.2.4. Qt Designer' da Oluşturulan Arayüzler

2.2.4.1. Görüntü Kaynaştırma Arayüzü

Görüntü kaynaştırma arayüzü Brovey, IHS ve dalgacık dönüşümü yöntemlerinin kullanılabildiği bir arayüzdür. Bu arayüzde dosya menüsünün altında yüksek konumsal çözünürlüklü pankromatik görüntünün seçilebildiği "Pan Görüntüyü Seç", düşük konumsal çözünürlüklü multispektral görüntünün seçilebildiği "Çok Bantlı Görüntüyü Seç" ve görüntü kaynaştırma işleminin gerçekleşebilmesi için hazırlanan "Kaynaştır" butonu bulunmaktadır. Pan görüntüyü seç butonuna basıldığında görüntü formatı olarak jpg, png ve tif formatlarında bir pankromatik görüntü seçilebilmektedir. Seçilen pankromatik görüntü dizini pan kısmında görülmekte ve pankromatik görüntü arayüzde layout oluşturularak hazırlanan kısımda küçük resim olarak gösterilmektedir. Multispektral görüntünün açılması ve işleme tabi tutulması içinde yine dosya menüsünün altında "Çok Bantlı Görüntüyü Seç" kısmından görüntü arayüze okutulmaktadır. Daha sonra arayüzde oluşturulan açılır araç kutusundan görüntü kaynaştırma yöntemlerinden bir tanesi seçilir. Eğer seçim yapılmadan kaynaştır butonuna basılırsa arayüz otomatik olarak kaynaştırma yöntemlerinden Brovey görüntü kaynaştırma yöntemini uygulamış olacaktır. Arayüzde "Kaynaşmış" yazan kısma oluşturulacak görüntünün adı ve formatı yazılır ve kaynaştır butonuna basılır böylece görüntü kaynaştırma işlemi gerçekleştirilmiş olur. Sonuç görüntüsü kullanılan dizine otomatik olarak kaydedilir. Eğer "Kaynaşmış" yazan kısma sonuç görüntüsünün adı yazılmaz ise otomatik olarak görüntü "output. tif " formatında kaydedilir. Görüntü kaynaştırma arayüzü oluşturulurken yazılan kodlar Ek 2' de verilmektedir.

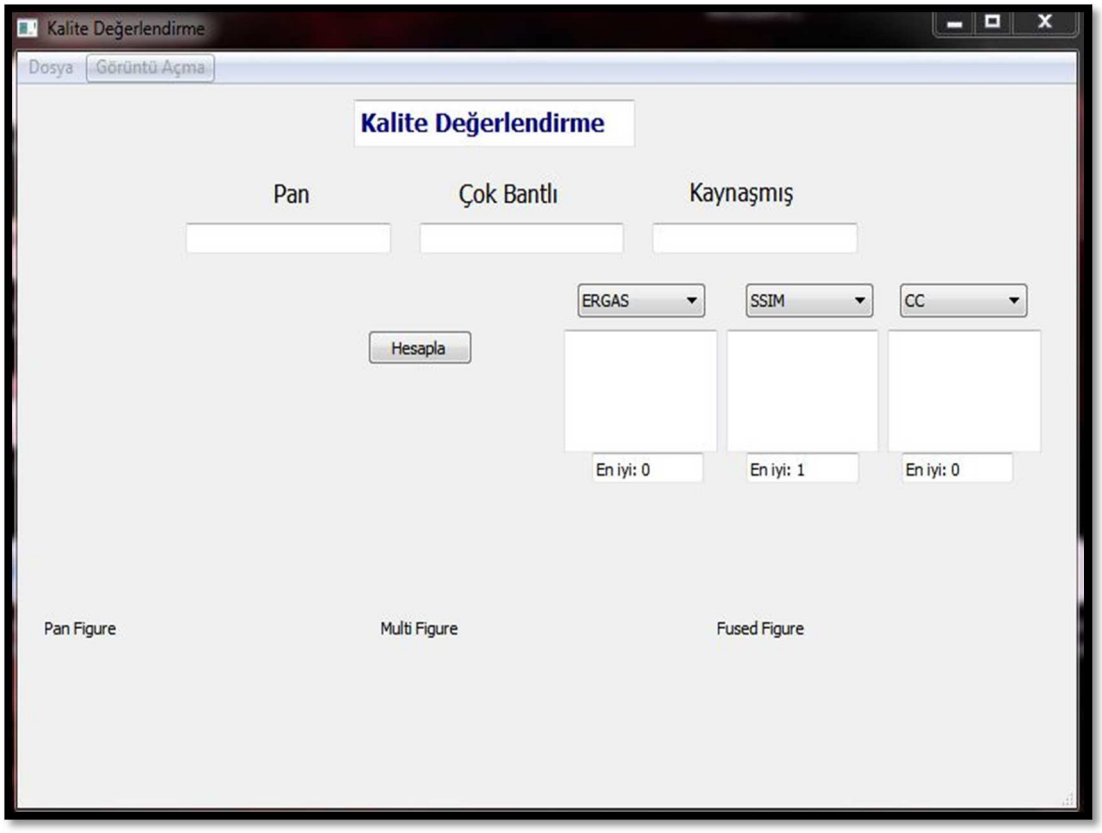


Şekil 13. Görüntü kaynaştırma arayüzü

Şekil 14. Görüntü kaynaştırma arayüzü Brovey yöntemi uygulanmış hali

2.2.4.2. Kalite Değerlendirme Arayüzü

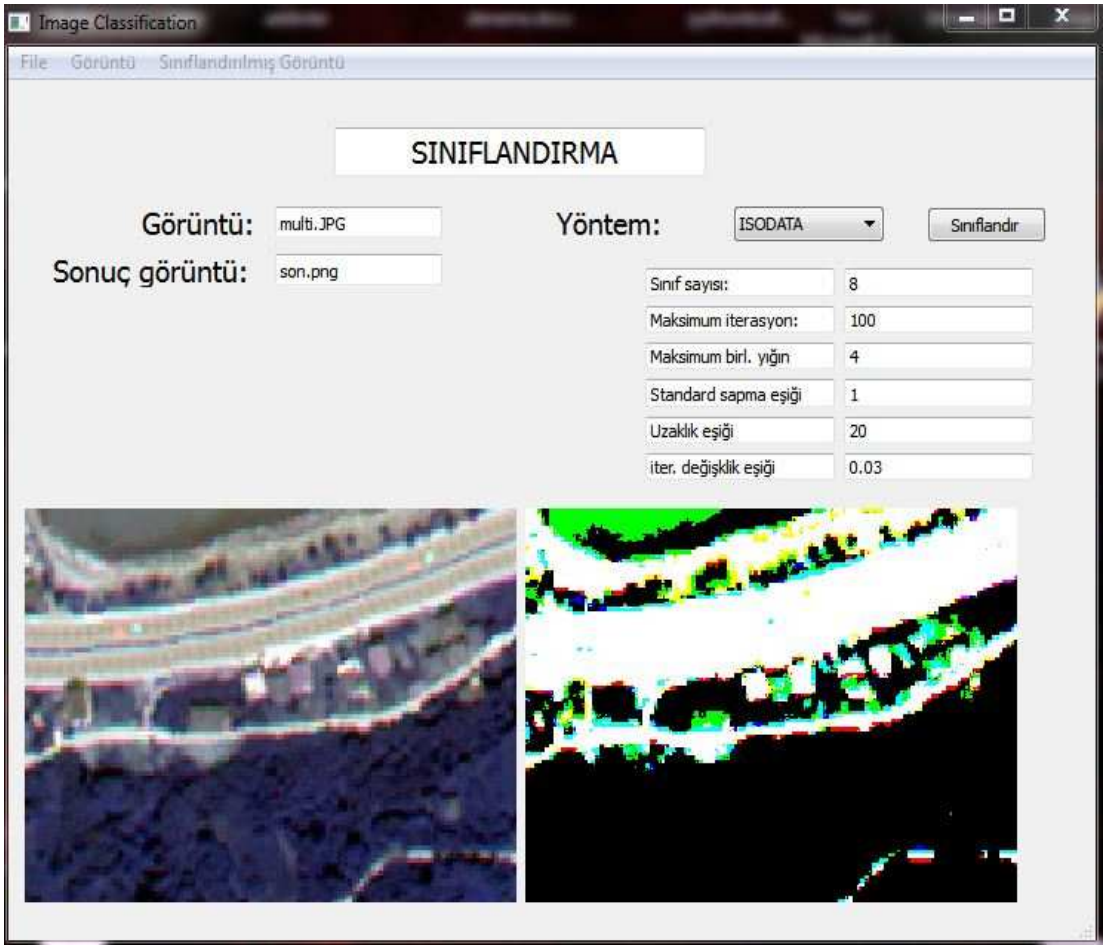
Kalite değerlendirme arayüzü ERGAS, SSIM ve CC kalite değerlendirme metriklerini içermektedir. Dosya menüsü altında “Dizin Seç”, “Pan Görüntü Seç” ,”Kaynaşmış Görüntü Seç” “Görüntü Seç” ve “Hesapla” butonları bulunmaktadır. Dizin seç ile görüntülerin açılacağı klasör otomatik olarak tanımlatılması sağlanmaktadır. Bundan sonra pan görüntüyü seç kullanıldığında otomatik olarak resim dosyasının bulunduğu klasör açılır. Bu arayüzün çalışması da görüntü kaynaştırma arayüzü ile aynıdır. Pankromatik görüntü arayüze okutulur ve seçilen görüntü küçük resim olarak arayüzde görülür sonra multispektral görüntü arayüze okutulur ve son olarak bu pankromatik ve multispektral görüntünün herhangi bir kaynaştırma yöntemine göre kaynaştırılması sonucu oluşturulan görüntü açılır. Başka bir işlem yapılmadan hesapla butonuna basılırsa Ergas, SSIM ve CC kalite değerlendirme metriklerine göre hesap yapıp sonuçları gösterir. Kalite değerlendirme arayüzü oluşturulurken yazılan kodlar Ek 3’ de verilmektedir.



Şekil 15.Kalite değerlendirme arayüzü

2.2.4.3. Görüntü Sınıflandırma Arayüzü

Görüntü sınıflandırma arayüzü ISODATA ve k-means yöntemlerine göre sınıflandırma yapabilmektedir. Arayüzde File menüsünden sınıflandırılacak görüntü seçilir ve ISODATA sınıflandırması için sınıf sayısı, maksimum iterasyon sayısı, maksimum birim yığınlığı, standart sapma eşiği, uzaklık eşiği, iterasyon değişiklik eşiği değerleri parametre olarak girilebilmektedir. Görüntü sınıflandırma arayüzü oluşturulurken yazılan kodlar Ek 4' de verilmektedir.

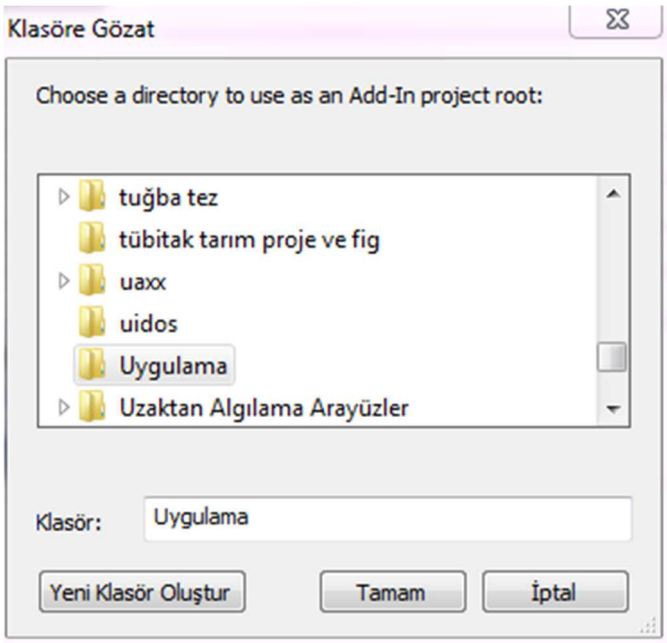


Şekil 16. Görüntü sınıflandırma arayüzü

2.2.5. ArcGIS Yazılımına Arayüzün Eklenmesi

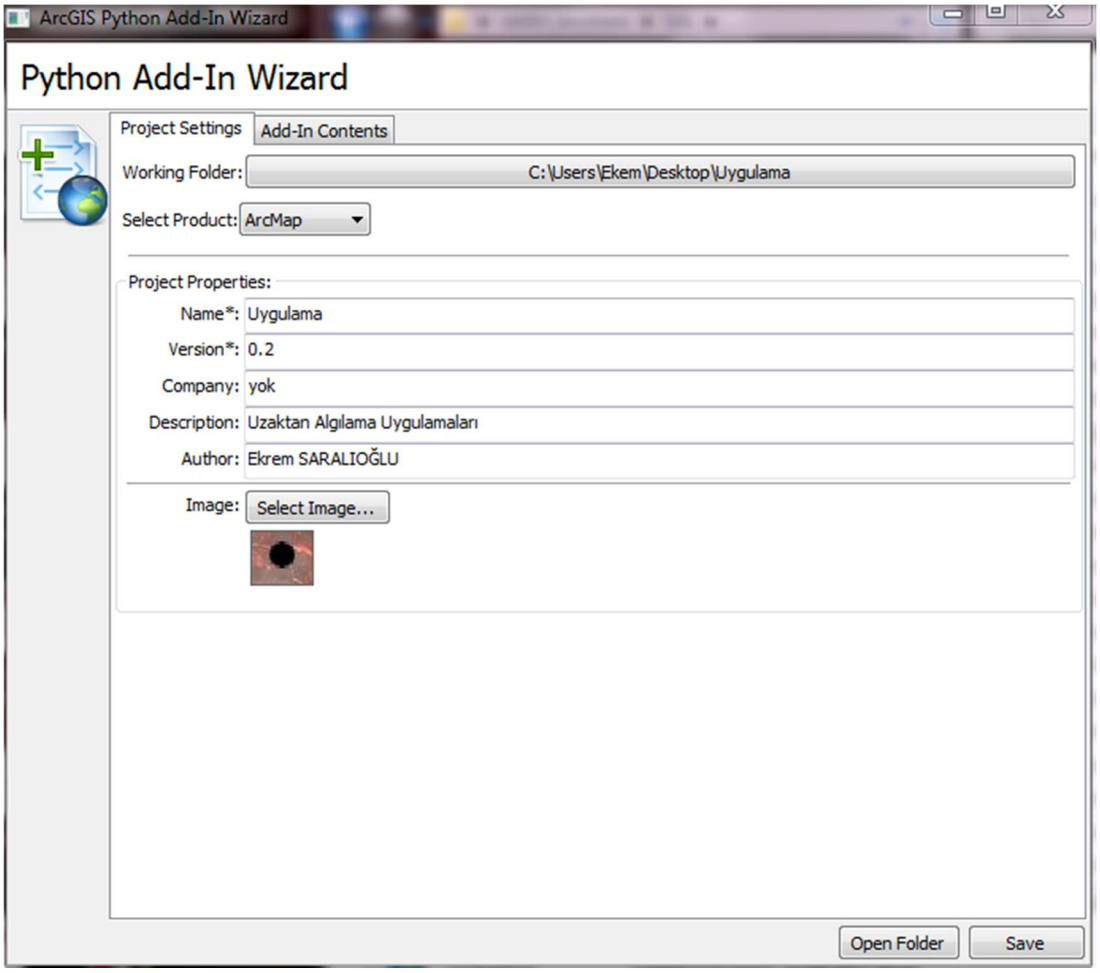
ArcGIS yazılımına eklenti yapılmasında aşağıdaki işlem adımları takip edilmiştir:

1. ArcGIS için eklenti yapılmasında kullanılan addin_assistant dosyası içerisindeki addin_assistant.exe dosyasına açılarak uygulamanın oluşturulacağı Uygulama adlı dosya seçilmiştir.



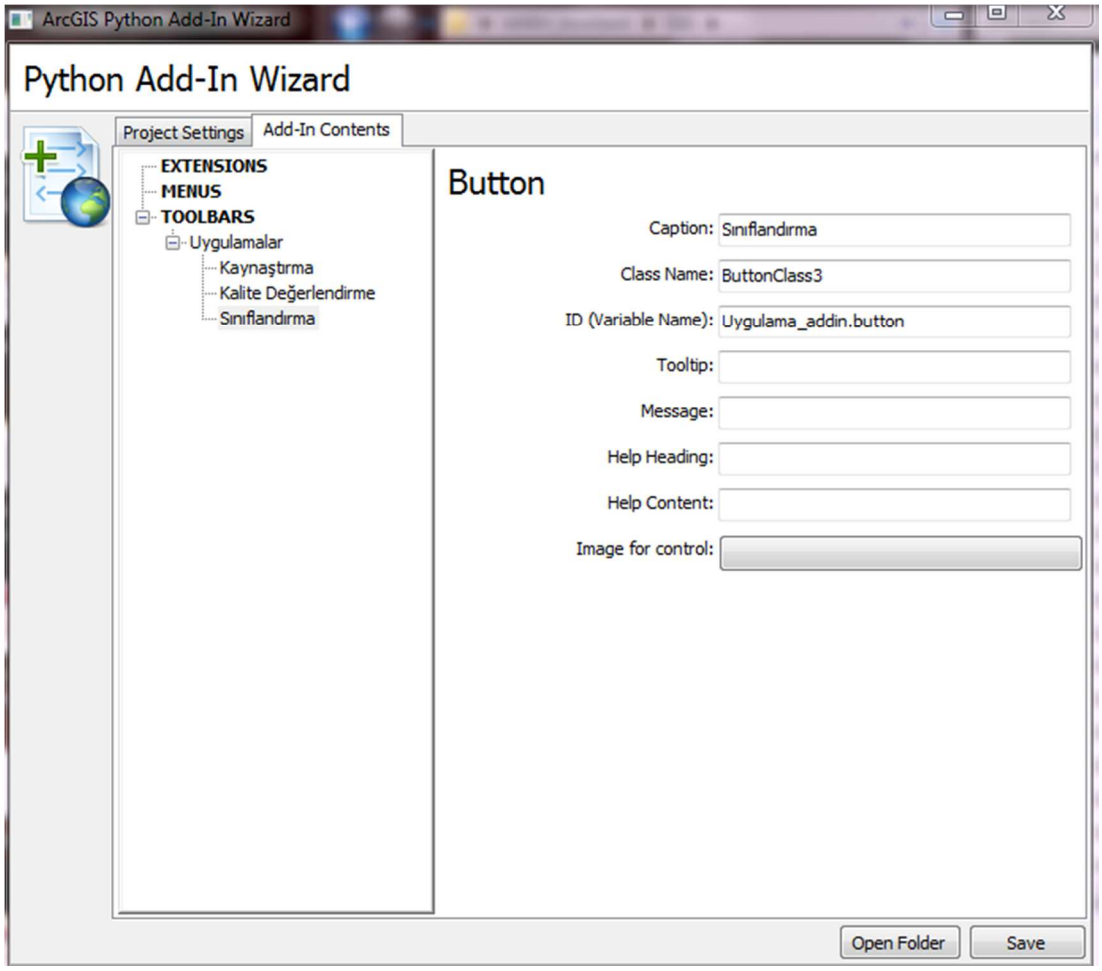
Şekil 17. Arayüzün oluşturulacağı dosyanın belirlenmesi

2. Python eklenti sihirbazından proje ayarları kısmından proje adı, kullanılan sürüm, şirket, tanımlama ve yazar gibi projenin oluşturulmasında kullanılacak açıklama kısımları oluşturulmuştur.



Şekil 18. Proje ayarları penceresi

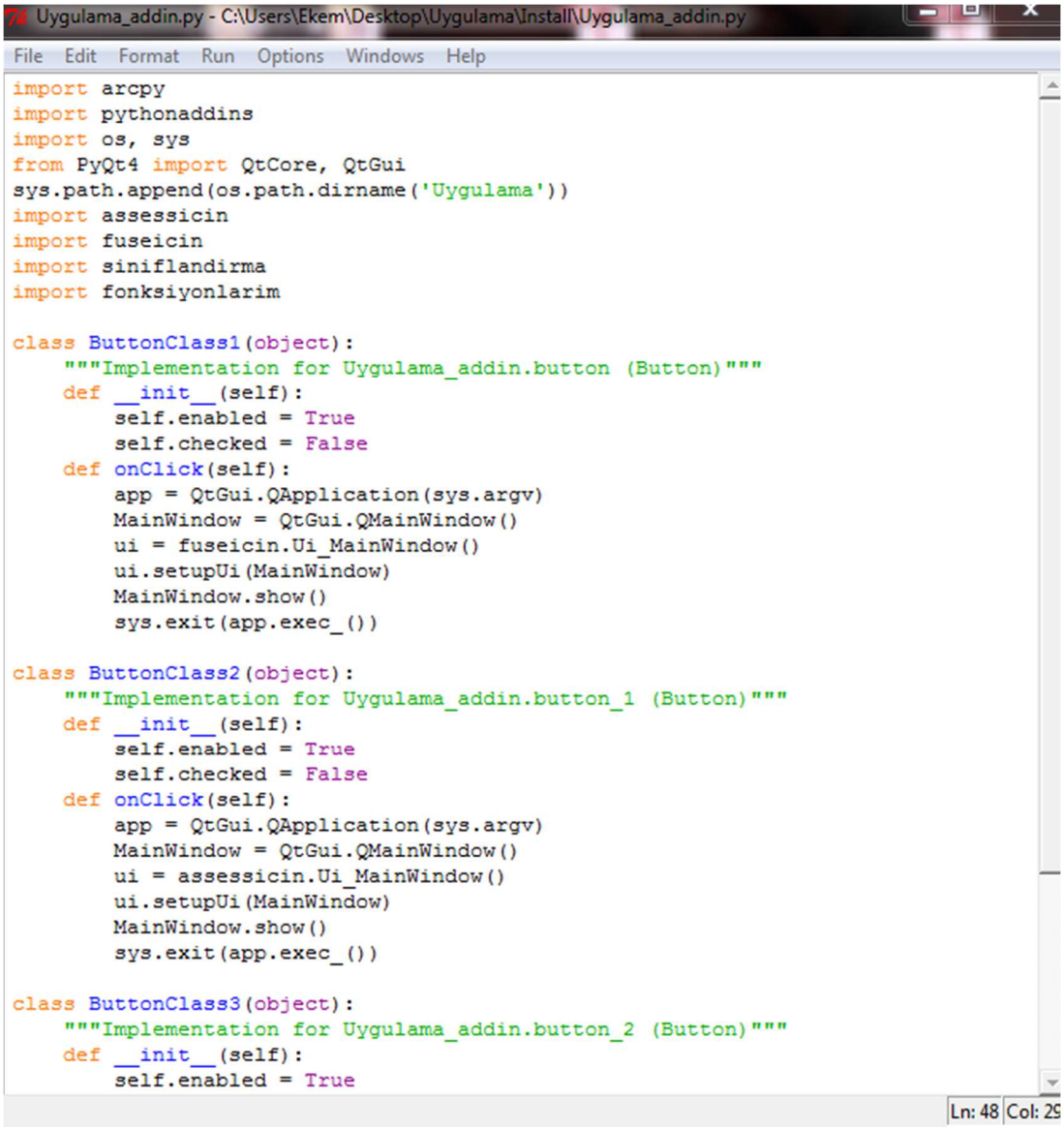
3. Eklentilerin içeriklerinin belirlendiği Add-In Contents kısmından Şekil 19. ’ da görüldüğü gibi Uygulama adında bir araç çubuğu (toolbar) oluşturuldu. Bu araç çubuğunun içine hazırlanan arayüzü aktif edecek butonlar sağ tuşa basılarak buton ekle ile oluşturuldu. Burada oluşturulan üç arayüz için Kaynaştırma, Kalite Değerlendirme ve Sınıflandırma adında üç tane buton oluşturulmuştur. Bu işlemden sonra dosya kaydedilmiştir.



Şekil 19. Araç kutusu ve buton oluşturma

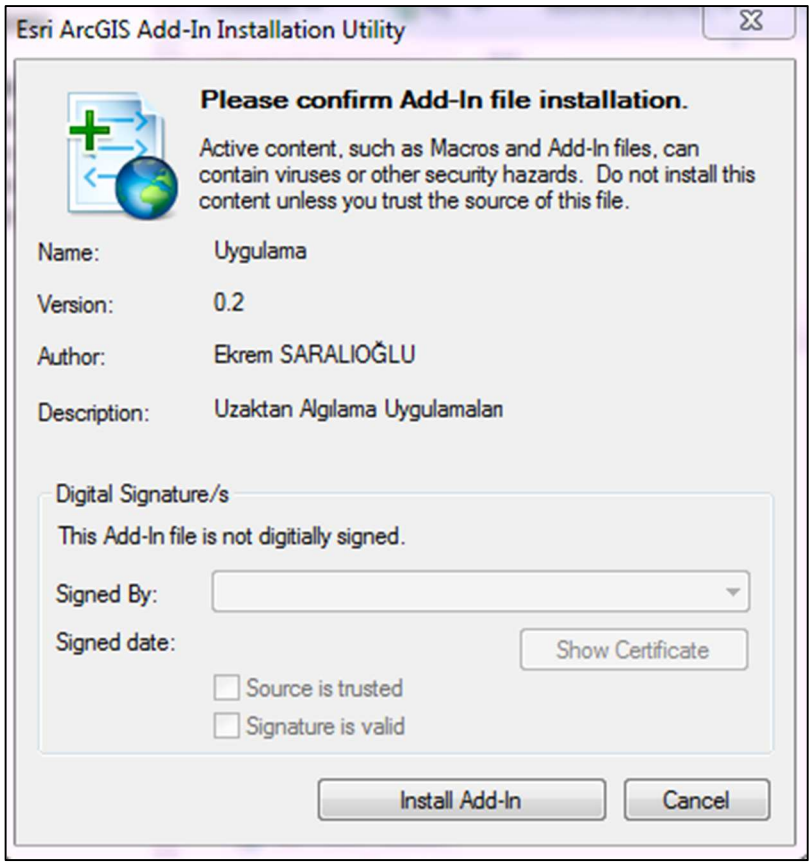
4. Dosya kaydedildikten sonra projenin oluşturulduğu klasörde Images, Install, config.xml ve makeaddin.py dosyaları otomatik olarak oluşmaktadır. Bu klasörde Install dosyasının içerisine görüntü kaynaştırma işlemleri için oluşturulan fuseicin.py, kalite değerlendirme için oluşturulan assesicin.py, sınıflandırma işlemleri için oluşturulan sınıflandırma.py dosyaları ve bu dosyaların içinde kullanılan bazı işlemlerde kolaylık sağlanması için yazılan fonksiyonlarım.py dosyası atılmıştır.
5. Daha sonra otomatik olarak oluşturulan fakat butonlara bağlantıların yapılmadığı Uygulama_addin.py dosyası Python editör ile açılarak arcpy ve pythonaddins modüllerinin yüklü olduğu kısma gerekli diğer modüller yüklenmiştir. Bu modüller assessicin, fuseicin, siniflandirma, fonksiyonlarim, os, sys, QtCore ve QtGui modülleridir.

6. Uygulama dosyasının olduğu dizin sys.path.append(os.path.dirname (‘Uygulama’)) komutu ile programa tanıtılmıştır.
7. Oluşturulan butonlara kaynaştırma, kalite değerlendirme ve sınıflandırma arayüzlerini tetikleyecek komutlar yazılmıştır. Bu işlem için def oneClick ile tıklama işleminin tanımlandığı yerde pass kısmı silinip bu kısma komutların yazılmasıyla gerçekleştirilmiştir. Yazılan kodlar aşağıda Şekil 20’ de gösterilmektedir.



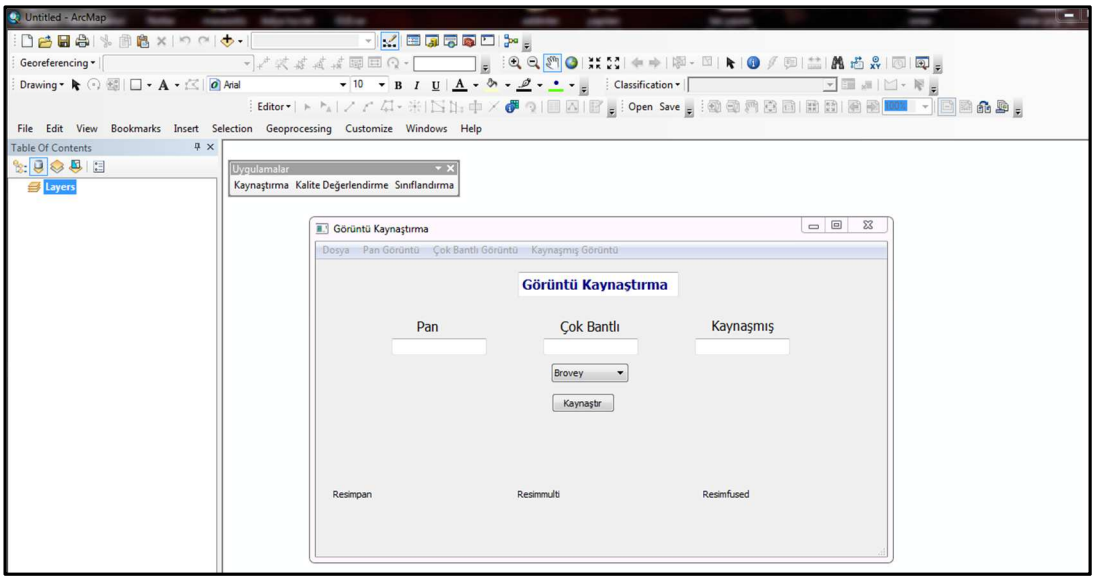
Şekil 20. Arayüzlerin butonlara bağlanması

8. Uygulama kalasörü içerisinde bulunan makeaddin.py dosyasına tıklanarak uygulama.esriaddin dosyası oluşturulmuştur. Bu dosya görünümü aşağıda Şekil 21’ de gösterilmektedir. Install Add-in komutu ile eklenti ArcGIS yazılımına yüklenmiştir.



Şekil 21. Uygulamanın ArcGIS yazılımına yüklenmesi

9. Oluşturulan eklentiyi açmak için ArcGIS yazılımı açılarak araç çubukları bölümünde sağ tuşa basılıp araç çubukları açılmıştır. Bu araç çubuklarının içinde hazırlanan Uygulamalar ismindeki araç çubuğu da oluşmuştur. Aşağıda Şekil 22’ de Uygulamalar adında hazırlanmış ArcGIS eklentisi gösterilmektedir.



Şekil 22. ArcGIS oluşturulan uygulamalar eklentisi

3. BULGULAR VE İRDELEMELER

ArcGIS 10.0 sürümü görüntü kaynaştırmada IHS, Brovey, Simple Mean ve kendi oluşturdukları ESRI kaynaştırma yöntemlerinin kullanılmasına imkan sağlamaktadır. ArcGIS yazılımı 10.2 sürümüne ise yeni bir yöntem olarak Gram-Schmidt yöntemi eklenmiştir. Fakat bu yöntemler literatürde detay korumada başarıyı arttıran fakat renk bilgilerini korumada etkinliği çok olan yöntemler değildir. Dalgacık tabanlı yöntemler ise renk bilgilerini daha iyi koruyabilmektedir ve görüntü kaynaştırmada yaygın olarak kullanılmaktadırlar. Fakat bu yöntemler ArcGIS yazılımında mevcut değildir.

Kaynaştırılmış görüntülerin kalitelerinin değerlendirilmesinde çeşitli metrikler kullanılmakladır. Bu kalite değerlendirme için bir keskinleştirilmiş görüntünün orijinal MS görüntü ile ne kadar uyumlu ve benzer olduğu metrikler ile kontrol edilir. ArcGIS yazılımı görüntü kaynaştırma yöntemlerinden bazılarına imkan sağlamasına rağmen kaynaşmış görüntülerde kalite değerlendirmesi için uygulamalara imkan sağlamamaktadır. Bu nedenle kullanıcılar yaptıkları görüntü kaynaştırmanın kalitelerinin değerlendirememektedirler. Oluşturulan arayüzde kalite değerlendirmesi için ERGAS, SSIM ve CC metrikleri kullanılmıştır. Bu sayede kullanıcılar görüntü kaynaştırma uygulaması yaptıklarında kaynaşmış görüntülerin hassasiyetlerini değerlendirebileceklerdir.

İlk olarak ArcGIS 9.0 sürümü ile Python yazılımı ArcGIS kullanıcılarına tanıtılmıştır. ArcGIS 10.0 sürümü ile de ArcGIS yazılımında çok yaygın bir rol oynamaya başlamıştır. Bundan sonra çıkan her sürümde de Python' un ArcGIS' de kullanım alanları arttırılarak devam etmiştir. Öyle ki ArcGIS yazılımı yönetici penceresi bölümüne Python penceresi eklenmiştir. Bu da ArcGIS yazılımının Python programlama dilini iyice benimsediğini göstermektedir.

ArcGIS yazılımı 10.0 sürümü ile ArcGIS masaüstü ürünlerine kolayca eklenti yapılabilmesi ve paylaşılabilmesi için Add-ins (eklenti) yapısını kullanmaya başlamıştır. Add-ins ile kullanıcılar arasında herhangi bir başka program yüklemeden paylaşım yapılabilmektedir. Oluşturulan bir Add-ins dosyası ArcGIS yazılımına yüklendikten sonra "Install" klasörü içeresinde oluşturulan kullanıcı arayüzü elemanlarına yapılacak uygulamaların eklendiği Python dosyasında güncelleme yapıldığında bazı durumlarda araç çubuğundaki butonların kaybolduğu görülmüştür. Bir başka durumda da yazılan kodlarda ki güncellemelerin programa aktarılmadığı görülmüştür.

Python programlama dili uzaktan algılama yöntemlerini içeren uygulamaların kodlanmasında ve arayüz oluşturmada etkili bir programdır. Uzaktan algılama ile ilgili uygulamalarda kodların yazımı için geniş Python kütüphaneleri mevcuttur. Ayrıca Python programının açık kaynak kodlu olması ve dünya çapında geniş bir kullanıcı kitlesi olması, yapılan çalışmaların paylaşılması ve geliştirilmesi sayesinde uzaktan algılama uygulamaları Python programlama dili ile kodlanabilmektedir.

Python programı ile arayüzler tasarlanabilmektedir. Birden çok platformu destekleyen bir grafiksel kullanıcı arayüzü geliştirme araç takımı Qt' nin Python ile uyumlu sürümü olan PyQt4 ile kapsamlı arayüzler tasarlanabilmektedir. Ayrıca PyQt' nin Qt tasarımcısı kullanılarak arayüz tasarlarken oluşturulacak pencere araçları kod yazmaya gerek kalmadan sürükle bırak yöntemiyle oluşturulabilmektedir. Pythonun kendi arayüz hazırlama kütüphanesi Tkinter da arayüz hazırlama için bir tasarımcı yoktur ayrıca PyQt ile daha kapsamlı arayüzler tasarlanabildiği görülmüştür. Bu nedenle arayüz tasarımında önce Tkinter denenmiş fakat Qt' nin avantajlarından dolayı PyQt4 ile arayüzler oluşturulmuştur.

Python programlama dilinin ücretsiz olması oldukça önemli bir avantajdır. ERDAS, NVI, ArcGIS gibi ticari yazılımlar oldukça yüksek maliyetlerde satılmaktadırlar ve bu yazılımların bir birlerine göre üstünlükleri ve eksiklikleri vardır. Bu yazılımlardan birini satın alan bir araştırmacı karşılaştığı bir eksikliği giderebilmek için programlama dillerine ihtiyaç duymaktadır. Python programlama dili geniş kütüphaneleri ve modülleri sayesinde bu ihtiyacı karşılamaktadır.

4. SONUÇ VE ÖNERİLER

Uzaktan algılama ile etkileşimli olarak kullanılan CBS' nin önde gelen yazılımlarından ArcGIS 10.2 yazılımı uzaktan algılama uygulamalarında eksiklikler içermektedir. Görüntü kaynaştırma yöntemlerinden literatürde çokça kullanılan dalgacık tabanlı yöntem bu yazılımda mevcut değildir. Görüntü kaynaştırma arayüzünde bu eksiklik giderilmiştir. Ayrıca kaynaştırılmış görüntülerin kalitelerinin değerlendirilmesi için kullanılan kalite değerlendirme metrikleri bu yazılımda yoktur. Yani kullanıcılar bir görüntüyü kaynaştırdığında bu görüntünün ne kadar doğrulukta olduğunu bilememektedirler. Oluşturulan arayüz ile kalite değerlendirme kısmında bu eksiklik giderilmektedir.

Uzaktan algılama araştırmacılarına yönelik bir arayüz oluşturulmuştur. Bu arayüz Python programlama dili ile kodlanıp PyQt4 ile tasarlanmıştır. Bu arayüz de görüntü kaynaştırma (Brovey, IHS ve dalgacık tabanlı) yöntemleri uygulanabilmektedir. Bu sayede çok bantlı renkli uydu görüntülerinin konumsal çözünürlüklerinin aynı bölgeye ait daha iyi konumsal çözünürlüğe sahip pankromatik görüntülerle iyileştirilerek pankromatik görüntüdeki konumsal detay çok bantlı görüntüye aktarılması sağlanmış olmaktadır. Bu arayüzde, yapılan görüntü kaynaştırmanın kalite değerlendirmesi de yapılabilmektedir. Bunun için (ERGAS, SSIM ve CC) metrikleri kodlanıp arayüze eklenmiştir. Bu metrikler ile kullanıcılar yapılan bir görüntü kaynaştırmanın hassasiyetini öğrenebileceklerdir. Son olarak bu arayüze kontrolsüz görüntü sınıflandırma için ISODATA ve k-means yöntemleri eklenmiştir.

Python programlama dili, içerdiği geniş kütüphaneler ve modülleri sayesinde görüntü işleme ile ilgili uygulamaların yapılmasında etkin olarak kullanılabilecek bir programlama dilidir. Fakat çok fazla sayıda kütüphane ve modül içermesi her modülün de bir biri ile uyumlu olarak kullanılabileceği anlamına gelmemektedir. Örneğin bir görüntü dosyası PIL (Python Imaging Library) kütüphanesi kullanılarak açıldığında farklı bir formatta programda tutulurken GDAL (Geospatial Data Abstraction Library) kütüphanesi kullanıldığında daha başka bir formatta bulunmaktadır. Bu nedenle yapılacak çalışmalarda kütüphanelerden uygun olanları belirlenip o kütüphaneler kullanılmalıdır.

5. KAYNAKLAR

Alparone, L., Baronti, S., Garzelli, A. and Nencini, F., 2004. A Global Qality Measurament of Pan-Sharpened Multispectral Imagery. IEEE Geoscince and Remote Sensing Letters, 1, 4, 313-317.

ArcGIS Resorces. http://resources.arcgis.com/en/help/arcobjects-net/conceptualhelp/ index html#//0001000000tv000000, 15 Haziran 2015.

Beşer, M., 2013. Python, 2. Baskı, dikeyeksen, İstanbul.

Dangermond, How We Hit $912 Million in Sales, http://www.inc.com/magazine/201307/ christine-lagorio/jack-dangermond-how-he-started-esri.html, 10 Ocak 2015.

Elkner, J., Downey, A. B. ve Meyers, C., 2008, Bilgisayar Bilimcisi gibi Düşünmek, Tahir Emre Kalaycı, 2. Baskı, http:// cse.cbu. edu.tr/ ~tekrei /dersler/ bbgd_p/ BBGD_PIO.pdf, 21 Haziran 2014.

Esri Online Help, http://proceedings.esri.com/library /userconf/devsummit11/papers/ tech/ hitchhikers_guide_to_python_and_arcgis.pdf, , 05 Haziran 2014.

Francica , J., Update: Esri has 40+% of GIS Marketshare, http://www.directionsmag.com/ entry/esri-has-40-of-gis-marketshare/215188, 21 Ekim 2011.

Gungor, O., 2008. Multi Sensor Multi Resolution Image Fusion, Phd Thesis, Purdue University, Indiana, USA.

Harwani, B. M., 2012. Introduction to Python® Programming and Developing GUI Applications with PyQT, Course Technology, Boston.

Honeycutt, D. ve Wynne, D., 2010. ESRI Developer Summit, Getting Started With Python in ArcGIS.

Kavzoğlu, T. ve Çölkesen, İleri Uzaktan Algılama Teknolojileri ve Uygulama Alanları.

Longley, P., Goodchild. M., Maguire, D., ve Rhind, D., 2011. Geographic Information Systems and Science, 3rd Edition.

Lutz, M., 2013. Learning Python, 5th Edition, O'Reilly Media.

Martinez, W. L., 2011. Graphical user interfaces, WIREs Comp Stat, 3: 119–133. doi: 10.1002/ wics.150.

Malkoç , B., 2012, Akademik Bilişim'12 - XIV. Akademik Bilişim Konferansı Bildirileri, Şubat, Uşak, Bildiriler kitabı, 200-210.

Özgül, F., 2013. Her Yönüyle Python, 2. Baskı, Kodlab, İstanbul.

Özgül, F., Sözlükler, http://belgeler.istihza.com/py3/sozlukler.html, 2 Ocak 2015.

Özendi, M., Topan, H., Oruç, M. ve Cam, A., 2014. Pléidas-1A Görüntülerinin Pan Sharpening Performansının İncelenmesi, 5. Uzaktan Algılama ve CBS Sempozyumu, İstanbul.

Python Documantation, https://docs.python.org/2/library/sets.html?highlight=sets#set-objects, 13 Haziran 2015

Pimpler, E.,2013. Programming ArcGIS 10.1 with Python Cookbook: Packt Publishing Ltd.

Raymond, E., 2000. LinuxJournal.com: Why Python?, http://www.linuxjournal.com/ article/3882, 01 Haziran 2014.

Rossum, G., 2003. The Python Tutorial, https://docs.python.org/2/tutorial/index.html20 Ocak 2015.

Sanderson, R., Introduction to Remote Sensing, New Mexico State University, 2010.

Swaroop, C. H., 2003. A Byte of Python.

Xua, H. ve Gaob, P., 2008. Custom Image Processing Capabilities In ArcGIS, The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Vol. XXXVII, Part B4, 264-266

Venners, B., 2003. Artima Developer: Python and the Programmer, A Conversation with Bruce Eckel, Part I,http://www.artima.com/intv/aboutme.html, 15 Aralık 2014.

Yomralıoğlu, T., Coğrafi Bilgi Sistemleri: Temel Kavramlar ve Uygulamalar, Beşinci Baskı, İstanbul. 2000.

Yıldırım, D ve Güngör, O., 2012. IKONOS Uydu Görüntüleri ile Yeni Bir Görüntü Kaynaştırma Yöntemi, Jeodezi ve Jeoinformasyon Dergisi, Cilt 1, Sayı 1, 75-83

URL-1, http://www.esriturkey.com.tr/eğitim, 7 Şubat 2014.

URL2, http://resources.arcgis.com/en/help/gettingstarted/articles/026n000014000000. htm.

URL-3, http://www.pythondersleri.com/2014/09/numpy-scipy.html, 2 Aralık 2014.

URL-4, http://belgeler.istihza.com/py3/kumeler_ve_dondurulmus_kumeler.html, Python 3 için Türkçe Kılavuz, Kümeler, 3 Aralık 2014.

URL-5, http://belgeler.istihza.com/py3/islecler.html, Python 3 için Türkçe Kılavuz, İşleçler, 3 Aralık 2014.

URL-6, http://www.pythondersleri.com/2013/04/kosul-ifadeleri.htm, 5 Aralık 2014.

URL-7, http://www.pythondersleri.com/2013/04/kosul-ifadeleri.html, 5 Aralık 2014.

URL-8, http://www.pythondersleri.com/2013/04/kosul-ifadeleri.html, 5 Aralık 2014.

URL-9, http://toastytech.com/guis/guitimeline.html, 15 Ocak 2015.

URL-10, http://tr.wikipedia.org/wiki/Qt_%28%C3%A7al%C4%B1%C5%9Fma_alanC4 12 Şubat 2015.

6. EKLER

EK 1. SSIM yöntemi kodları

```
def ssim(im1, im2, window=None, k=(0.01, 0.03), l=255):
    from scipy.signal import fftconvolve,general_gaussian
    im1=ortbant(im1)
    im2=ortbant(im2)
    if window is None:
       boyboy=11
       window=general_gaussian(boyboy**2,0.95,sig=2,sym=True)
       window=window.reshape((boyboy,boyboy),order="F")
    for a, b in zip(window.shape, im1.shape):
       if a > b:
          return None, None
    # Values in k must be positive according to the base implementation.
    for ki in k:
       if ki < 0:
          return None, None
    c1 = (k[0] * l) ** 2
    c2 = (k[1] * l) ** 2
    window = window/numpy.sum(window)
    mu1 = fftconvolve(im1, window, mode='valid')
    mu2 = fftconvolve(im2, window, mode='valid')
    mu1_sq = mu1 * mu1
    mu2_sq = mu2 * mu2
    mu1_mu2 = mu1 * mu2
    sigma1_sq = fftconvolve(im1 * im1, window, mode='valid') - mu1_sq
    sigma2_sq = fftconvolve(im2 * im2, window, mode='valid') - mu2_sq
    sigma12 = fftconvolve(im1 * im2, window, mode='valid') - mu1_mu2
```

Ek 1’ in devamı

```
    if c1 > 0 and c2 > 0:
       num = (2 * mu1_mu2 + c1) * (2 * sigma12 + c2)
       den = (mu1_sq + mu2_sq + c1) * (sigma1_sq + sigma2_sq + c2)
       ssim_map = num / den
    else:
       num1 = 2 * mu1_mu2 + c1
       num2 = 2 * sigma12 + c2
       den1 = mu1_sq + mu2_sq + c1
       den2 = sigma1_sq + sigma2_sq + c2
       ssim_map = numpy.ones(numpy.shape(mu1))
       index = (den1 * den2) > 0
       ssim_map[index] = (num1[index] * num2[index]) / (den1[index] * den2[index])
       index = (den1 != 0) & (den2 == 0)
       ssim_map[index] = num1[index] / den1[index]

    mssim = ssim_map.mean()
    return mssim**(0.4)
```

Ek 2. Görüntü Kaynaştırma Arayüzü Kodları

```
# -*- coding: utf8 -*-
from PyQt4 import QtCore, QtGui
from fonksiyonlarim import jpegyapgoster, toutf8

try:
    _fromUtf8 = QtCore.QString.fromUtf8
except AttributeError:
    def _fromUtf8(s):
        return s

try:

 _encoding = QtGui.QApplication.UnicodeUTF8
    def _translate(context, text, disambig):
        return QtGui.QApplication.translate(context, text, disambig, _encoding)
except AttributeError:
    def _translate(context, text, disambig):
        return QtGui.QApplication.translate(context, text, disambig)

class Ui_MainWindow(QtGui.QMainWindow):
    def setupUi(self, MainWindow):
        MainWindow.setObjectName(_fromUtf8("MainWindow"))
        MainWindow.resize(678, 385)
        self.centralwidget = QtGui.QWidget(MainWindow)
        self.centralwidget.setObjectName(_fromUtf8("centralwidget"))
        self.baslik = QtGui.QTextBrowser(self.centralwidget)
        self.baslik.setGeometry(QtCore.QRect(240, 15, 191, 31))
        self.baslik.setObjectName(_fromUtf8("baslik"))
        self.metodlar = QtGui.QComboBox(self.centralwidget)
        self.metodlar.setGeometry(QtCore.QRect(280, 125, 91, 22))
        self.metodlar.setObjectName(_fromUtf8("metodlar"))
```

Ek 2’ nin devamı

```
            self.metodlar.addItem(_fromUtf8(""))
            self.metodlar.addItem(_fromUtf8(""))
            self.metodlar.addItem(_fromUtf8(""))

            self.horizontalLayoutWidget = QtGui.QWidget(self.centralwidget)
            self.horizontalLayoutWidget.setGeometry(QtCore.QRect(1, 185, 678, 181))

self.horizontalLayoutWidget.setObjectName(_fromUtf8("horizontalLayoutWidget"))
            self.resimlayout = QtGui.QHBoxLayout(self.horizontalLayoutWidget)
            self.resimlayout.setSpacing(20)
            self.resimlayout.setContentsMargins(20, -1, 20, 0)
            self.resimlayout.setObjectName(_fromUtf8("resimlayout"))
            self.resimpan = QtGui.QLabel(self.horizontalLayoutWidget)
            self.resimpan.setObjectName(_fromUtf8("resimpan"))
            self.resimlayout.addWidget(self.resimpan)
            self.resimmulti = QtGui.QLabel(self.horizontalLayoutWidget)
            self.resimmulti.setObjectName(_fromUtf8("resimmulti"))
            self.resimlayout.addWidget(self.resimmulti)
            self.resimfused = QtGui.QLabel(self.horizontalLayoutWidget)
            self.resimfused.setObjectName(_fromUtf8("resimfused"))
            self.resimlayout.addWidget(self.resimfused)

            self.panad = QtGui.QLineEdit(self.centralwidget)
            self.panad.setGeometry(QtCore.QRect(90, 95, 113, 20))
            self.panad.setObjectName(_fromUtf8("panad"))
            self.multiad = QtGui.QLineEdit(self.centralwidget)
            self.multiad.setGeometry(QtCore.QRect(270, 95, 113, 20))
            self.multiad.setObjectName(_fromUtf8("multiad"))
            self.fusedad = QtGui.QLineEdit(self.centralwidget)
            self.fusedad.setGeometry(QtCore.QRect(450, 95, 113, 20))
            self.fusedad.setObjectName(_fromUtf8("fusedad"))
```

Ek 2’ nin devamı

```
self.dugme = QtGui.QPushButton(self.centralwidget)
self.dugme.setGeometry(QtCore.QRect(280, 160, 75, 23))
self.dugme.setObjectName(_fromUtf8("dugme"))
self.pansabit = QtGui.QLabel(self.centralwidget)
self.pansabit.setGeometry(QtCore.QRect(120, 70, 61, 21))
font = QtGui.QFont()
font.setPointSize(12)
self.pansabit.setFont(font)
self.pansabit.setObjectName(_fromUtf8("pansabit"))
self.multisabit = QtGui.QLabel(self.centralwidget)
self.multisabit.setGeometry(QtCore.QRect(290, 70, 71, 21))
font = QtGui.QFont()
font.setPointSize(12)
self.multisabit.setFont(font)
self.multisabit.setObjectName(_fromUtf8("multisabit"))
self.fusedsabit = QtGui.QLabel(self.centralwidget)
self.fusedsabit.setGeometry(QtCore.QRect(470, 70, 100, 20))
font = QtGui.QFont()
font.setPointSize(12)
self.fusedsabit.setFont(font)
self.fusedsabit.setObjectName(_fromUtf8("fusedsabit"))
MainWindow.setCentralWidget(self.centralwidget)
self.menubar = QtGui.QMenuBar(MainWindow)
self.menubar.setGeometry(QtCore.QRect(0, 0, 678, 21))
self.menubar.setObjectName(_fromUtf8("menubar"))
self.menuPan = QtGui.QMenu(self.menubar)
self.menuPan.setObjectName(_fromUtf8("menuPan"))
self.menuMulti = QtGui.QMenu(self.menubar)
self.menuMulti.setObjectName(_fromUtf8("menuMulti"))
self.menuFused = QtGui.QMenu(self.menubar)
self.menuFused.setObjectName(_fromUtf8("menuFused"))
```

Ek 2’ nin devamı

```
            self.menuFile = QtGui.QMenu(self.menubar)
             self.menuFile.setObjectName(_fromUtf8("menuFile"))
             MainWindow.setMenuBar(self.menubar)
             self.statusbar = QtGui.QStatusBar(MainWindow)
             self.statusbar.setObjectName(_fromUtf8("statusbar"))
             MainWindow.setStatusBar(self.statusbar)
             self.actionOpen_Folder = QtGui.QAction(MainWindow)
             self.actionOpen_Folder.setObjectName(_fromUtf8("actionOpen_Folder"))
             self.actionOpen_Pan = QtGui.QAction(MainWindow)
             self.actionOpen_Pan.setObjectName(_fromUtf8("actionOpen_Pan"))
             self.actionOpen_Multi = QtGui.QAction(MainWindow)
             self.actionOpen_Multi.setObjectName(_fromUtf8("actionOpen_Multi"))
             self.actionRun = QtGui.QAction(MainWindow)
             self.actionRun.setObjectName(_fromUtf8("actionRun"))
             self.menuFile.addAction(self.actionOpen_Folder)
             self.menuFile.addAction(self.actionOpen_Pan)
             self.menuFile.addAction(self.actionOpen_Multi)
             self.menuFile.addAction(self.actionRun)
             self.menubar.addAction(self.menuFile.menuAction())
             self.menubar.addAction(self.menuPan.menuAction())
             self.menubar.addAction(self.menuMulti.menuAction())
             self.menubar.addAction(self.menuFused.menuAction())

             self.retranslateUi(MainWindow)
             QtCore.QObject.connect(self.actionOpen_Pan,
QtCore.SIGNAL(_fromUtf8("triggered()")), self.pannapcaz)
             QtCore.QObject.connect(self.actionOpen_Multi,
QtCore.SIGNAL(_fromUtf8("triggered()")), self.multinapcaz)
             QtCore.QObject.connect(self.actionRun,
QtCore.SIGNAL(_fromUtf8("triggered()")), self.dugme.click)
```

Ek 2’ nin devamı

```
        QtCore.QObject.connect(self.dugme, QtCore.SIGNAL(_fromUtf8("clicked()")),
self.calistir)

        QtCore.QMetaObject.connectSlotsByName(MainWindow)

    def pannapcaz(self):
        fname = QtGui.QFileDialog.getOpenFileName(self, _fromUtf8("Open file"),
_fromUtf8("*.jpg;*.tif;*.png"))

        self.panad.setText(fname)
        jpegyapgoster(fname, self.resimpan)

    def multinapcaz(self):
        fname = QtGui.QFileDialog.getOpenFileName(self, _fromUtf8("Open file"),
_fromUtf8("*.jpg;*.tif;*.png"))

        self.multiad.setText(fname)
        jpegyapgoster(fname, self.resimmulti)

    def calistir(self):
        import fonkfuse
        from os import path
        f1 = toutf8(self.panad.text())
        f2 = toutf8(self.multiad.text())
        f3 = toutf8(self.fusedad.text())
        if f3 == "":
            f3 = "output.tif"

        if path.dirname(f3) == "":
            f3 = path.join(path.dirname(f2), f3)
```

Ek 2’ nin devamı

```
            if path.splitext(f3)[1] == "":
                f3 += '.tif'

            met = (self.metodlar.currentIndex())
            if met == 0:
                fonkfuse.fusebrovey(f2, f1, f3)
            elif met == 1:
                fonkfuse.fuseihs(f2, f1, f3)
            elif met == 2:
                fonkfuse.fusewave(f2, f1, f3)

            jpegyapgoster(f3, self.resimfused)

        def retranslateUi(self, MainWindow):
            MainWindow.setWindowTitle(_translate("MainWindow", "Görüntü
Kaynaştırma", None))
            self.baslik.setHtml(_translate("MainWindow", "<!DOCTYPE HTML PUBLIC
\"-//W3C//DTD HTML 4.0//EN\" \"http://www.w3.org/TR/REC-html40/strict.dtd\">\n"
        "<html><head><meta name=\"qrichtext\" content=\"1\" /><style
type=\"text/css\">\n"
        "p, li { white-space: pre-wrap; }\n"
        "</style></head><body style=\" font-family:\'MS Shell Dlg 2\'; font-size:8.25pt; font-
weight:400; font-style:normal;\">\n"
        "<p style=\" margin-top:0px; margin-bottom:0px; margin-left:0px; margin-right:0px;
-qt-block-indent:0; text-indent:0px;\"><span style=\" font-size:12pt; font-weight:600;
color:#00007f;\">Görüntü Kaynaştırma</span></p></body></html>", None))
            self.metodlar.setItemText(0, _translate("MainWindow", "Brovey", None))
            self.metodlar.setItemText(1, _translate("MainWindow", "IHS", None))
            self.metodlar.setItemText(2, _translate("MainWindow", "Wavelet", None))
            self.resimpan.setText(_translate("MainWindow", "Resimpan", None))
            self.resimmulti.setText(_translate("MainWindow", "Resimmulti", None))
```

Ek 2’ nin devamı

```
            self.resimfused.setText(_translate("MainWindow", "Resimfused", None))
            self.dugme.setText(_translate("MainWindow", "Kaynaştır", None))
            self.pansabit.setText(_translate("MainWindow", "Pan", None))
            self.multisabit.setText(_translate("MainWindow", "Çok Bantlı", None))
            self.fusedsabit.setText(_translate("MainWindow", "Kaynaşmış", None))
            self.menuPan.setTitle(_translate("MainWindow", "Pan Görüntü", None))
            self.menuMulti.setTitle(_translate("MainWindow", "Çok Bantlı Görüntü",
None))
            self.menuFused.setTitle(_translate("MainWindow", "Kaynaşmış Görüntü",
None))
            self.menuFile.setTitle(_translate("MainWindow", "Dosya", None))
            self.actionOpen_Folder.setText(_translate("MainWindow", "Dizin Seç", None))
            self.actionOpen_Pan.setText(_translate("MainWindow", "Pan Görüntü Seç",
None))
            self.actionOpen_Multi.setText(_translate("MainWindow", "Çok Bantlı Görüntü
Seç", None))
            self.actionRun.setText(_translate("MainWindow", "Kaynaştır", None))

        if __name__ == "__main__":
            import sys
            app = QtGui.QApplication(sys.argv)
            MainWindow = QtGui.QMainWindow()
            ui = Ui_MainWindow()
            ui.setupUi(MainWindow)
            MainWindow.show()
            sys.exit(app.exec_())
```

Ek 3. Kalite Değerlendirme Arayüzü Kodları

```
# -*- coding: utf8 -*-
import os
from PyQt4 import QtCore, QtGui
from osgeo.gdal_array import LoadFile
from fonksiyonlarim import jpegyapgoster, toutf8, adkismi
import fonkmetric

try:
    _fromUtf8 = QtCore.QString.fromUtf8
except AttributeError:
    def _fromUtf8(s):
        return s

try:
    _encoding = QtGui.QApplication.UnicodeUTF8
    def _translate(context, text, disambig):
        return QtGui.QApplication.translate(context, text, disambig, _encoding)
except AttributeError:
    def _translate(context, text, disambig):
        return QtGui.QApplication.translate(context, text, disambig)


class Ui_MainWindow(QtGui.QMainWindow):

    def setupUi(self, MainWindow):
        MainWindow.setObjectName(_fromUtf8("MainWindow"))
        MainWindow.resize(756, 528)
        self.centralwidget = QtGui.QWidget(MainWindow)
        self.centralwidget.setObjectName(_fromUtf8("centralwidget"))
        self.metodergassam = QtGui.QComboBox(self.centralwidget)
```

Ek 3’ ün devamı

```
            self.metodergassam.setGeometry(QtCore.QRect(400, 130, 91, 22)
            self.metodergassam.setObjectName(_fromUtf8("metodergassam"))
             self.metodergassam.addItem(_fromUtf8(""))
             self.metodergassam.addItem(_fromUtf8(""))
             self.dugme = QtGui.QPushButton(self.centralwidget)
             self.dugme.setGeometry(QtCore.QRect(250, 160, 75, 23))
             self.dugme.setObjectName(_fromUtf8("dugme"))
             self.pansabit = QtGui.QLabel(self.centralwidget)
             self.pansabit.setGeometry(QtCore.QRect(183, 60, 61, 21))
             font = QtGui.QFont()
             font.setPointSize(12)
             self.pansabit.setFont(font)
             self.pansabit.setObjectName(_fromUtf8("pansabit"))
             self.multisabit = QtGui.QLabel(self.centralwidget)
             self.multisabit.setGeometry(QtCore.QRect(315, 60, 71, 21))
             font = QtGui.QFont()
             font.setPointSize(12)
             self.multisabit.setFont(font)
             self.multisabit.setObjectName(_fromUtf8("multisabit"))
             self.fusedsabit = QtGui.QLabel(self.centralwidget)
             self.fusedsabit.setGeometry(QtCore.QRect(480, 60, 100, 20))
             font = QtGui.QFont()
             font.setPointSize(12)
             self.fusedsabit.setFont(font)
             self.fusedsabit.setObjectName(_fromUtf8("fusedsabit"))
             self.horizontalLayoutWidget = QtGui.QWidget(self.centralwidget)
             self.horizontalLayoutWidget.setGeometry(QtCore.QRect(0, 270, 740, 160))

self.horizontalLayoutWidget.setObjectName(_fromUtf8("horizontalLayoutWidget"))
             self.resimlayout = QtGui.QHBoxLayout(self.horizontalLayoutWidget)
             self.resimlayout.setSpacing(20)
```

Ek 3’ ün devamı

```
self.resimlayout.setContentsMargins(20, -1, 20, 0)
self.resimlayout.setObjectName(_fromUtf8("resimlayout"))
self.resimpan = QtGui.QLabel(self.horizontalLayoutWidget)
self.resimpan.setObjectName(_fromUtf8("resimpan"))
self.resimlayout.addWidget(self.resimpan)
self.resimmulti = QtGui.QLabel(self.horizontalLayoutWidget)
self.resimmulti.setObjectName(_fromUtf8("resimmulti"))
self.resimlayout.addWidget(self.resimmulti)
self.resimfused = QtGui.QLabel(self.horizontalLayoutWidget)
self.resimfused.setObjectName(_fromUtf8("resimfused"))
self.resimlayout.addWidget(self.resimfused)
self.metodssimspat = QtGui.QComboBox(self.centralwidget)
self.metodssimspat.setGeometry(QtCore.QRect(520, 130, 91, 22))
self.metodssimspat.setObjectName(_fromUtf8("metodssimspat"))
self.metodssimspat.addItem(_fromUtf8(""))
self.metodssimspat.addItem(_fromUtf8(""))
self.sonucwidget = QtGui.QWidget(self.centralwidget)
self.sonucwidget.setGeometry(QtCore.QRect(390, 160, 341, 80))
self.sonucwidget.setObjectName(_fromUtf8("sonucwidget"))
self.sonuclayout = QtGui.QHBoxLayout(self.sonucwidget)
self.sonuclayout.setMargin(0)
self.sonuclayout.setObjectName(_fromUtf8("sonuclayout"))
self.sonucergassam = QtGui.QTextEdit(self.sonucwidget)
self.sonucergassam.setObjectName(_fromUtf8("sonucergassam"))
self.sonuclayout.addWidget(self.sonucergassam)
self.sonucssimspat = QtGui.QTextEdit(self.sonucwidget)
self.sonucssimspat.setObjectName(_fromUtf8("sonucssimspat"))
self.sonuclayout.addWidget(self.sonucssimspat)
self.sonucccccccc = QtGui.QTextEdit(self.sonucwidget)
self.sonucccccccc.setObjectName(_fromUtf8("sonucccccccc"))
```

Ek 3’ ün devamı

```
            self.sonuclayout.addWidget(self.sonucccccc)
            self.metodccccc = QtGui.QComboBox(self.centralwidget)
            self.metodccccc.setGeometry(QtCore.QRect(630, 130, 91, 22))
            self.metodccccc.setObjectName(_fromUtf8("metodccccc"))
            self.metodccccc.addItem(_fromUtf8(""))
            self.metodccccc.addItem(_fromUtf8(""))
            self.horizontalLayoutWidget_3 = QtGui.QWidget(self.centralwidget)
            self.horizontalLayoutWidget_3.setGeometry(QtCore.QRect(100, 60, 500, 80))

self.horizontalLayoutWidget_3.setObjectName(_fromUtf8("horizontalLayoutWidget_3"))
            self.isimlayout = QtGui.QHBoxLayout(self.horizontalLayoutWidget_3)
            self.isimlayout.setSpacing(20)
            self.isimlayout.setContentsMargins(20, -1, 0, -1)
            self.isimlayout.setObjectName(_fromUtf8("isimlayout"))
            self.panad = QtGui.QLineEdit(self.horizontalLayoutWidget_3)
            self.panad.setObjectName(_fromUtf8("panad"))
            self.isimlayout.addWidget(self.panad)
            self.multiad = QtGui.QLineEdit(self.horizontalLayoutWidget_3)
            self.multiad.setObjectName(_fromUtf8("multiad"))
            self.isimlayout.addWidget(self.multiad)
            self.fusedad = QtGui.QLineEdit(self.horizontalLayoutWidget_3)
            self.fusedad.setObjectName(_fromUtf8("fusedad"))
            self.isimlayout.addWidget(self.fusedad)
            self.horizontalLayoutWidget_4 = QtGui.QWidget(self.centralwidget)
            self.horizontalLayoutWidget_4.setGeometry(QtCore.QRect(390,     160,     341,
180))

self.horizontalLayoutWidget_4.setObjectName(_fromUtf8("horizontalLayoutWidget_4"))
            self.bestooolayout = QtGui.QHBoxLayout(self.horizontalLayoutWidget_4)
            self.bestooolayout.setSpacing(30)
            self.bestooolayout.setContentsMargins(20, 2, 20, 2)
```

Ek 3’ ün devamı

```
self.bestooolayout.setObjectName(_fromUtf8("bestooolayout"))
self.bestoilk = QtGui.QLineEdit(self.horizontalLayoutWidget_4)
self.bestoilk.setObjectName(_fromUtf8("bestoilk"))
 self.bestooolayout.addWidget(self.bestoilk)
 self.bestoiki = QtGui.QLineEdit(self.horizontalLayoutWidget_4)
 self.bestoiki.setObjectName(_fromUtf8("bestoiki"))
 self.bestooolayout.addWidget(self.bestoiki)
 self.bestouc = QtGui.QLineEdit(self.horizontalLayoutWidget_4)
 self.bestouc.setObjectName(_fromUtf8("fusedad"))
 self.bestooolayout.addWidget(self.bestouc)
 self.bestoilk.setText(_fromUtf8("En iyi: 0"))
 self.bestoiki.setText(_fromUtf8("En iyi: 1"))
 self.bestouc.setText(_fromUtf8("En iyi: 0"))

 self.baslik = QtGui.QTextBrowser(self.centralwidget)
 self.baslik.setGeometry(QtCore.QRect(240, 10, 201, 31))
 self.baslik.setObjectName(_fromUtf8("baslik"))
 MainWindow.setCentralWidget(self.centralwidget)
 self.statusbar = QtGui.QStatusBar(MainWindow)
 self.statusbar.setObjectName(_fromUtf8("statusbar"))
 MainWindow.setStatusBar(self.statusbar)
 self.menubar = QtGui.QMenuBar(MainWindow)
 self.menubar.setGeometry(QtCore.QRect(0, 0, 756, 21))
 self.menubar.setObjectName(_fromUtf8("menubar"))
 self.menuFused_Image = QtGui.QMenu(self.menubar)
 self.menuFused_Image.setObjectName(_fromUtf8("menuFused_Image"))
 self.menuChoose_Folder = QtGui.QMenu(self.menubar)
 self.menuChoose_Folder.setObjectName(_fromUtf8("menuChoose_Folder"))
 self.menuImage_Viewer = QtGui.QMenu(self.menubar)
 self.menuImage_Viewer.setObjectName(_fromUtf8("menuImage_Viewer"))
 MainWindow.setMenuBar(self.menubar)
```

```
        self.actionOpen_Folder = QtGui.QAction(MainWindow)
        self.actionOpen_Folder.setObjectName(_fromUtf8("actionOpen_Folder"))
        self.actionOpen_Pan = QtGui.QAction(MainWindow)
        self.actionOpen_Pan.setObjectName(_fromUtf8("actionOpen_Pan"))
        self.actionOpen_Multi = QtGui.QAction(MainWindow)
        self.actionOpen_Multi.setObjectName(_fromUtf8("actionOpen_Multi"))
        self.actionOpen_Fused = QtGui.QAction(MainWindow)
        self.actionOpen_Fused.setObjectName(_fromUtf8("actionOpen_Fused"))
        self.actionRun = QtGui.QAction(MainWindow)
        self.actionRun.setObjectName(_fromUtf8("actionRun"))
        self.menuFused_Image.addAction(self.actionOpen_Folder)
        self.menuFused_Image.addAction(self.actionOpen_Pan)
        self.menuFused_Image.addAction(self.actionOpen_Multi)
        self.menuFused_Image.addAction(self.actionOpen_Fused)
        self.menuFused_Image.addAction(self.actionRun)
        self.menubar.addAction(self.menuFused_Image.menuAction())
        self.menubar.addAction(self.menuImage_Viewer.menuAction())
        self.dirnamee = QtCore.QString(".")

        self.retranslateUi(MainWindow)
        QtCore.QObject.connect(self.actionRun,
QtCore.SIGNAL(_fromUtf8("triggered()")), self.dugme.click)
        QtCore.QObject.connect(self.actionOpen_Folder,
QtCore.SIGNAL(_fromUtf8("triggered()")), self.dizinegit)
        QtCore.QObject.connect(self.actionOpen_Pan,
QtCore.SIGNAL(_fromUtf8("triggered()")), self.pannapcaz)
        QtCore.QObject.connect(self.actionOpen_Multi,
QtCore.SIGNAL(_fromUtf8("triggered()")), self.multinapcaz)
        QtCore.QObject.connect(self.actionOpen_Fused,
QtCore.SIGNAL(_fromUtf8("triggered()")), self.fusednapcaz)
        QtCore.QObject.connect(self.dugme, QtCore.SIGNAL(_fromUtf8("clicked()")),
self.calistir)
        QtCore.QMetaObject.connectSlotsByName(MainWindow)
```

Ek 3’ ün devamı

```
        def dizinegit(self, diro="C:/Users/Ekem/Dropbox/ekrem deniz/pycharmicin"):

            self.dirnamee = QtGui.QFileDialog.getExistingDirectory(self, _fromUtf8("Open Folder"), _fromUtf8(diro))
            os.chdir(toutf8(self.dirnamee))

        def pannapcaz(self):
            os.chdir(toutf8(self.dirnamee))
            fname = QtGui.QFileDialog.getOpenFileName(self, _fromUtf8("Open file"), _fromUtf8("*.jpg;*.tif;*.png"))

            self.panad.setText(adkismi(fname))
            self.panad.dirnamee = self.dirnamee
            jpegyapgoster(fname, self.resimpan)

        def multinapcaz(self):
            os.chdir(toutf8(self.dirnamee))
            fname = QtGui.QFileDialog.getOpenFileName(self, _fromUtf8("Open file"), _fromUtf8("*.jpg;*.tif;*.png"))

            self.multiad.setText(adkismi(fname))
            jpegyapgoster(fname, self.resimmulti)

        def fusednapcaz(self):
            os.chdir(toutf8(self.dirnamee))
            fname = QtGui.QFileDialog.getOpenFileName(self, _fromUtf8("Open file"), _fromUtf8("*.jpg;*.tif;*.png"))

            self.fusedad.setText(adkismi(fname))
            jpegyapgoster(fname, self.resimfused)
```

Ek 3’ ün devamı

```
    def calistir(self):
        os.chdir(toutf8(self.dirnamee))
        pan = LoadFile(toutf8(self.panad.text()))
        imulti = LoadFile(toutf8(self.multiad.text()))
        f3 = toutf8(self.fusedad.text())
        if f3 == "":
            f3 = "output.tif"

#       if path.dirname(f3) == "":
#           f3 = path.join(toutf8(self.dirnamee), f3)

        if os.path.splitext(f3)[1] == "":
            f3 += '.tif'

        ifused = LoadFile(f3)

        mete = (self.metodergassam.currentIndex())
        mets = (self.metodssimspat.currentIndex())
        metc = (self.metodccccccc.currentIndex())



        if mete == 0:
            self.sonucergassam.setText(_fromUtf8(str(fonkmetric.ergas(imulti, ifused))))
        elif mete == 1:
            self.sonucergassam.setText(_fromUtf8(str(fonkmetric.sam(imulti, ifused))))

        if mets == 0:
            self.sonucssimspat.setText(_fromUtf8(str(fonkmetric.ssim(pan, ifused))))
        elif mets == 1:
```

Ek 3’ ün devamı

```
                self.sonucssimspat.setText(_fromUtf8(str(fonkmetric.spatmetrik(ifused,
pan))))

            if metc == 0:

               self.sonucccccccc.setText(_fromUtf8(str(fonkmetric.cc(imulti, ifused))))
            elif metc == 1:
               self.sonucccccccc.setText(_fromUtf8(str(fonkmetric.cccc(imulti, ifused))))

         def retranslateUi(self, MainWindow):
            MainWindow.setWindowTitle(_translate("MainWindow", "Kalite
Değerlendirme", None))
            self.baslik.setHtml(_translate("MainWindow", "<!DOCTYPE HTML PUBLIC
\"-//W3C//DTD HTML 4.0//EN\" \"http://www.w3.org/TR/REC-html40/strict.dtd\">\n"
        "<html><head><meta name=\"qrichtext\" content=\"1\" /><style
type=\"text/css\">\n"
        "p, li { white-space: pre-wrap; }\n"
        "</style></head><body style=\" font-family:\'MS Shell Dlg 2\'; font-size:8.25pt; font-
weight:400; font-style:normal;\">\n"
        "<p style=\" margin-top:0px; margin-bottom:0px; margin-left:0px; margin-right:0px;
-qt-block-indent:0; text-indent:0px;\"><span style=\" font-size:12pt; font-weight:600;
color:#00007f;\">Kalite Değerlendirme</span></p></body></html>", None))
            self.metodergassam.setItemText(0, _translate("MainWindow", "ERGAS",
None))
            self.metodergassam.setItemText(1, _translate("MainWindow", "SAM", None))
            self.dugme.setText(_translate("MainWindow", "Hesapla", None))
            self.pansabit.setText(_translate("MainWindow", "Pan", None))
            self.multisabit.setText(_translate("MainWindow", "Çok Bantlı", None))
            self.fusedsabit.setText(_translate("MainWindow", "Kaynaşmış", None))
            self.resimpan.setText(_translate("MainWindow", "Pan Figure", None))
            self.resimmulti.setText(_translate("MainWindow", "Multi Figure", None))
```

Ek 3’ ün devamı

```
            self.resimfused.setText(_translate("MainWindow", "Fused Figure", None))
            self.metodssimspat.setItemText(0, _translate("MainWindow", "SSIM", None))
            self.metodssimspat.setItemText(1, _translate("MainWindow", "Spat", None))
            self.metodccccccc.setItemText(0, _translate("MainWindow", "CC", None))
            self.metodccccccc.setItemText(1, _translate("MainWindow", "CCCC", None))
            self.panad.setText(_translate("MainWindow", "", None))
            self.multiad.setText(_translate("MainWindow", "", None))
            self.fusedad.setText(_translate("MainWindow", "", None))
            self.menuFused_Image.setTitle(_translate("MainWindow", "Dosya", None))
            self.menuImage_Viewer.setTitle(_translate("MainWindow", "Görüntü Açma",
None))
            self.actionOpen_Folder.setText(_translate("MainWindow", "Dizin Seç", None))
            self.actionOpen_Pan.setText(_translate("MainWindow", "Pan Görüntü Seç",
None))
            self.actionOpen_Multi.setText(_translate("MainWindow", "Çok Bantlı Görüntü
Seç", None))
            self.actionOpen_Fused.setText(_translate("MainWindow", "Kaynaşmış Görüntü
Seç", None))
            self.actionRun.setText(_translate("MainWindow", "Hesapla", None))


    if __name__ == "__main__":
        import sys
        app = QtGui.QApplication(sys.argv)
        MainWindow = QtGui.QMainWindow()
        ui = Ui_MainWindow()
        ui.setupUi(MainWindow)
        MainWindow.show()
        sys.exit(app.exec_())
```

Ek 4. Sınıflandırma Arayüzü Kodları

```
# -*- coding: utf-8 -*-
import os
from PyQt4 import QtCore, QtGui
from osgeo.gdal_array import SaveArray
from fonksiyonlarim import toutf8, jpegyapgoster, adkismi,isodata,floati
from siniflar import Goruntu

try:
    _fromUtf8 = QtCore.QString.fromUtf8
except AttributeError:
    def _fromUtf8(s):
        return s

try:
    _encoding = QtGui.QApplication.UnicodeUTF8
    def _translate(context, text, disambig):
        return QtGui.QApplication.translate(context, text, disambig, _encoding)
except AttributeError:
    def _translate(context, text, disambig):
        return QtGui.QApplication.translate(context, text, disambig)

class Ui_MainWindow(QtGui.QMainWindow):
    def setupUi(self, MainWindow):
        MainWindow.setObjectName(_fromUtf8("MainWindow"))
        MainWindow.resize(713, 558)
        MainWindow.setAutoFillBackground(False)
        MainWindow.setDockNestingEnabled(False)
        self.centralwidget = QtGui.QWidget(MainWindow)
        self.centralwidget.setObjectName(_fromUtf8("centralwidget"))
        self.dugme = QtGui.QPushButton(self.centralwidget)
        self.dugme.setGeometry(QtCore.QRect(620, 80, 81, 23))
```

Ek 4’ ün devamı

```
self.dugme.setObjectName(_fromUtf8("dugme"))
self.metodlar = QtGui.QComboBox(self.centralwidget)
self.metodlar.setGeometry(QtCore.QRect(490, 80, 101, 22))
self.metodlar.setObjectName(_fromUtf8("metodlar"))
self.metodlar.addItem(_fromUtf8(""))
self.metodlar.addItem(_fromUtf8(""))
self.metodlar.addItem(_fromUtf8(""))
self.adilk = QtGui.QLineEdit(self.centralwidget)
self.adilk.setGeometry(QtCore.QRect(180, 80, 113, 20))
self.adilk.setObjectName(_fromUtf8("adilk"))
self.sabitilk = QtGui.QLabel(self.centralwidget)
self.sabitilk.setGeometry(QtCore.QRect(90, 80, 111, 21))
font = QtGui.QFont()
font.setPointSize(14)
self.sabitilk.setFont(font)
self.sabitilk.setObjectName(_fromUtf8("sabitilk"))
self.baslik = QtGui.QLineEdit(self.centralwidget)
self.baslik.setGeometry(QtCore.QRect(220, 30, 251, 31))
font = QtGui.QFont()
font.setPointSize(14)
self.baslik.setFont(font)
self.baslik.setObjectName(_fromUtf8("baslik"))
self.sabitmetod = QtGui.QLabel(self.centralwidget)
self.sabitmetod.setGeometry(QtCore.QRect(370, 80, 81, 21))
font = QtGui.QFont()
font.setPointSize(14)
self.sabitmetod.setFont(font)
self.sabitmetod.setObjectName(_fromUtf8("sabitmetod"))
self.sabitson = QtGui.QLabel(self.centralwidget)
self.sabitson.setGeometry(QtCore.QRect(30, 110, 131, 21))
font = QtGui.QFont()
```

Ek 4’ ün devamı

```
font.setPointSize(14)
 self.sabitson.setFont(font)
 self.sabitson.setObjectName(_fromUtf8("sabitson"))
 self.adson = QtGui.QLineEdit(self.centralwidget)
 self.adson.setGeometry(QtCore.QRect(180, 110, 113, 20))
 self.adson.setObjectName(_fromUtf8("adson"))
 self.gridLayoutWidget = QtGui.QWidget(self.centralwidget)
 self.gridLayoutWidget.setGeometry(QtCore.QRect(430, 110, 271, 152))
 self.gridLayoutWidget.setObjectName(_fromUtf8("gridLayoutWidget"))
 self.parametreler = QtGui.QGridLayout(self.gridLayoutWidget)
 self.parametreler.setContentsMargins(0, -1, -1, -1)
 self.parametreler.setObjectName(_fromUtf8("parametreler"))
 self.deger2 = QtGui.QLineEdit(self.gridLayoutWidget)
 self.deger2.setObjectName(_fromUtf8("deger2"))
 self.parametreler.addWidget(self.deger2, 1, 1, 1, 1)
 self.par6 = QtGui.QLineEdit(self.gridLayoutWidget)
 self.par6.setObjectName(_fromUtf8("par6"))
 self.parametreler.addWidget(self.par6, 5, 0, 1, 1)
 self.deger3 = QtGui.QLineEdit(self.gridLayoutWidget)
 self.deger3.setObjectName(_fromUtf8("deger3"))
 self.parametreler.addWidget(self.deger3, 2, 1, 1, 1)
 self.par5 = QtGui.QLineEdit(self.gridLayoutWidget)
 self.par5.setObjectName(_fromUtf8("par5"))
 self.parametreler.addWidget(self.par5, 4, 0, 1, 1)
 self.deger4 = QtGui.QLineEdit(self.gridLayoutWidget)
 self.deger4.setObjectName(_fromUtf8("deger4"))
 self.parametreler.addWidget(self.deger4, 3, 1, 1, 1)
 self.deger5 = QtGui.QLineEdit(self.gridLayoutWidget)
 self.deger5.setObjectName(_fromUtf8("deger5"))
 self.parametreler.addWidget(self.deger5, 4, 1, 1, 1)
```

Ek 4’ ün devamı

```
            self.par2 = QtGui.QLineEdit(self.gridLayoutWidget)
             self.par2.setObjectName(_fromUtf8("par2"))
             self.parametreler.addWidget(self.par2, 1, 0, 1, 1)
             self.par3 = QtGui.QLineEdit(self.gridLayoutWidget)
             self.par3.setObjectName(_fromUtf8("par3"))
             self.parametreler.addWidget(self.par3, 2, 0, 1, 1)
             self.par4 = QtGui.QLineEdit(self.gridLayoutWidget)
             self.par4.setObjectName(_fromUtf8("par4"))
             self.parametreler.addWidget(self.par4, 3, 0, 1, 1)
             self.par1 = QtGui.QLineEdit(self.gridLayoutWidget)
             self.par1.setObjectName(_fromUtf8("par1"))
             self.parametreler.addWidget(self.par1, 0, 0, 1, 1)
             self.deger6 = QtGui.QLineEdit(self.gridLayoutWidget)
             self.deger6.setObjectName(_fromUtf8("deger6"))
             self.parametreler.addWidget(self.deger6, 5, 1, 1, 1)
             self.deger1 = QtGui.QLineEdit(self.gridLayoutWidget)
             self.deger1.setObjectName(_fromUtf8("deger1"))
             self.parametreler.addWidget(self.deger1, 0, 1, 1, 1)
             self.horizontalLayoutWidget = QtGui.QWidget(self.centralwidget)
             self.horizontalLayoutWidget.setGeometry(QtCore.QRect(10, 270, 671, 251))

self.horizontalLayoutWidget.setObjectName(_fromUtf8("horizontalLayoutWidget"))
             self.resimler = QtGui.QHBoxLayout(self.horizontalLayoutWidget)
             self.resimler.setContentsMargins(1, 1, 0, 1)
             self.resimler.setObjectName(_fromUtf8("resimler"))
             self.goruntuilk = QtGui.QLabel(self.horizontalLayoutWidget)
             self.goruntuilk.setObjectName(_fromUtf8("goruntuilk"))
             self.resimler.addWidget(self.goruntuilk)
             self.goruntuson = QtGui.QLabel(self.horizontalLayoutWidget)
             self.goruntuson.setObjectName(_fromUtf8("goruntuson"))
             self.resimler.addWidget(self.goruntuson)
```

Ek 4’ ün devamı

```
MainWindow.setCentralWidget(self.centralwidget)
self.menubar = QtGui.QMenuBar(MainWindow)
self.menubar.setGeometry(QtCore.QRect(0, 0, 713, 21))
self.menubar.setObjectName(_fromUtf8("menubar"))
self.menuFile = QtGui.QMenu(self.menubar)
self.menuFile.setObjectName(_fromUtf8("menuFile"))
self.menuGoruntu = QtGui.QMenu(self.menubar)
self.menuGoruntu.setObjectName(_fromUtf8("menuGoruntu"))
self.menuSiniflandirilmis = QtGui.QMenu(self.menubar)
self.menuSiniflandirilmis.setObjectName(_fromUtf8("menuSiniflandirilmis"))
self.menuChoose_Folder = QtGui.QMenu(self.menubar)
self.menuChoose_Folder.setObjectName(_fromUtf8("menuChoose_Folder"))
MainWindow.setMenuBar(self.menubar)
self.statusbar = QtGui.QStatusBar(MainWindow)
self.statusbar.setObjectName(_fromUtf8("statusbar"))
MainWindow.setStatusBar(self.statusbar)
self.actionOpen_Folder = QtGui.QAction(MainWindow)
self.actionOpen_Folder.setObjectName(_fromUtf8("actionOpen_Folder"))
self.actionSec = QtGui.QAction(MainWindow)
self.actionSec.setObjectName(_fromUtf8("actionSec"))
self.actionSiniflandir = QtGui.QAction(MainWindow)
self.actionSiniflandir.setObjectName(_fromUtf8("actionSiniflandir"))
self.menuFile.addAction(self.actionOpen_Folder)
self.menuFile.addAction(self.actionSec)
self.menuFile.addAction(self.actionSiniflandir)
self.menubar.addAction(self.menuFile.menuAction())
self.menubar.addAction(self.menuGoruntu.menuAction())
self.menubar.addAction(self.menuSiniflandirilmis.menuAction())
self.dirnamee = QtCore.QString(".")
```

Ek 4’ ün devamı

```
        self.retranslateUi(MainWindow)
        QtCore.QObject.connect(self.dugme, QtCore.SIGNAL(_fromUtf8("clicked()")),
self.calistir)
        QtCore.QMetaObject.connectSlotsByName(MainWindow)
        QtCore.QObject.connect(self.actionSiniflandir,
QtCore.SIGNAL(_fromUtf8("triggered()")), self.dugme.click)
        QtCore.QObject.connect(self.actionOpen_Folder,
QtCore.SIGNAL(_fromUtf8("triggered()")), self.dizinegit)
        QtCore.QObject.connect(self.actionSec,
QtCore.SIGNAL(_fromUtf8("triggered()")), self.multinapcaz)
        QtCore.QObject.connect(self.dugme, QtCore.SIGNAL(_fromUtf8("clicked()")),
self.calistir)
        QtCore.QMetaObject.connectSlotsByName(MainWindow)

    def dizinegit(self, diro="C:/Users/Ekem/Dropbox/ekrem deniz/pycharmicin"):
        self.dirnamee = QtGui.QFileDialog.getExistingDirectory(self, _fromUtf8("Open
Folder"), _fromUtf8(diro))
        os.chdir(toutf8(self.dirnamee))

    def multinapcaz(self):
        os.chdir(toutf8(self.dirnamee))
        fname = QtGui.QFileDialog.getOpenFileName(self, _fromUtf8("Open file"),
_fromUtf8("*.jpg;*.tif;*.png;*.img"))

        self.adilk.setText(adkismi(fname))
        jpegyapgoster(fname, self.goruntuilk)
```

Ek 4’ ün devamı

```
        def calistir(self):
            a=Goruntu(ad=toutf8(self.adilk.text()), dizin=toutf8(self.dirnamee))
            b=floati(isodata(a.veri,
K=2,I=100,P=4,THETA_M=10,THETA_S=1,THETA_C=20,THETA_O=0.03))
            b /= b.max()
            SaveArray((b*255).astype('uint8'), toutf8(self.adson.text()))
            jpegyapgoster(qs=self.adson.text(),hangi=self.goruntuson)

        def retranslateUi(self, MainWindow):
            MainWindow.setWindowTitle(_translate("MainWindow",                "Image
Classification", None))
            self.dugme.setText(_translate("MainWindow", "Sınıflandır", None))
            self.metodlar.setItemText(0, _translate("MainWindow", "ISODATA", None))
            self.metodlar.setItemText(1, _translate("MainWindow", "K-Means", None))
            self.metodlar.setItemText(2, _translate("MainWindow", "SVM", None))
            self.adilk.setText(_translate("MainWindow", "m.img", None))
            self.sabitilk.setText(_translate("MainWindow", "Görüntü:", None))
            self.baslik.setText(_translate("MainWindow", "      SINIFLANDIRMA", None))
            self.sabitmetod.setText(_translate("MainWindow", "Yöntem:", None))
            self.sabitson.setText(_translate("MainWindow", "Sonuç görüntü:", None))
            self.adson.setText(_translate("MainWindow", "son.png", None))
            self.deger2.setText(_translate("MainWindow", "100", None))
            self.par6.setText(_translate("MainWindow", "iter. değişklik eşiği", None))
            self.deger3.setText(_translate("MainWindow", "4", None))
            self.par5.setText(_translate("MainWindow", "Uzaklık eşiği", None))
            self.deger4.setText(_translate("MainWindow", "1", None))
            self.deger5.setText(_translate("MainWindow", "20", None))
            self.par2.setText(_translate("MainWindow", "Maksimum iterasyon:", None))
            self.par3.setText(_translate("MainWindow", "Maksimum birl. yığın", None))
```

Ek 4’ ün devamı

```
            self.par4.setText(_translate("MainWindow", "Standard sapma eşiği", None))
            self.par1.setText(_translate("MainWindow", "Sınıf sayısı:", None))
            self.deger6.setText(_translate("MainWindow", "0.03", None))
            self.deger1.setText(_translate("MainWindow", "2", None))
            self.goruntuilk.setText(_translate("MainWindow", "Orijinal görüntü", None))
            self.goruntuson.setText(_translate("MainWindow",  "Sınıflandırılmış  görüntü",
None))
            self.menuFile.setTitle(_translate("MainWindow", "File", None))
            self.menuGoruntu.setTitle(_translate("MainWindow", "Görüntü", None))
            self.menuSiniflandirilmis.setTitle(_translate("MainWindow",   "Sınıflandırılmış
Görüntü", None))
            self.actionSec.setText(_translate("MainWindow", "Görüntüyü Seç", None))
            self.actionSiniflandir.setText(_translate("MainWindow", "Sınıflandır", None))
            self.actionOpen_Folder.setText(_translate("MainWindow", "Dizin Seç", None))


    if __name__ == "__main__":
        import sys
        app = QtGui.QApplication(sys.argv)
        MainWindow = QtGui.QMainWindow()
        ui = Ui_MainWindow()
        ui.setupUi(MainWindow)
        MainWindow.show()
        sys.exit(app.exec_())
```

ÖZGEÇMİŞ

13.06.1989 yılında Trabzon’da doğdu. İlk öğrenimini Kanuni İlköğretim Okulu’nda lise öğrenimini de Yunus Emre Lisesi’nde gördü. 2007 yılında Karadeniz Teknik Üniversitesi Harita Mühendisliği bölümünü kazandı. 1 yılı İngilizce hazırlık olmak üzere 5 yıllık lisans eğitimini 2012 yılında tamamladı. Aynı yıl Karadeniz Teknik Üniversitesinde yüksek lisans eğitimine başladı. Aynı yıl Artvin Çoruh Üniversitesi Harita Mühendisliği Anabilim Dalı'na Öğretim üyesi yetiştirme programı (ÖYP) kadrosundan araştırma görevlisi olarak atandı. Bu kapsamda 6 ay Akdeniz Üniversitesi’nde İngilizce eğitimi gördü. Daha sonra 35. Madde görevlendirmesi ile Karadeniz Teknik Üniversitesi Harita Mühendisliği Bölümü’ne görevlendirilmesi yapıldı. Halen bu üniversite de görevine ve yüksek lisans eğitimine devam etmektedir.