

T.C.
BEYKENT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI
BİLGİSAYAR MÜHENDİSLİĞİ BİLİM DALI

**BİLGİSAYAR GÖRÜSÜ TEKNİKLERİ İLE
GÖKYÜZÜ GÖRÜNTÜLERİNİN ANALİZİ VE
BULUT HAREKETLERİNİN İZLENMESİ**

Yüksek Lisans Tezi

Tezi Hazırlayan:
Recep ALTIN

İSTANBUL, 2018

T.C.
BEYKENT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI
BİLGİSAYAR MÜHENDİSLİĞİ BİLİM DALI

**BİLGİSAYAR GÖRÜŞÜ TEKNİKLERİ İLE
GÖKYÜZÜ GÖRÜNTÜLERİNİN ANALİZİ VE
BULUT HAREKETLERİNİN İZLENMESİ**

Yüksek Lisans Tezi

Tezi Hazırlayan:
Recep ALTIN

Öğrenci No:
160820810

Danışman:
Dr. Ediz ŞAYKOL

İSTANBUL, 2018

YEMİN METNİ

Yüksek Lisans Tezi olarak sunduğum “**Bilgisayar Görüşü Teknikleri İle Gökyüzü Görüntülerinin Analizi Ve Bulut Hareketlerinin İzlenmesi**” başlıklı bu çalışmanın, bilimsel ahlak ve geleneklere uygun şekilde tarafımdan yazıldığını, yararlandığım eserlerin tamamının kaynaklarda gösterildiğini ve çalışmamın içinde kullanıldıkları her yerde bunlara atıf yapıldığını belirtir ve bunu onurumla doğrularım. 11/06/2018

Recep ALTIN

T.C.
BEYKENT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

YÜKSEK LİSANS TEZ SAVUNMA SINAVI SONUÇ TUTANAĞI

Beykent Üniversitesi
Fen Bilimleri Enstitüsü Müdürlüğü'ne,

Aşağıda tez adı belirtilen yüksek lisans öğrencisi ..160820810... no'lu
.....REZEP ALTIN.....'in ..11/6/18 tarihinde yapılan tez savunma sınavı¹
sonucunda 45 dakika süreyle sunduğu ve savunduğu tezi hakkında² oybirliğiyle,
.....KABUL..... kararı verilmiştir.

Bilgilerinize saygılarımızla arz ederiz.

Anabilim Dalı : BİLGİSAYAR MÜHENDİSLİĞİ

Programı : BİLGİSAYAR MÜHENDİSLİĞİ

Tez Başlığı³ : Bilgiyağın Görsel Teknikleri ile Görsel Görsel Görsel
Analizi ve Bulun Hareketlerinin İzlenesi

Tez Sınav Jürisi

Öğretim Üyesi

İmza

Danışman

: Dr. Ediz ŞAYKOL

Üye

: Dr. Atıf YILMAZ

Üye

: Prof. Dr. Gülşen DİLİTAZARÖĞÜ

¹Jüri üyeleri, söz konusu tezin kendilerine teslim edildiği tarihten itibaren en geç bir ay içinde toplanarak öğrenciyi tezsınavına alır. Tez savunma sınav süresi en az 45, en çok 90 dakikadır. Jüri üyeleri, sınav öncesi yapılacak toplantıda, kendilerinin danışmanından bir üyeyi başkan seçer. Tezsınavı, tez çalışmasının sunulması ve bunu izleyen soru-cevap bölümünden oluşur. Tezsınavı, öğretim elemanları, lisansüstü öğrenciler ve alanın uzmanlarından oluşan dinleyicilerin katılımına açık ortamlarda gerçekleştirilir. Belirlenengündeyapılamayan jüri toplantısı, katılanların hazırladığı bir tutanakla enstitü yönetimine bildirilir. Bu durumda, jüri en geç on beş gün içinde toplanarak adayın tez savunmasını alır. (05 Ağustos 2017 tarihli 30145 sayılı Resmi Gazete'de Yayınlanan Değişiklik-Madde 29-3)

²Tezsınavının tamamlanmasından sonra jüri, tez hakkında salt çoğunlukla "kabul", "düzeltme" veya "ret" karar verir. Jüri başkanı, jüri üyelerince imzalanmış karar tutanağını, tezsınavını izleyen üç gün içinde ilgili enstitü yönetimine teslim eder. Tezi hakkında düzeltme kararı verilen öğrenci en geç üç ay içinde gerekli düzeltmeleri yaparak ve birinci fıkradaki süre gereğini aynı jüri önünde yeniden savunur. Süresi içerisinde "düzeltme" savunmasını yapmayan öğrencinin enstitü ile ilişkisi kesilir. (Beykent Üniversitesi Lisansüstü Eğitim ve Öğretim Yönetmeliği-Madde 29-4)

³İleride doğabilecek aksaklıkların engellenmesi için tezin başlığını yazılması gerekmektedir.

Adı ve Soyadı : Recep ALTIN
Danışmanı : Dr. Ediz ŞAYKOL
Türü ve Tarihi : Yüksek Lisans, 2018
Alanı : Bilgisayar Mühendisliği
Anahtar Kelimeler : Bulut Tanıma, Bulut Hareket Takibi

ÖZ

BİLGİSAYAR GÖRÜŞÜ TEKNİKLERİ İLE GÖKYÜZÜ GÖRÜNTÜLERİNİN ANALİZİ VE BULUT HAREKETLERİNİN İZLENMESİ

Günümüzde nesne tanıma, bilgisayar görüşü alanında oldukça popüler ve birçok otomatize sistemin temelini teşkil eder durumdadır. Bu tez çalışmasında gökyüzü görüntülerinin analiz edilerek bulutların tespit edilebilirliği, tespit edilecek bulutların hareketlerinin takibe alınabilirliği ve bulutların hareket hızlarına göre güneş ışınlarını ne kadar süre sonra kesebileceğinin tahmin edilebilirliği incelenmiştir.

Tez çalışmasının sonucunda olası güneş kesintileri için bir erken uyarı sisteminin geliştirilebileceği ve özellikle solar enerji tesislerinde kullanım alanı bulabileceği görülmüştür.

Name and Surname : Recep ALTIN
Supervisor : Dr. Ediz ŞAYKOL
Degree and Date : Master, 2018
Major : Computer Engineering
Keywords : Cloud Detection, Cloud Motion Tracking

ABSTRACT

SKY IMAGE ANALYSIS AND CLOUD MOVEMENT TRACKING WITH COMPUTER VISION TECHNIQUES

Today, object recognition is very popular in the field of computer vision and is the basis of many automatized systems. In this study, it has been studied, through the analysis of the images taken from the sky, the detectability of the clouds, the trackability of the clouds to be detected and the estimation of how long it will take the clouds to block the sun light according to the speed of the clouds.

It has been observed, at the end of the study, that an early-warning system can be developed against possible solar interruptions, and especially, this will be widely in use in solar energy plants.

İÇİNDEKİLER

Sayfa No.

ÖZ

ABSTRACT

ŞEKİLLER LİSTESİ	ii
KISALTMALAR	iii
1. GİRİŞ.....	1
2. BENZER ÇALIŞMALAR	2
3. GÖKYÜZÜ KARAKTERİSTİĞİ VE GÖRÜNTÜLEME DONANIMI	3
3.1. Gökyüzü Karakteristiği	3
3.2. Görüntüleme Donanımı.....	4
4. GÖKYÜZÜ GÖRÜNTÜLERİNİN İŞLENMESİ.....	11
4.1. Görüntü Aktarımı	11
4.2. Gökyüzü Bölgesinin Eleminasyonu	12
4.2.1. Görüntünün Renk Kanallarına Ayırıştırılması	12
4.2.2. Eşikleme ve Morfolojik Operasyonlar	17
4.2.3. Gökyüzü Rekonstrüksiyonu	21
4.3. Bulutların Tanımlanması.....	23
4.3.1. Çevritleme	23
4.3.2. Bulutların Ayırıştırılması	25
4.4. Bulutların Takibi	29
4.4.1. Çerçeveler Arası Bulut Eşleme	29
4.4.2. Bulutların Hareket Yönlerinin ve Hızlarının Belirlenmesi	32
4.5. Bulutların Güneş Kapatma Analizi	39
4.5.1. Güneş Konumunun ve Rotasının Hesaplanması	39
4.5.2. Bulut ve Güneş Çarpışmalarının Belirlenmesi.....	42
5. UYGULAMA GELİŞTİRME	46
5.1. Uygulamanın Sistem Gereksinimleri	46
5.2. Uygulama ve Kullanıcı Arayüzünün Tasarımı	46
6. SONUÇ	50
KAYNAKÇA.....	52

ŞEKİLLER LİSTESİ

Şekil-1: Rayleigh Saçılımı'nın gökyüzü rengine etkisi.	3
Şekil-2: Su damlasının ışığı dağıtması.	4
Şekil-3: Fujinon FE185C057HA-1 balıkgözü lensine dair bilgiler.	5
Şekil-4: Çalışmaya dair donanım ile yakalanan bir örnek görüntü.	6
Şekil-5: Net bir mercek parlaması (lens flare) örneği.	7
Şekil-6: Tümsek ayna yansıtmalı bir sistem örneği.	8
Şekil-7: Çalışmaya özgün olarak tasarlanan güneş kapatma mekanizması.	9
Şekil-8: Güneş kapatma mekanizması ile gökyüzü görüntüsü.	10
Şekil-9: GigE Vision destekli kameralar ve yerel ağ bağlantıları.	12
Şekil-10: Bir görüntünün RGB kanalları ve monokrom dönüşümleri.	13
Şekil-11: Gökyüzü (solda) ve sanal temiz gökyüzü (sağda) görüntüleri.	14
Şekil-12: Mavi-Kırmızı kanal farkı monokrom imgesi üzerinde piksel düzeltme. .	16
Şekil-13: Mavi-Kırmızı kanal fark görüntüsünün farklı değerler ile eşiklenmesi. ..	18
Şekil-14: Mavi-Kırmızı kanal farkı monokrom imgesinin eşiklenmesi.	19
Şekil-15: Kırpma maskesi tanımlı (altta) ve tanımsız (üstte) eşikleme çıktısı.....	21
Şekil-16: Basit gökyüzü rekonstrüksiyonu.	22
Şekil-17: İkili imge çevritleme örneği.	24
Şekil-18: İkili imge çıktısı ile çevritleme.	25
Şekil-19: Yeniden çevritleme için ikili imge üretimi.....	27
Şekil 20: Yeniden çevritleme ile bulutların etiketlenmesi.	28
Şekil 21: “ $t : 0$ ” ve “ $t : 10$ ” anlarında bulut takibi.....	30
Şekil-22: Bulutların hareket yönlerinin belirlenmesi. (“ $t : 50$ ” - “ $t : 100$ ”).....	35
Şekil-23: Bulutların hareket yönlerinin belirlenmesi. (“ $t : 150$ ” - “ $t : 300$ ”).....	36
Şekil-24: Bulutların hareket yönlerinin belirlenmesi. (“ $t : 250$ ” - “ $t : 300$ ”).....	37
Şekil-25: Bulutların hareket yönlerinin belirlenmesi. (“ $t : 350$ ” - “ $t : 400$ ”).....	38
Şekil-26: Güneşin ufuk ve yükseliş açıları.....	39
Şekil-27: Güneşin görüntü üzerinde konumlandırılması.	40
Şekil-28: Güneş rotasının görüntü üzerinde konumlandırılması.	42
Şekil-29: Bulut-Güneş çarpışma analizi.....	44
Şekil-30: Çalışma kapsamında geliştirilen uygulama ve arayüzü.....	48
Şekil-31: Alarm koşulları tanımlama arayüzü ve “Red Alarm” durumu.	49
Şekil-32: Bulut tanıma çıktısının benzer çalışmaların çıktıları ile karşılaştırması. .	51

KISALTMALAR

bkz.	: Bakınız
FOV	: Görüş Alanı (Field Of View)
HYTA	: Hybrid Thresholding Algorithm Dataset
IP	: İnternet Protokolü (Internet Protocol)
NRBR	: Normalize Kırmızı-Mavi Oranı (Normalized Red-Blue Ratio)
PC	: Kişisel Bilgisayar (Personal Computer)
RGB	: Kırmızı, Yeşil, Mavi (Red, Green, Blue)
TCP	: Aktarma Kontrol Protokolü (Transmission Control Protocol)
Thr.	: Eşik (Threshold)
SDK	: Yazılım Geliştirme Kiti (Software Development Kit)
SPA	: Solar Pozisyon Algoritması (Solar Position Algorithm)
SWIMSEG	: Singapore Whole Sky Imaging Segmentation Dataset
vb.	: ve benzeri
vd.	: ve diğerleri

1. GİRİŞ

Bilgisayar görsü ile gökyüzü görüntülerinin analiz edilerek, bulut bölgelerinin belirlenmesi ve görüntü akışıyla bu analizlerin iteratif olarak yapılması neticesinde bulut bölgelerinin hareket yönlerinin belirlenmesi birkaç aşamadan oluşacaktır.

İlk aşama gökyüzü görüntüsünün olabildiğince geniş açılı bir lens ile elde edilmesi olacaktır. Bu amaca yönelik olarak, yaklaşık 180° balıkgözü lens donanımlı bir IP kamera ile ufuktan ufuğa gökyüzü görüntüleri yakalanıp, bu görüntülerin dijital ortama aktarımı gerçekleştirilecektir.

Sonraki aşamalarda elde edilen dijital görüntü üzerinde C++ dili ve OpenCV kütüphanesi yardımıyla çeşitli görüntü işleme teknikleri uygulanarak gökyüzü ile bulut bölgelerinin bir birlerinden ayrıştırılmaları, dolayısıyla bulutların tespitleri sağlanacaktır.

Son aşamada görüntü üzerinde zamana bağlı olarak güneş rotasının oluşturulması ve tespit edilmiş bulut parçalarının her bir yeni çerçeve (frame) ile eşleştirilmesi dolayısıyla değişimlerine bağlı olarak hareket yönlerinin, hızlarının ve güneş rotasına olası temaslarına göre tahmini güneş kapatma zamanlarının hesaplanması gerçekleştirilecektir. Elde edilen veriler ışığında, ortaya çıkarılacak bir örnek uygulamanın bağlı çalışacağı yan sistemlere erken uyarı alarmı üretmesi sağlanacaktır.

2. BENZER ÇALIŞMALAR

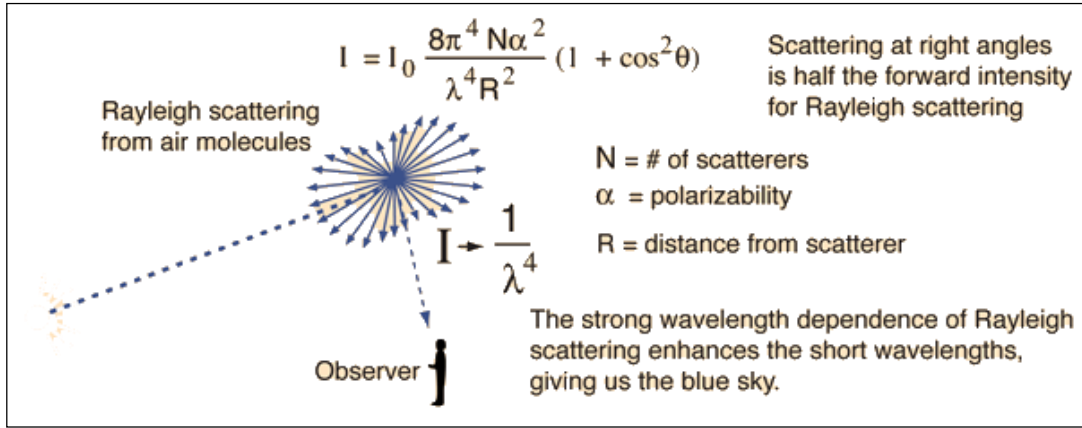
Renkler gökyüzünde bulutların ayırt edilmesi adına temel bir özelliktir ve literatürde bulut tespitine dair yer alan birçok çalışma da bu özellikten faydalanılmaktadır. Long vd. [1] RGB renk uzayındaki kırmızı ve mavi kanal oranlarının eşiklemeye tabi tutularak bulut tespiti adına iki imge üretiminde kullanılabileceğini ortaya koymuştur. Heinle vd. [2] kırmızı ve mavi kanal farkının daha başarılı bulut tespit sonuçları üreteceğini göstermiştir. Liu vd. [3] kırmızı-mavi kanal farkını kendi süperpiksel tabanlı bulut bölümleme yapıları için kullanmıştır. Souza-Echer vd. [4] doygunluk kanalını kullanarak piksellerin sınıflandırılmasına dair bir yöntem geliştirmiştir. Mantelli-Neto vd. [5] RGB renk uzayında bulut piksellerinin geometrik yerleri üzerine araştırmalarda bulunmuştur. Li vd. [6] normalize kırmızı-mavi kanalı içinde uyarlanmış eşikleme yöntemini sunmuştur. Yuan vd. [7] görüntü niteliklerinin süperpiksel sınıflandırmasını kullanarak bir bulut tanıma yapısı sunmuştur. Dev vd. [8] bulut tespiti için kısmi en küçük kareler regresyonu tabanlı bir olasılık yaklaşımı geliştirmiştir. Tüm bu çalışmalar incelendiğinde, bulutların tespit edilmesine dair birçok çalışmanın yapıldığı ve başarılı yöntemlerin geliştirilmiş olduğu görülmektedir.

Bulut hareketlerinin izlenmesine yönelik ise genellikle uydu görüntüleri üzerinden bulut hareketlerinin takibini gerçekleştirmeyi hedefleyen çalışmalar ağırlıktadır. Wood-Bradley vd. [9] OpenCV kütüphanesinde yer alan optik akış (optical flow) temelli bir bulut hareket takibi yöntemi kullanarak güneş kapatma tahmini gerçekleştirmektedir. Ancak optik akış temelli yaklaşımın gerek güvenilirlik, gerek performans ve gerek çalışabileceği hava şartları açısından verimli sonuçlar üretmediği, bir erken uyarı sisteminin hedeflerine göre ihtiyacı karşılayamayacak olduğu gözlenmektedir. Peng vd. [10] birden çok görüntüleme donanımından faydalanarak üç boyutlu uzayda bulutların takip edilmesine dair bir yaklaşım geliştirmiştir. Bulut hareketlerinin takibine dair pek az çalışmanın literatürde yer aldığı ve özellikle yalnızca tek bir görüntüleme donanımı kaynağından elde edilecek gökyüzü görüntüsünü kullanarak, bulut hareketlerinin takibini gerçekleştirmeye dayalı ihtiyacı karşılayabilir bir erken uyarı sisteminin henüz literatürde yer almadığı görülmüştür.

3. GÖKYÜZÜ KARAKTERİSTİĞİ VE GÖRÜNTÜLEME DONANIMI

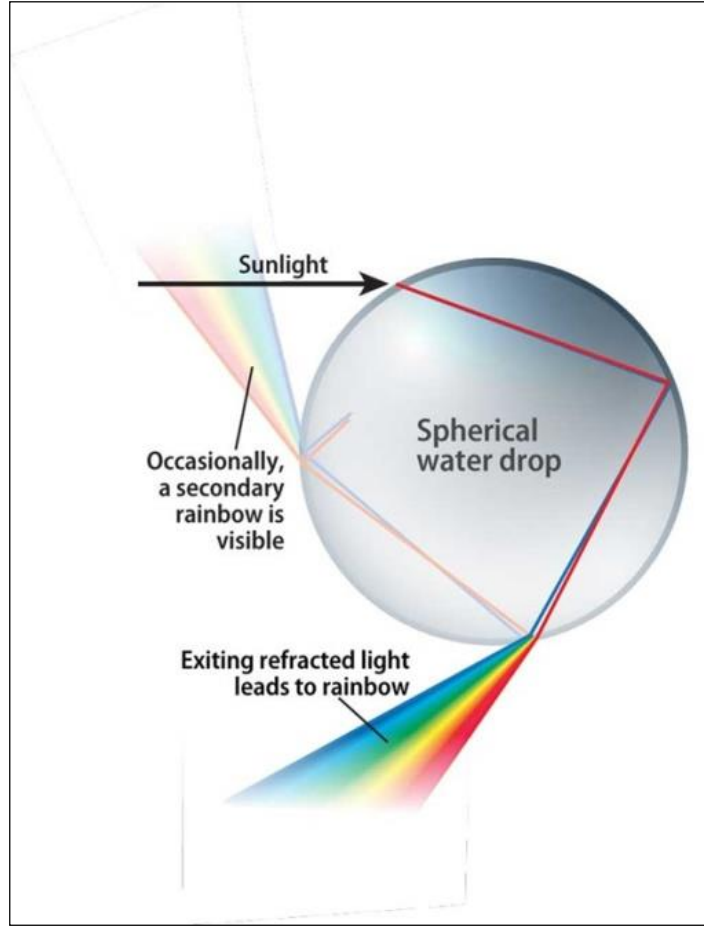
3.1. Gökyüzü Karakteristiği

Güneş ışığının atmosferdeki parçacıklar üzerinde saçılma uğraması, yeryüzünde bulunan bir gözlem noktası için gökyüzünün rengini belirlemektedir ve bu saçılım “*Rayleigh Saçılımı*” olarak adlandırılmaktadır. [13] Şekil-1’de ifade edildiği üzere, güneş ışığının yönüne göre geniş bir açı ile yeryüzüne saçtığı ışık, kısa dalga boyu sebebiyle mavi tayfa yer almaktadır, dolayısıyla da yeryüzündeki bir gözlem noktası için gökyüzünün mavi renkte görünmesine sebep olmaktadır. [13]



Şekil-1: Rayleigh Saçılımı’nın gökyüzü rengine etkisi. [13]

Gökyüzünde yer alan su damlacıkları ise içerisinden geçen ışığı dağıtarak kabaca her yöne aynı etkiyle saçarlar, su damlacıkları bulutları oluşturmak üzere kümелendiklerinde tüm damlacıklarla dağıtılan ışıkların birleşimi beyaz ışığı ortaya çıkarır, dolayısıyla bulutlar gökyüzünde beyaz renkte görünürler. [13] Ancak bulutların iç kısımlarına nüfus eden ışığın zayıflaması sebebiyle yeryüzündeki bir noktaya göre çeperleri daha beyaz görünürken, orta kısımları ise gri bir renk alabilmektedir. [13] Şekil-2’de bir su damlasının ışığı nasıl dağıttığı ifade edilmektedir.



Şekil-2: Su damlasının ışığı dağıtması.[15]

Bulutlar deniz seviyesinden yüksekliklerine göre yüksek seviye (5-13 km), orta seviye (2-7 km) ve alçak seviye (0-2 km) olmak üzere 3 ana grupta incelenirler. [14] Yüksek seviyede sirrüs, sirrokümülüs ve sirrostratüs, orta seviyede altokümülüs, altostratüs ve nimbostratüs, alçak seviyede ise stratokümülüs, stratüs, kümülüs ve kümülonimbüs olmak üzere başlıca 10 adet bulut tipi bulunmakta ve her bir bulut tipinin gökyüzünde farklı biçimsel özellikleri bulunmaktadır. [14]

3.2. Görüntüleme Donanımı

Çalışmada hedeflenen sistemin tek girdisini gökyüzü görüntüsü oluşturmaktadır. Gökyüzü görüntüsünü olabildiğince yüksek çözünürlüklü, ufuktan ufuğa, geniş açılı bir lens ile yakalamak, ilerleyen bölümlerde bahsedilecek olan görüntü işleme yöntemlerinin uygulanabilirliği açısından önem arz etmektedir.

Görüntüleme donanımı tarafından yakalanan görüntülerin çözünürlük kalitesi arttıkça daha sağlıklı analizler yapabilmek mümkün olsa da, özellikle birim zamanda işlenecek piksel sayısının artışıyla ortaya çıkacak işlem yükü artışı sebebiyle sistemin çalışma performansı negatif yönde etkilenecektir. Bu sebeple bu çalışma için 1.3MP (1280 piksel X 1024 piksel) çözünürlük değerine sahip bir renkli endüstriyel kamera olan “*Basler acA1300-60gc*” yeterli görülmüştür. Ayrıca bu aygıtın GIGE standartlarında olması ethernet bağlantısı ile uzak mesafelerden basitçe görüntü aktarımını mümkün kılmaktadır.

Görüntüleme donanımının bir diğer önemli bileşeni ise lenstir. Bu çalışma için ufuktan ufuğa bir görüntünün elde edilmek istenmesi sebebiyle 180° ve üzeri derecede FOV değerine sahip bir balıkgözü lense ihtiyaç duyulmuştur. Bu sebeple çalışmada tercih edilmiş kamera aygıtı ile kullanılabilir bir lens olan “*Fujinon FE185C057HA-1*” ürünü tercih edilmiştir. Şekil-3’de lense dair bilgiler yer almaktadır.

FE185C057HA-1		
Focal Length (mm)		1.8
Iris Range		F1.4 - F16
Operation	Focus	Fixed
	Iris	Manual
Angle Of View (HxV)	2/3°	185° x 185° (±5.7mm)
	1/2°	185°01' x 154°08'
	1/3°	154°08' x 115°27'
Focusing Range (From Front Of The Lens) (m)		∞ - 0.1
Back Focal Distance (in air) (mm)		9.70
Exit Pupil Position (From Image Plane) (mm)		-61
Filter Thread (mm)		-
Mount		C
Mass (g)		135
Remarks	• With Metal Mount • With Locking Knob for Iris	

[Fish-Eye Lens Image]



2/3°

1/2°

1/3°

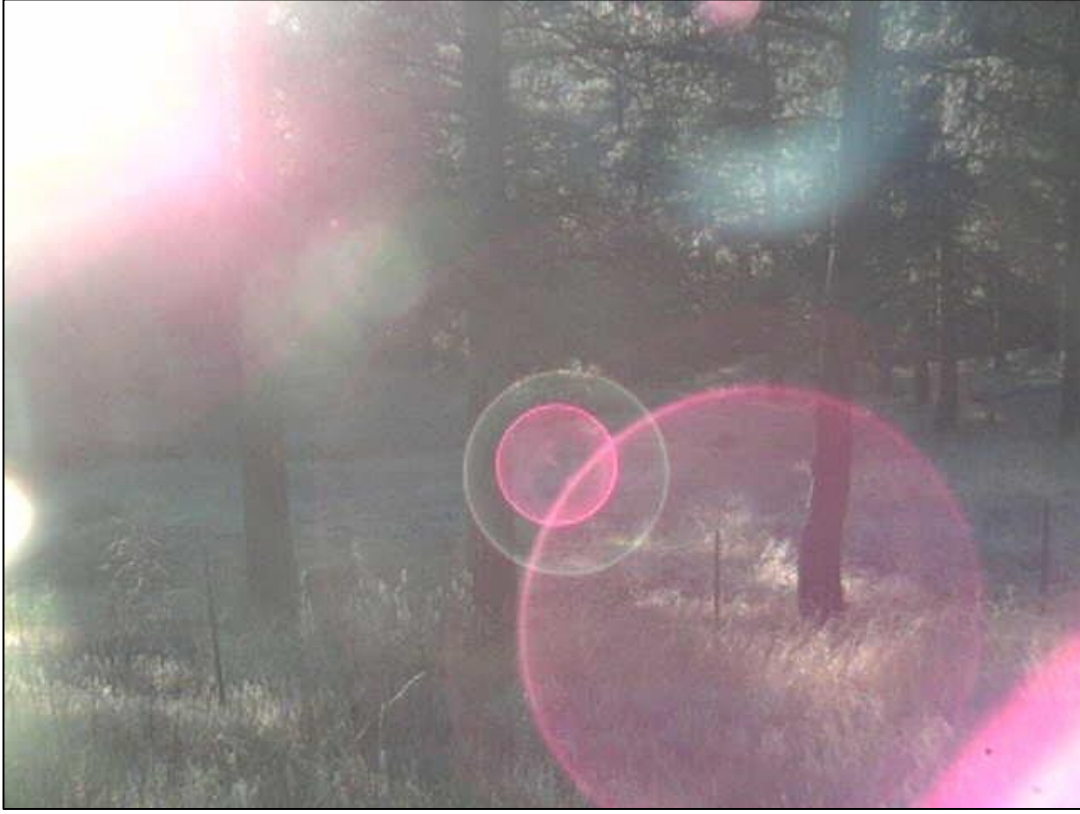
Şekil-3: Fujinon FE185C057HA-1 balıkgözü lensine dair bilgiler. [16]

Tercih edilmiş balıkgözü lens ve kamera donanımı ile birlikte yakalanmış görüntü Şekil-4’de örneklenmiştir. Örneklenen bu görüntüden anlaşılabileceği üzere, görüntüleme donanımı dik biçimde gökyüzüne bakacak şekilde

odaklandığında bulunulan konum itibari ile gökyüzü görüntüsü bir kubbe halinde dijital ortama taşınabilmesi mümkün olmaktadır. Ancak güneş ışığının doğrudan lens içinden geçmesi mercek parlaması (lens flare) olarak adlandırılan bir soruna sebebiyet vermektedir. Bu sorun doğrudan gelen ışığın lensten geçerken görüntü üzerinde istenmeyen geometrik şekiller veya renk değişimleri oluşturması ile ilgilidir. (bkz. Şekil-5)



Şekil-4: Çalışmaya dair donanım ile yakalanan bir örnek görüntü.



Şekil-5: Net bir mercek parlaması (lens flare) örneği. [17]

Mercek parlamasının önüne geçebilmek için güneş ışığının doğrudan lens içerisine girişini engellemek gerekmektedir. Ancak güneş zamana bağlı olarak yer değiştirmekte olduğundan bunu sağlamak için güneş ışığını engelleyecek cismin bir hareketli bir mekanizma ile sürekli güneşi kapatır konumda tutulmasına ihtiyaç duyulmaktadır. Benzer çalışmalarda bu mekanizmayı basitleştirmek adına geniş açılı balıkgözü lens ile doğrudan gökyüzünü çekmek yerine, dar açılı lens ile görüntüyü bir tümsek ayna üzerinden almak ve güneşin parlama yapacağı noktayı ayna üzerinde mat bir şerit cisim ile kapatma yönteminin de uygulandığı görülmüştür. Şekil-6'da bu tür bir sisteme dair örnek verilmiştir, söz konusu sistemde tümsek aynayı kendi ekseninde çevirmek suretiyle güneşi ayna üzerine yapıştırılmış mat bir şerit içerisinde tutarak mercek parlamasının önüne geçilmektedir.



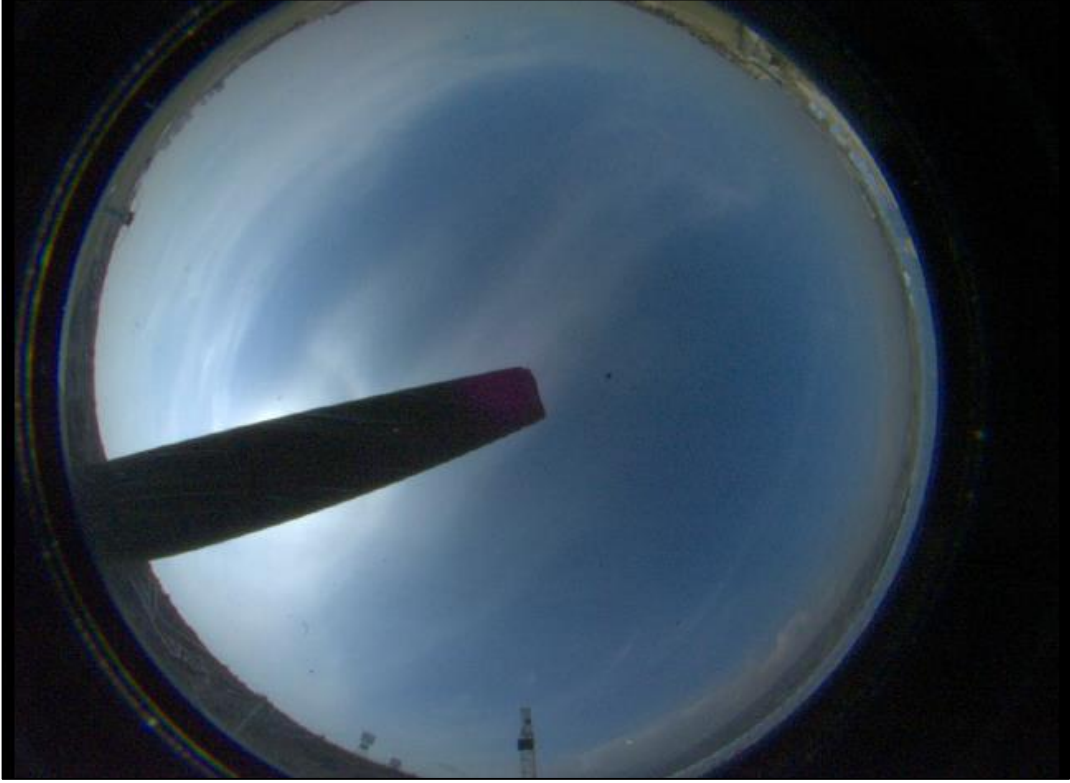
Şekil-6: Tümsek ayna yansıtmalı bir sistem örneği. [19]

Bu çalışma sonucu ortaya çıkarılacak olan ilgili sistemin kullanım hedef alanı solar enerji sahaları olması sebebiyle, görüntüleme donanımının ve güneş kapatma mekanizmasının tozlu ortamlarda çalışabilecek, mümkün kılınabilecek en az bakıma ihtiyaç duyacak bir sistem olması hedeflenmektedir. Bu bağlamda bu çalışma için tercih edilen yöntem ise mat renkli bir metal çubuğu, güneşin gün içindeki hareketi baz alınarak lens ekseninde hareket ettirecek basit bir mekanizmanın tertip edilmesi olmuştur. Şekil-7’de bu çalışmaya özgün olarak ortaya çıkarılmış güneş kapat

mekanizması ve Şekil-8’de bu mekanizma çalışır durumdayken yakalanmış bir örnek gökyüzü görüntüsü yer almaktadır.



Şekil-7: Çalışmaya özgün olarak tasarlanan güneş kapatma mekanizması.



Şekil-8: Güneş kapatma mekanizması ile gökyüzü görüntüsü.

Tertip edilen bu basit mekanizma vasıtası ile gökyüzü alanından küçük miktarda kayıp verilse de, mercek parlamasının önüne geçilmiş olmaktadır. Ayrıca tek eksenle hareket edebilen düz ve sabit boyutlu bir çubuk yerine, daha karmaşık bir mekanizma ile iki eksenle hareket edebilen ve tam olarak güneş üzerine oturacak daire benzeri bir cisim ile gökyüzü alanından daha az kayıp vermenin de mümkün olduğu öngörülmektedir.

4. GÖKYÜZÜ GÖRÜNTÜLERİNİN İŞLENMESİ

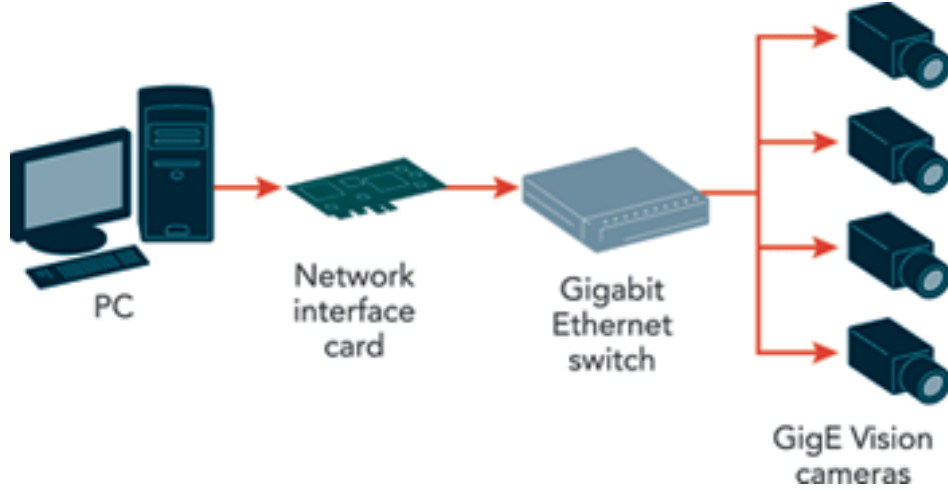
Görüntüleme donanımı ile gerçek zamanlı olarak elde edilecek görüntüler dijital ortama aktarıldıktan sonra, bulut ve gökyüzü bölgelerinin bir birlerinden farklı karakteristik özelliklere sahip olmaları sebebiyle bir takım görüntü işleme teknikleri kullanılarak bu alanların kolayca bir birlerinden ayrıştırılabileceği öngörülmektedir.

Gerçek zamanlı görüntü işlemenin gerektireceği performans ve görüntüleme donanımından dijital görüntüyü aktarabilmek için bir takım kütüphanelere ihtiyaç olduğu göz önünde bulundurularak, bu çalışma için C++ dili ve gereksinim duyulacak temel görüntü işleme tekniklerini bünyesinde barındıran OpenCV kütüphanesinin kullanımı uygun görülmüştür.

4.1. Görüntü Aktarımı

Görüntüleme donanımı ile yakalanan görüntülerin, görüntü işleme ortamına aktarılabilmesi için kullanılacak programlama dili olan C++ ile kullanılabilir bir kütüphane gereksinimi bulunmaktadır. Tercih edilen görüntüleme donanımının “*GIGE Vision*” standartlarını taşıması bu amaçla kullanılabilir birçok kütüphane seçeneği sunmaktadır.

Bu çalışmada tercih edilmiş olan görüntüleme donanımının üreticisi kendi donanımları için “*pylon SDK*” adında bir yazılım geliştirme kitini de müşterilerine sunmaktadır. Bu geliştirme kiti vasıtası ile yerel ağ üzerindeki kamera aygıt(lar)ı yazılımsal olarak bulunabilir ve kendilerinden görüntü aktarımı kolayca başlatılabilir. (bkz. Şekil-9)



Şekil-9: GigE Vision destekli kameralar ve yerel ağ bağlantıları. [18]

4.2. Gökyüzü Bölgesinin Eleminasyonu

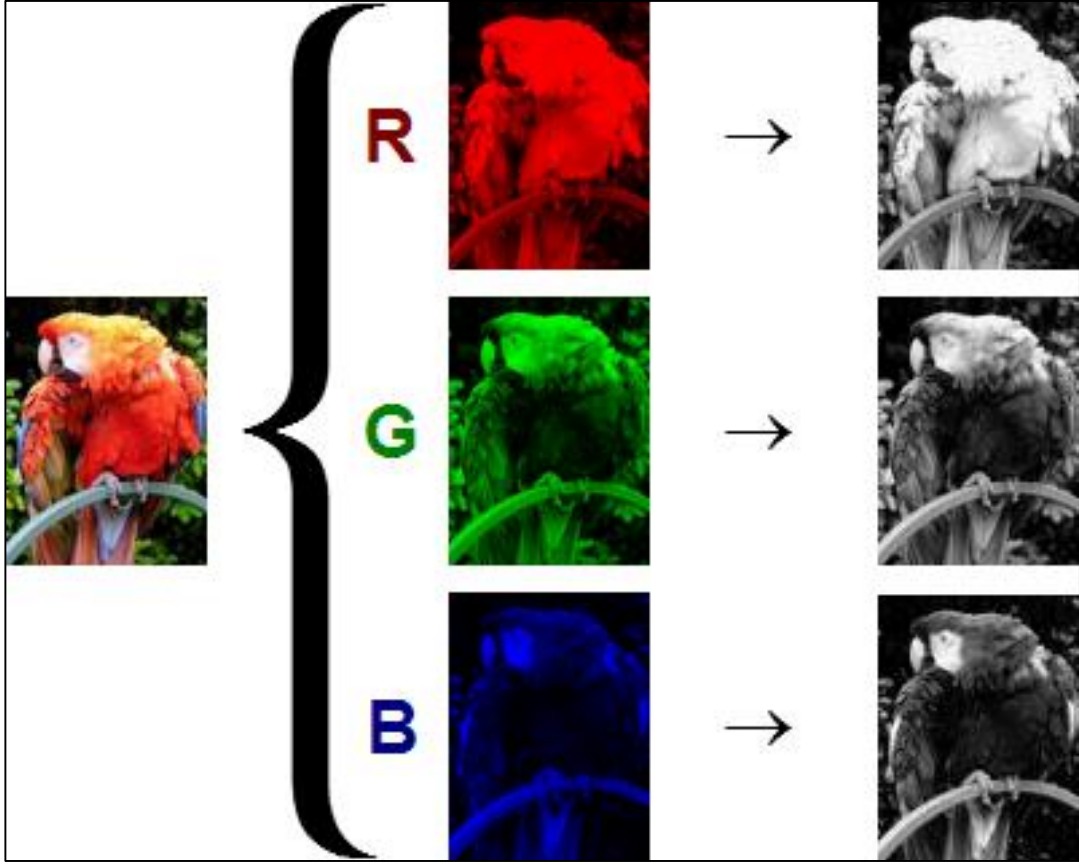
Gökyüzü eleminasyonu bu çalışmaya dair en önemli aşamalardan biridir. Görüntüleme donanımı görüntülerinin yazılımsal olarak dijital ortama aktarılmasından sonra OpenCV kütüphanesi yardımı ile bu görüntüler üzerindeki gökyüzü-bulut bölgelerinin ayrıştırılması amacıyla asıl görüntü işlemenin yapılacağı bu aşamaya geçilebilir.

4.2.1. Görüntünün Renk Kanallarına Ayrıştırılması

Gökyüzü karakteristiğine dair bölümde ifade edildiği üzere farklı renk karakteristiğine sahip bulut ve gökyüzü bölgeleri dijital görüntü üzerinde de farklı piksel karakteristiklerine sahip olmaktadır. Bulut bölgelerinin tespit edilebilmesi için bu renk farkından yararlanarak gökyüzü bölgeleri kabaca elemine edilebilir.

Bilgisayar sistemlerinde pikseller “*RGB*” renk uzayıyla ifade edilmektedir, yani her bir piksel temelde Kırmızı, Yeşil ve Mavi olmak üzere 3 ayrı renk değerine sahiptir. Bir dijital görüntüyü meydana getiren bu pikseller ve farklı renk değerleri

ortaya “*kanal*” terimini çıkartmaktadır, yani bir dijital görüntü aynı zamanda piksellerinin renk değerlerine göre 3 ayrı kanaldan oluşmaktadır. (bkz. Şekil-10)

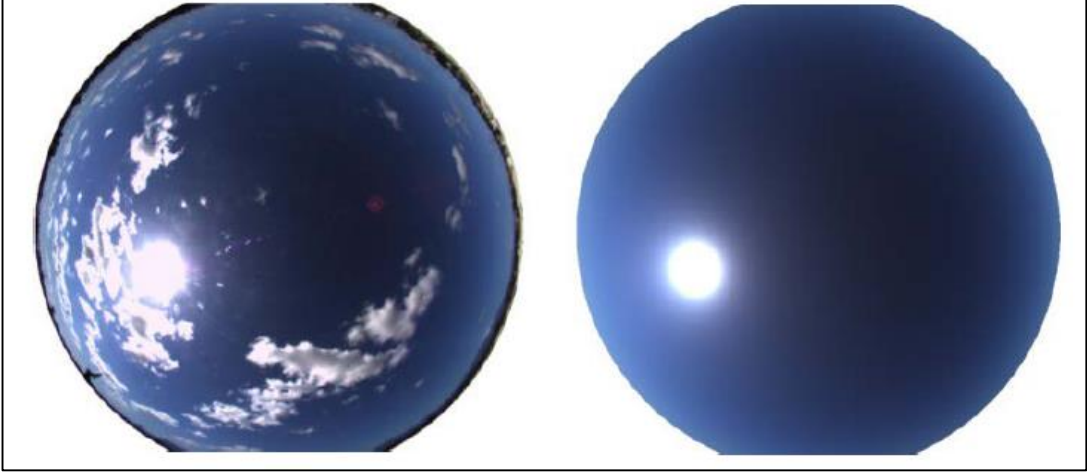


Şekil-10: Bir görüntünün RGB kanalları ve monokrom dönüşümleri. [20]

Dijital görüntü üzerinde bulut bölgeleri beyaz, yani her üç kanalda da yüksek piksel değerlerine sahip olmakta, gökyüzü bölgeleri ise mavi, yani tek bir kanalda yüksek ve diğer kanallarda düşük piksel değerlerine sahip olmaktadır. [11] Bu bilgi ışığında dijital görüntü kanallarına ayrılıp, Mavi-Kırmızı kanal farkı ile ortaya çıkarılacak monokrom görüntü üzerinde eşikleme yapıldığında bulut veya gökyüzü bölgelerinin eliminasyonunun kolayca yapılabileceği görülmüştür.

Balıkgözü lens etkisi ile kubbemsi biçimde görülen gökyüzü görüntüsünde kubbenin tepe noktası ve ufuklar arasında bir renk geçişi yaşanmaktadır ve dolayısıyla tüm görüntüye uygulanacak bir eşikleme tam olarak doğru sonuçları üretemeyecektir. [11] Bu sorunu bertaraf etmek üzere orijinal görüntü üzerinden ufuklara doğru olan renk geçişine de sahip sanal bir temiz gökyüzü görüntüsü

üretilebilir ve orijinal görüntü ile sanal temiz gökyüzü görüntüsünün farkına bakılarak bulut alanlarının doğrudan ortaya çıkması sağlanabilir. [11] (bkz. Şekil-11)



Şekil-11: Gökyüzü (solda) ve sanal temiz gökyüzü (sağda) görüntüleri. [11]

Çalışma kapsamında tüm gökyüzünün bulutlar ile kapalı olduğu veya hava kirliliği vb. doğal veya doğal olmayan etkenler ile gökyüzünde meydana gelebilecek olağandışı renk değişimlerinde üretilecek sanal temiz gökyüzü görüntüsü arasında fark bulunacağından bu yöntemden verim alınamayabileceği öngörülmüştür. Bu sebeple, bu çalışmada eşikleme ardından hatalı sınıflandırılmış olması muhtemel pikseller merkezden uzaklıklarına göre kontrol edilerek, NRBR ve doygunluk hesaplamalarına göre bulut/gökyüzü pikseli olup olmadığına karar verilip, Mavi-Kırmızı kanal fark görüntüsü üzerinde doğrudan düzeltme yapma yaklaşımı tercih edilmiştir. Ayrıca tercih edilen bu yaklaşımın, güneş hareketlerine de bağlı olarak sürekli değişken bir sanal temiz gökyüzü görüntüsü üretmekten kaçınılmış olması sebebiyle gerçek zamanlı çalışacak bir sistem için performans açısından daha verimli çalışacağı öngörülmüştür. Bu yaklaşıma dair oluşturulan algoritmanın kodlaması aşağıda yer almaktadır.

```
...  
// split to BGR channels  
cv::split(img_input, img_bgr_ch);  
  
// B-R diff. image  
cv::absdiff(img_bgr_ch[2], img_bgr_ch[0], img_brdiff);
```

```

double b, g, r, pix, sat, nrbr;
for(int y = 0; y < img_brdiff.rows; y++)
{
    for(int x = 0; x < img_brdiff.cols; x++)
    {
        double dist = cv::norm(CENTER-cv::Point(x, y)) / R;

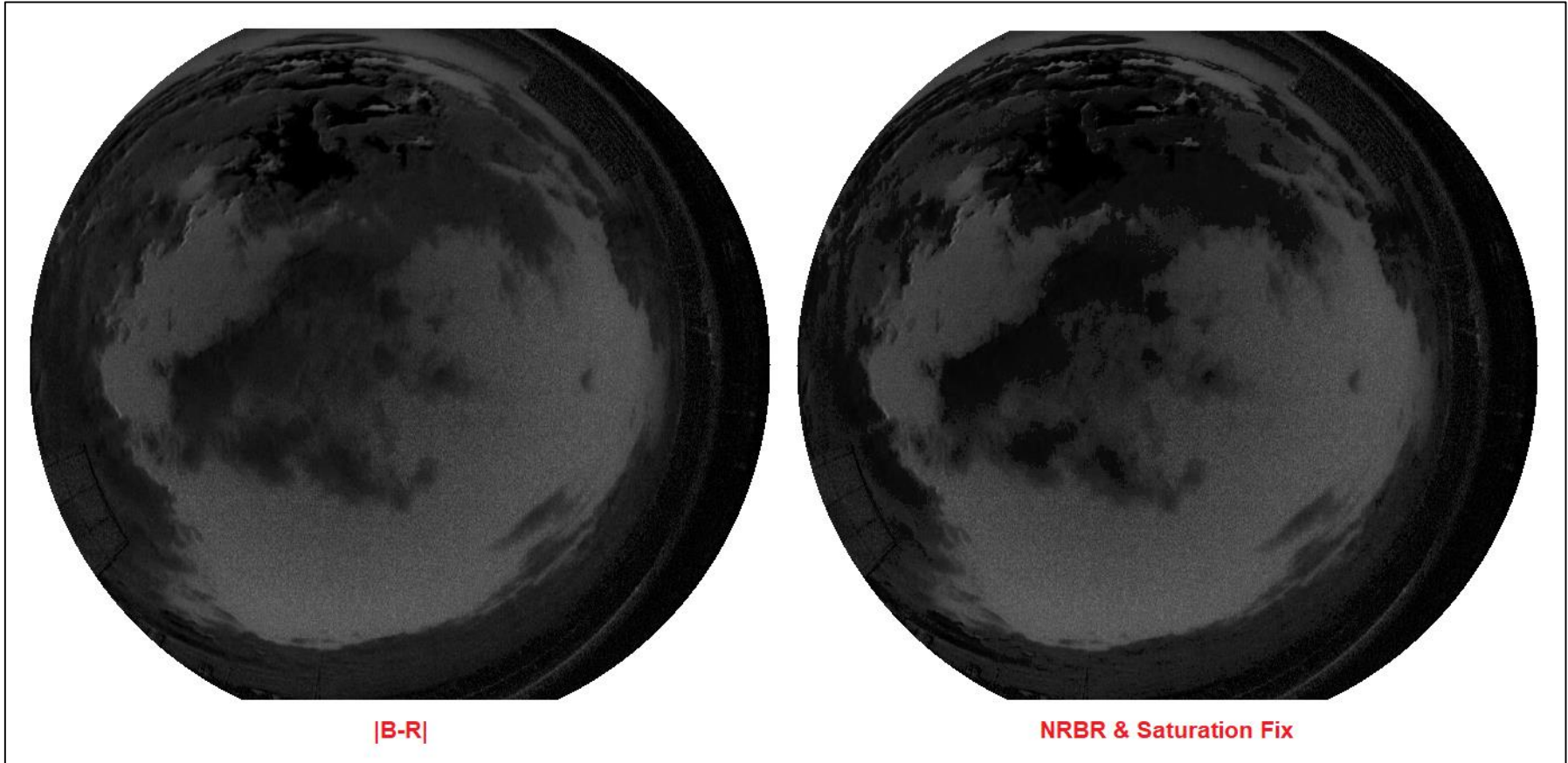
        pix = img_brdiff.at<uchar>(y, x);

        b = img_input.at<cv::Vec3b>(y, x)[0];
        g = img_input.at<cv::Vec3b>(y, x)[1];
        r = img_input.at<cv::Vec3b>(y, x)[2];
        sat = 1.0 - (std::min(b, std::min(g, r)) /
                    std::max(b, std::max(g, r)));
        nrbr = (r - b) / (r + b);

        if(nrbr < THR_NRBR && sat < THR_SAT)
            img_brdiff.at<uchar>(y, x) =
                (uchar)(pix * (FIX_MUL + dist / 10.0))
                % 256;
    }
}
...

```

Algoritma da yer alan sabitler olan “*THR_NRBR*”, “*THR_SAT*” ve “*FIX_MUL*” değerleri deneyler sonucu sırasıyla “0”, “0.2” ve “0.5” olarak belirlenmiştir ve ayrıca bu sabitler üzerinde yapılacak ince ayar neticesinde doğrudan güneş ışığına maruz kaldığında ışığı geçirecek kadar ince katmanlı bulutların elemine edilmesi de mümkün olmaktadır. “*CENTER*” ve “*R*” sabitleri ise işleme tabi olan görüntü alanının sırasıyla koordinatsal merkez noktasını ve yarıçap değerini ifade etmektedir. Şekil-12’de sol tarafta Mavi-Kırmızı kanal farkı monokrom imgesi ve sağ tarafta yukarıdaki kodlamanın sonucu olarak eşikleme aşaması öncesi düzeltilmiş monokrom imge yer almaktadır.



Şekil-12: Mavi-Kırmızı kanal farkı monokrom imgesi üzerinde piksel düzeltme.

4.2.2. Eşikleme ve Morfolojik Operasyonlar

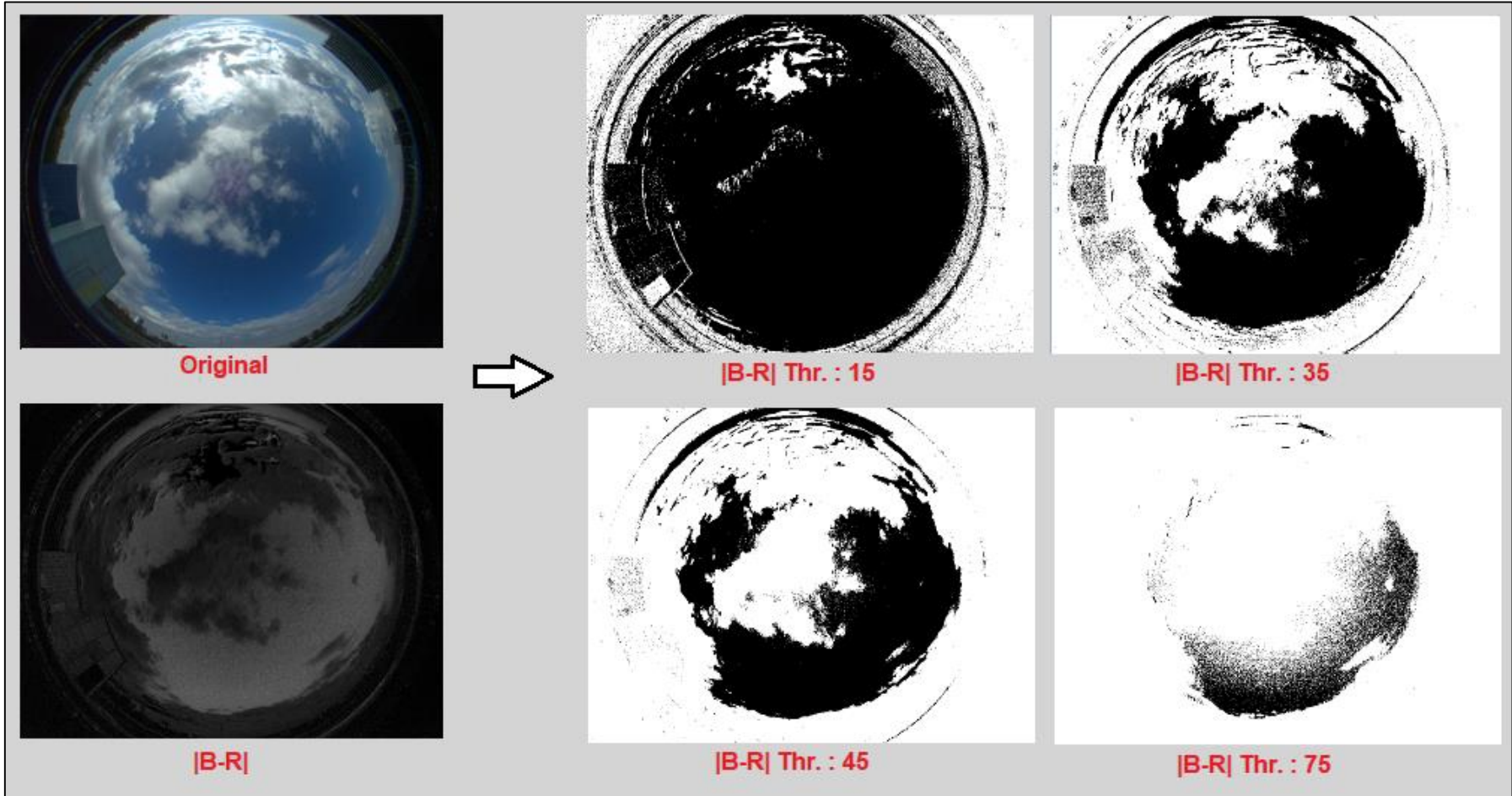
Eşikleme imge bölütleme (image segmentation) işlemlerine dair en temel operasyonlardan biridir. Bir imgenin belirli bir eşik değeri baz alınarak eşik değeri üzerindeki ve aşağısındaki piksellerine karşılık ikili imge çıktısını almak amacıyla kullanılır.

Bir önceki aşamada elde edilmiş olan monokrom Mavi-Kırmızı fark görüntüsü üzerinde eşikleme uygulandığında, eşik değeri üzerinde kalacak pikseller gökyüzü bölgesini, değeri altında kalacak pikseller ise bulut bölgelerini belirleyecek bir ikili imge (binary image) çıktısı alınabilmesini mümkün kılacaktır. OpenCV kütüphanesi kullanılarak aşağıdaki örnek kodlama ile eşikleme işlemi basitçe uygulanabilir:

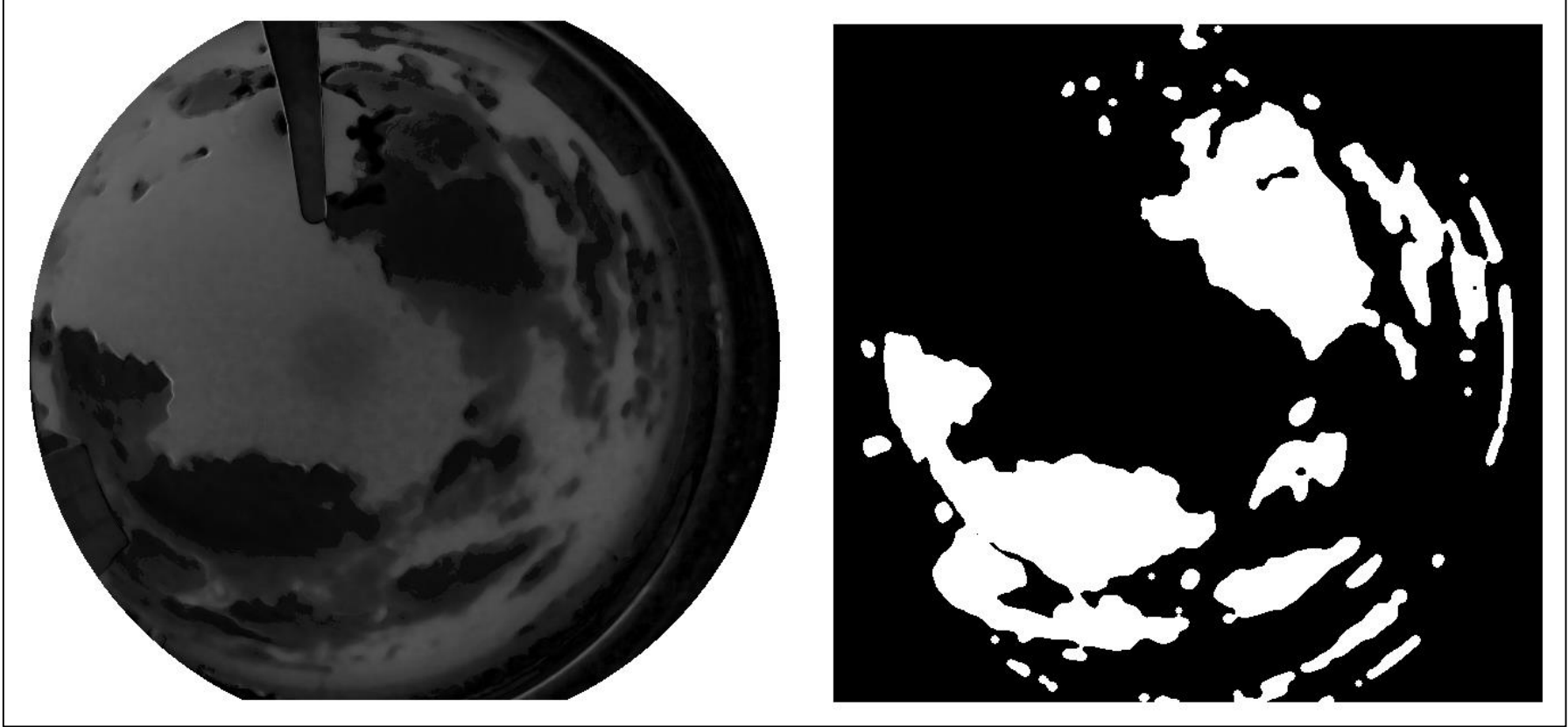
```
...  
cv::threshold(img_brdiff, img_brdiff, 35, 255,  
CV_THRESH_BINARY_INV);  
...
```

Şekil-11’de gökyüzü görüntüsü, Mavi-Kırmızı kanal fark görüntüsü ve Mavi-Kırmızı kanal fark görüntüsüne farklı değerlerde ters (inverted) eşikleme yapılarak üretilmiş ikili imgeleri gösterilmektedir. Şekilden görülebileceği üzere çok düşük veya çok yüksek eşik değerleri ile arzu edilen sonuca varılamadığı gözlenmiştir. Piksel bazında 0-255 aralığında olması gereken eşikleme değeri, çalışma esnasında gerçekleştirilen deney ve gözlemler neticesinde “35” olarak belirlenmiştir.

Eşikleme işlemi ardından ikili imge üzerindeki istenmeyen bir takım gürültülerden kurtulmak için filtreleme yapılması ve özellikle bulutlar arasındaki bağlantıları zayıflatmak amacıyla da morfolojik operasyonların (MORPH_OPEN) yapılmasına ihtiyaç görülmüştür. Ayrıca görüntüde yer alan güneş kapatma çubuğu gibi yabancı nesnelerin de ikili imge üzerinden kaldırılabilmesi adına, monokroma dönüştürülmüş ham görüntü üzerine farklı bir eşikleme daha yapılarak (Thr. : “40”) üretilen ikili imge ile bir önceki aşamada üretilmiş olan ikili imgenin kesişimlerini almak (bitwise-and) uygun bulunmuştur. (bkz. Şekil-14)



Şekil-13: Mavi-Kırmızı kanal fark görüntüsünün farklı değerler ile eşiklenmesi.



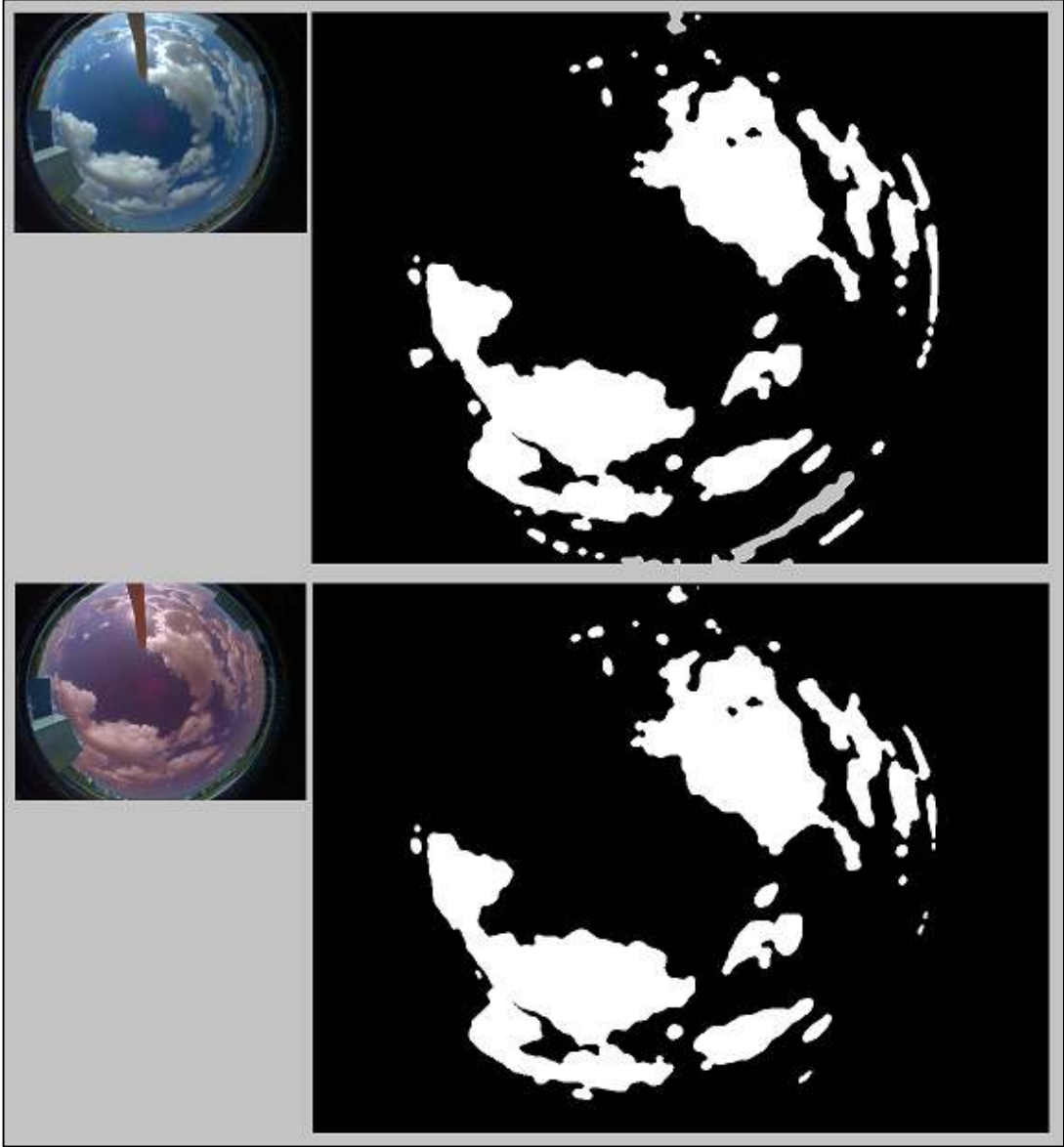
Şekil-14: Mavi-Kırmızı kanal farkı monokrom imgesinin eşiklenmesi.

Aşağıda Şekil-14 ile ifade edilmiş çıktıyı üreten yaklaşıma dair kodlama yer almaktadır:

```
...
cv::cvtColor(img_input, img_input_gray, CV_BGR2GRAY); // to mono.
cv::threshold(img_input_gray, img_in_thr, 40,
    255, CV_THRESH_OTSU|CV_THRESH_BINARY); // Thr. 40
cv::bitwise_and(img_brdiff, img_in_thr, img_brdiff); // Union
cv::morphologyEx(img_brdiff, img_brdiff, cv::MORPH_OPEN, EL,
    cv::Point(-1,-1), 1); // Opening op.
cv::medianBlur(img_brdiff, img_brdiff, 7); // Median filter
...
```

Gökyüzü görüntüsü üzerinde bazı durumlarda eşikleme ile atılamayacak türden yabancı cisimler de yer alabilir. Özellikle görüntüleme donanımı çevresinde sabit olarak yer alan ve ufuk çizgisinden görüntü alanına giren farklı renklere sahip cisimlerin görüntü alanından bir kereye mahsus olmak üzere bir maske gibi işaretlenerek kırılmasına eşikleme işlemleri öncesinde ihtiyaç duyulmaktadır.

Şekil-15’de çalışma kapsamında geliştirilen proje arayüzü vasıtasıyla girdi görüntü üzerine kırpma maskesinin tanımlanışı (sol altta) ve bu maskeye göre kırılan ikili imge görüntüsü (sağ altta), işlem yapılmamış halleri (yukarıda) ile birlikte gösterilmektedir.



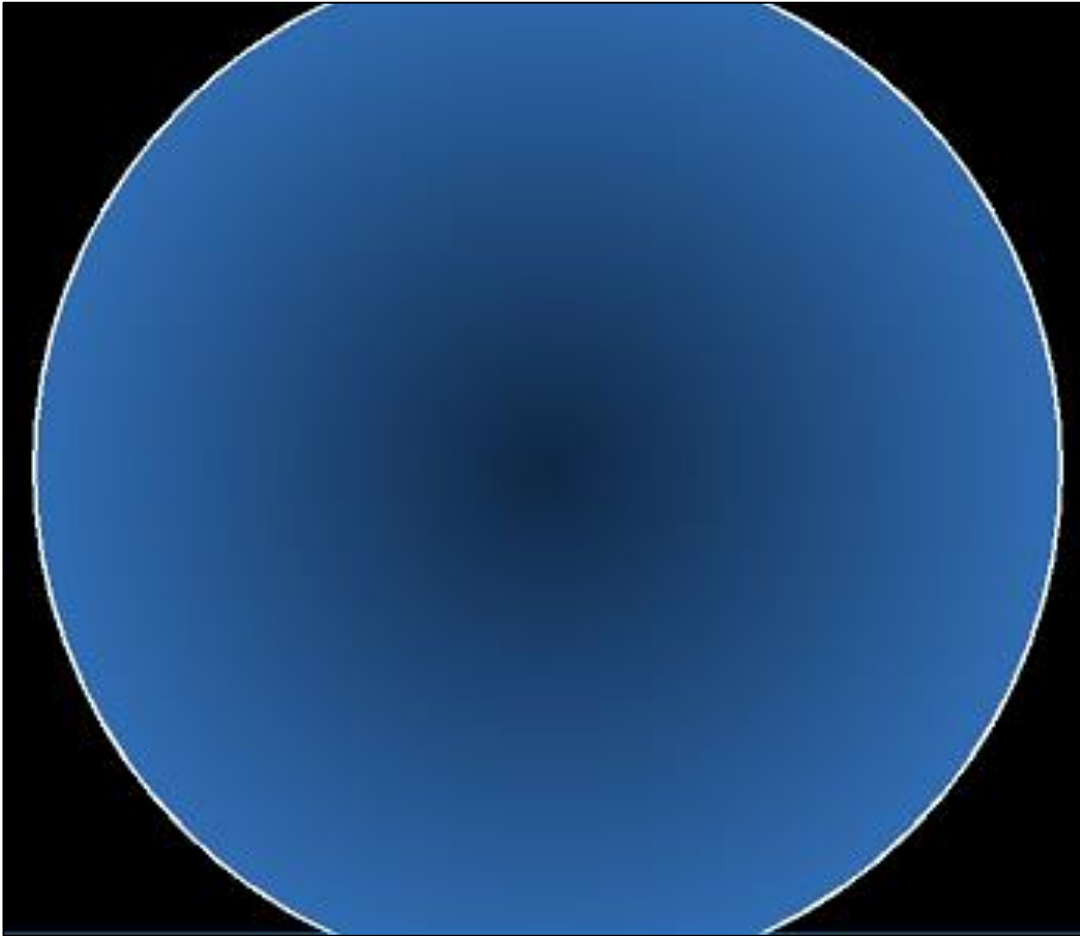
Şekil-15: Kırpma maskesi tanımlı (altta) ve tanımsız (üstte) eşikleme çıktısı.

4.2.3. Gökyüzü Rekonstrüksiyonu

Girdi görüntüsü üzerindeki gökyüzü bölgesinin elemine edilmesi neticesinde bulut bölgelerinin elde edilmesiyle, çalışma kapsamında geliştirilen sistemin görsel arayüzünde sonraki aşamalarda tespit edilecek bulutların daha anlaşılır biçimde sunulabilmesi için görüş alanı içerisindeki gökyüzünün basit bir yöntem ile tekrar üretilmesine ihtiyaç görülmüştür.

“Görüntünün Renk Kanallarına Ayırıştırılması” aşamasında görüntü pikselleri itere edilerek piksel yeniden sınıflandırması yapılırken, aynı döngü içerisinde aşağıdaki kod parçası ile ek bir performans kaybı yaşanmaksızın merkezden olan uzaklıklarına göre basit bir temiz gökyüzü rekonstrüksiyonu sağlanabilir: (bkz. Şekil-16)

```
...  
cv::Mat img_sky = cv::Mat::zeros(img_input.size(), CV_8UC3);  
cv::circle(img_sky, CENTER, R, cv::Scalar(255, 255, 255), 2);  
double dist = cv::norm(CENTER - cv::Point(x, y)) / R;  
img_sky.at<cv::Vec3b>(y, x) = cv::Vec3b( 110 * dist + 70,  
                                           70 * dist + 40,  
                                           35 * dist + 15  
                                           );  
...
```



Şekil-16: Basit gökyüzü rekonstrüksiyonu.

4.3. Bulutların Tanımlanması

Farklı bulut tiplerinin gökyüzünde kendine has biçimsel özellikleri mevcuttur; bulutların bu denli biçimsel değişkenlik göstermesi ve aynı zaman diliminde katmanlar halinde üst üste yer almaları gökyüzü görüntüsü üzerinde görüntü işleme yöntemleri ile spesifik bulut tiplerini tanımlamayı güç kılmaktadır. Bu sebeple görüntü üzerindeki gökyüzü bölgesinin görüntü işleme yöntemleri ile elemine edilışinden sonra geriye kalan bölgeleri bulut bölgesi olarak ele almak yerinde ve işlemi basitleştiren bir yaklaşım olacaktır.

“*Gökyüzü Bölgesinin Eliminasyonu*” bölümündeki aşamalar ardından elde edilen iki imge görüntüsü bulut bölgelerini işaret eder niteliktedir. Çalışma ile görölmüştür ki OpenCV kütüphanesinde yer alan çevritleme (contouring) fonksiyonları yardımı ile ikili imge üzerindeki bulut bölgeleri kolayca tespit edilip, birer nesne olarak ele alınabilirler ve böylece üzerlerinde programatik işlemler uygulanabilir.

4.3.1. Çevritleme

Çevritler OpenCV kütüphanesinde basitçe aynı renk veya yoğunluğa sahip olan piksel gruplarını çeperleri boyunca birleştiren çokgenler olarak açıklanabilirler; şekil analizi, nesne algılama ve tanıma işlemleri için popüler ve de kullanışlı araçlardır.

Çevritleme işlemi ikili imgeler üzerinde uygulandığında daha verimli sonuçlar elde edilmektedir, bu sebeple çevritleme yapılacak görüntünün önce eşikleme veya kenar bulma (edge-detection) operasyonlarına tabi tutulması önerilmektedir. [21] Şekil-17’de bir ikili imge görüntüsünün (solda) çevritlenerek çevrit nesnesinin çizdirilmesi (ortada) ve çevrit nesnesi üzerinden yapılabilecek işlemlere örnek olarak dışbükey örtü (convex hull) ve kusur noktaları (contour defects) (sağda) gösterilmektedir.



Şekil-17: İkili imge çevritleme örneği. [21]

Çalışmanın önceki aşamalarında gökyüzü eliminasyonu çıktısı olarak üretilen bulut ikili imgesi üzerine çevritleme operasyonu yapılarak bulutların birer çevrit nesnesi olarak elde edilmesi mümkün olmaktadır. Elde edilecek bu çevritler üzerinden moment, dışbükey örtü, kusur noktaları vb. gibi hesaplamaların kolayca yapılabilmesi çalışmanın ilerleyen aşamalarında bu tür hesaplamalara ihtiyaç duyulacağı da göz önüne alınırsa çevritler ile çalışmayı oldukça faydalı kılmaktadır.

Şekil-18’de gökyüzü eliminasyonu aşaması çıktısı olarak elde edilen ikili imge üzerinde çevritleme işleminin uygulanışı neticesinde elde edilmiş çevrit nesneleri kırmızı renkli çeper biçimde çizdirilerek gösterilmiştir. Bu operasyona dair örnek kodlama aşağıda yer almaktadır:

```
...
cv::vector<cv::vector<cv::Point>> conts;
cv::vector<cv::Vec4i> hier;
cv::findContours(img_brdiff, conts, hier, cv::RETR_CCOMP,
    CV_CHAIN_APPROX_SIMPLE);
for(size_t i = 0; i < conts.size(); i++)
    cv::drawContours(img_out, conts, i, cv::Scalar(0,0,255), 1, 8,
        hier, 0);
...
```



Şekil-18: İkili imge çıktısı ile çevritleme.

4.3.2. Bulutların Ayrıştırılması

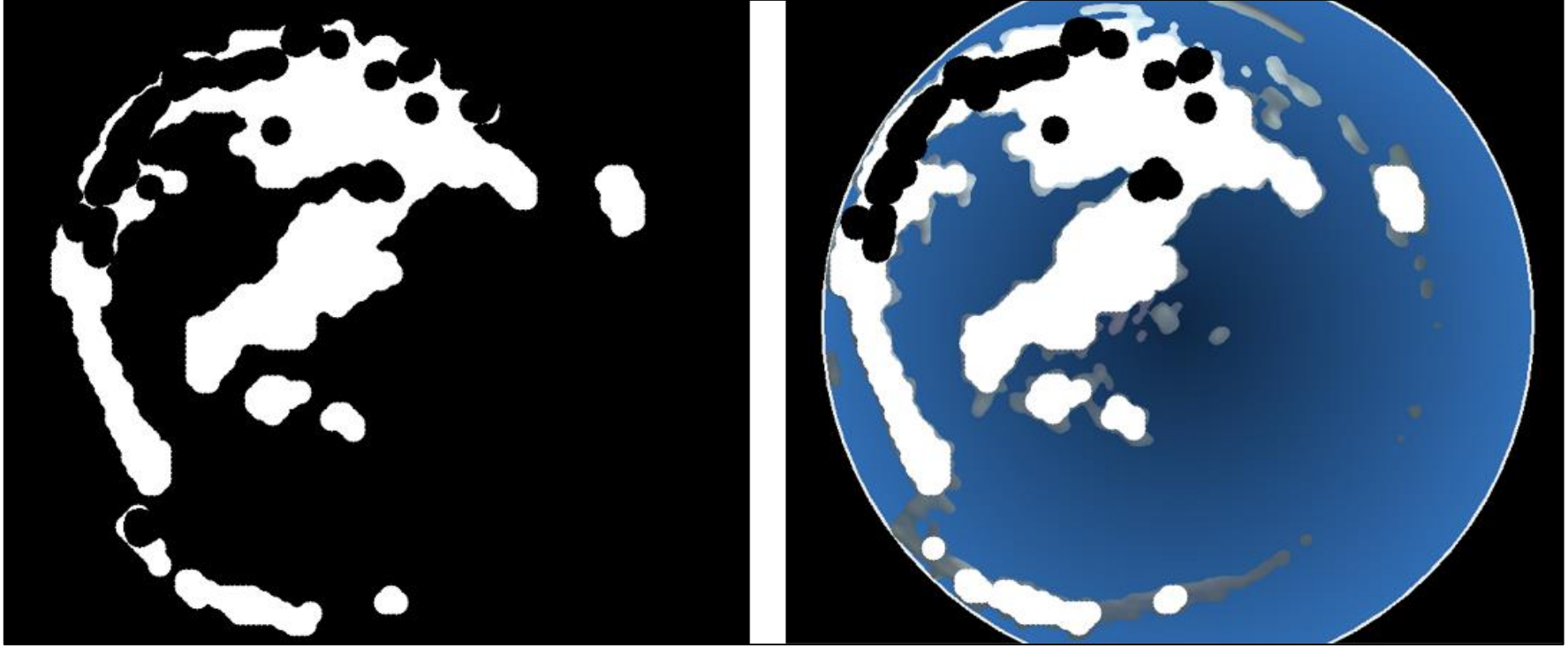
Gökyüzünde farklı veya aynı katmanlarda bulunan bulutların bir birleri üzerine hareket etmeleri ve yeryüzünden birleşik biçimde görünüyör olmaları sebebiyle çevritleme sonucunda bir birlerinden farklı birçok bulut bölgesinin tek bir çevrit nesnesi olarak tanımlanması sorunu ortaya çıkmaktadır ve Şekil-18’de bu durum gözlenebilir.

Çalışmanın ilerleyen aşamalarında bulutların bir birlerinden bağımsız olarak ele alınarak takip edilebilmesi adına görüntü üzerinde birleşim gösteren bulutların, dolayısıyla çevritlerinin de bir birlerinden olabildiğince ayrıştırılmaları gerekmektedir.

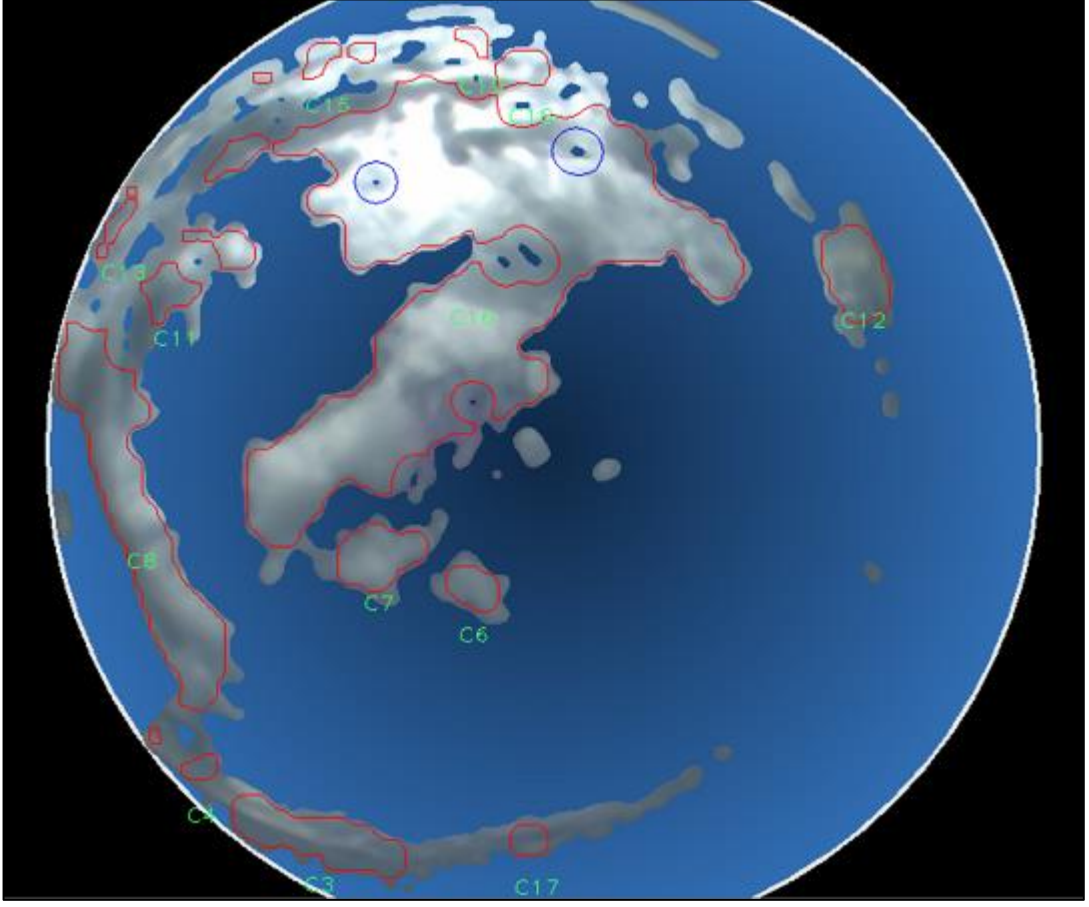
Çalışma sırasında bulut çevritlerinin kusur noktaları (defect points) üzerinden bir parçalama algoritması geliştirilebileceği de düşünülmüştür, fakat bulut çevritlerinin çerçeveden çerçeveye çok değişken bulunması ve de OpenCV kütüphanesinin çevrit kusur noktalarını ortaya çıkarmadaki tutarsız sonuçları bu yöntemden vazgeçilmesine sebep olmuştur.

Çalışma sırasındaki test ve gözlemler neticesinde bir birlerine zayıf bağlar ile bağlı bulunan bulutların çevritleme sonrasında iç bölgelerini sınırlarından içeriye “10” piksel boşluk bırakarak daire geometrileri ile doldurmak suretiyle zayıf bağların büyük çoğunlukla koparılabildiği ve yeniden ikili imge üretimine müteakip yeniden çevritleme yapıldığında bu yöntemden verim alınabildiği sonucuna varılmıştır.

Şekil-19’da sol tarafta çevrit iç bölgelerinin doldurulması ile yeniden üretilen ikili imge ve sağ tarafta girdi görüntü üzerinde bu ikili imgenin tekrar yerleşimi siyah renkli çevrit deliklerini gösterir şekilde ifade edilmiştir. Yeniden üretilen ikili imge ile ikinci kez çevritleme yapıldığında tespit edilecek bulut nesneleri gerçeği daha iyi yansıtır bir hal almaktadır. (bkz. Şekil-20)



Şekil-19: Yeniden çevritleme için ikili imge üretimi.



Şekil 20: Yeniden çevritleme ile bulutların etiketlenmesi.

Şekil-20’de bir birlerine bağlı bulunan bulutların büyük oranda ayrıştırılmış oldukları ve artık birer bulut olarak işaretlendikleri görülmektedir. Bulut çeperlerinde oluşacak geri çekilme ve şerit biçiminde düşük yoğunluklu bulutların tespit edilmiyor oluşu göz ardı edilebilir niteliktedir, zira kayda değer bir güneş kesintisi oluşturabilir türden bulutların belirli bir hacim yoğunluğuna sahip olması gerekmektedir. Bu bağlamda elde edilen çevritler üzerinden piksel cinsinden tanımlanmış bulut bölgelerinin alanları hesaplanıp, görüş alanına oranla belirli bir limit altında kalanların (örneğin “%1”) göz ardı edilmesi de bulutların takibine dair aşamalardaki işlem yükünü azaltmak adına yerinde bir yaklaşım olacaktır.

4.4. Bulutların Takibi

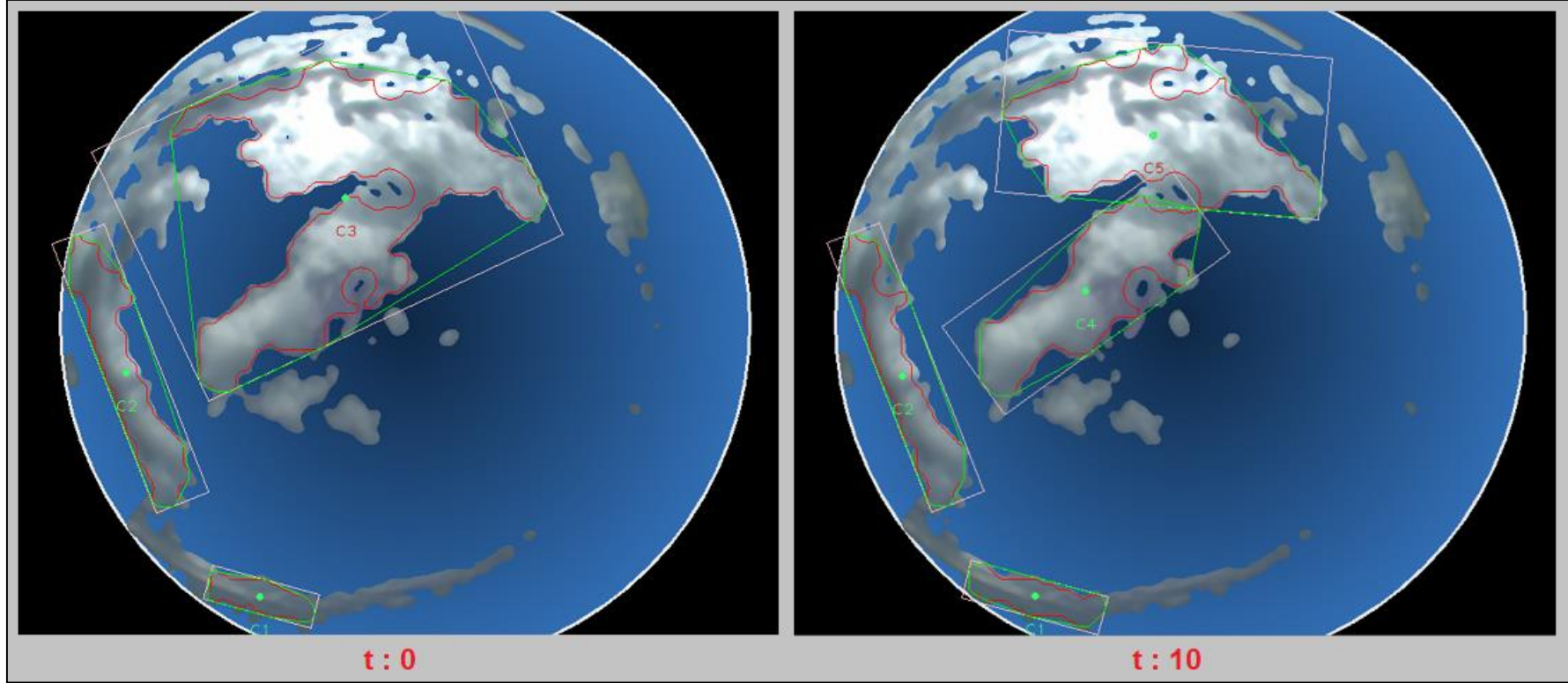
Çalışmanın bu aşamasında görüntüleme donanımından aktarılan her bir çerçeve için tanımlanan bulutlara dair bilgiler elde edilmiş bulunmaktadır. Her bir çerçeveye dair elde edilen bu bulut tanım bilgileri işlenerek, çerçeveler arasında bulutların eşlemeleri yapılabilir ve geometrik farklarından yola çıkılarak bulut hareketlerinin takibi gerçekleştirilebilir görülmektedir.

4.4.1. Çerçeveler Arası Bulut Eşleme

Tanımlanan ve etiketlenmiş bulut çevritlerine dair verilerin programatik açıdan bir yığın (struct) olarak saklanması ile her yeni bir çerçeve işlenirken bir önceki çerçeveye dair veriler karşılaştırılarak bulut çevritlerinin eşlemesi sağlanabilir ve dolayısıyla bulut hareketlerinin takibi mümkün kılınabilir. Söz konusu eşlemenin çevrit geometrilerinin benzerliği, çevrit ağırlık merkezlerinin yakınlığı, çevrit yüzey alanlarının benzerliği ve çevrit sınırlayıcı kutusunun (bounding box) konum benzerliği gibi çevrit niteliklerinin belirli sınırlar dahilinde örtüşüp örtüşmediğine bakılarak basitçe yapılabileceği görülmüştür. Örtüşmenin olmadığı durumlarda yeni çerçevede tanımlanan çevritler yeni birer bulut gibi ele alınabilirler ve daha sonraki çerçeveler için takip algoritması dahilinde işlenmeye devam edilebilirler.

Gökyüzünde bulutların bir birleri üzerine hareketleri söz konusu olduğunda takip edilen bulut çevritleri birleşerek tek bir bulut olarak takip edilmeye devam edilebilir, ya da benzer şekilde bir birlerinden ayrılan bulutlar da iki ayrı bulut parçası olarak tekrar takibe alınabilirler.

Şekil-21’de aralarında belirli bir fark olması amacıyla görüntüleme donanımından “10 saniye” aralıkla alınmış iki ayrı çerçevedeki tanımlanan bulut çevritlerinin sınırlayıcı kutuları, ağırlık noktaları vb. niteliklerinin bir birlerine olan benzerliği gösterilmiştir.



Şekil 21: “ $t:0$ ” ve “ $t:10$ ” anlarında bulut takibi.

Şekil-21’de, “ $t : 0$ ” anında “ $C1$ ”, “ $C2$ ” ve “ $C3$ ” olmak üzere tespit edilmiş üç ayrı bulutun takip edildiği, görüntü üzerinde kırmızı çizgiler ile çevrit çerperleri, yeşil nokta ile ağırlık merkezleri, yeşil çokgen ile dışbükey örtüleri ve pembe dikdörtgen ile çevrit sınırlayıcı kutuları gösterilmiştir. “ $t : 10$ ” anında ise 10 saniye sonrasında gelen bir çerçeve ile o ana kadar takip edilen “ $C1$ ” ve “ $C2$ ” bulutları eşlenmiş fakat “ $C3$ ” bulutunun parçalanmış olması sebebiyle eşlenemeyip, “ $C4$ ” ve “ $C5$ ” isimleriyle iki farklı bulut olarak tespit edilerek takibe alındığı gösterilmiştir. Aşağıda yer alan fonksiyon iki ayrı bulut çevritini analiz ederek, niteliklerinin uyumuna göre eşleşme olup olmadığına dair sonuç üretmektedir ve çalışmanın çerçeveler arası bulut eşleme algoritması dahilinde kodlanmıştır.

```
...
bool SkyAnalyzer::checkCloudMatch(const CloudDefinition &_c, const
CloudDefinition &_other)
{
    // bounding box check
    if(_c.centerPoint.x < _other.boundingRect.x ||
        _c.centerPoint.y < _other.boundingRect.y ||
        _c.centerPoint.x > (_other.boundingRect.x +
        _other.boundingRect.width) ||
        _c.centerPoint.y > (_other.boundingRect.y +
        _other.boundingRect.height))
        return false;

    // area check
    double areaUp = _other.area + (_other.area * AREA_DIFF_MUL);
    double areaDown = _other.area - (_other.area * AREA_DIFF_MUL);
    if(_c.area > areaUp || _c.area < areaDown)
        return false;

    // center point check
    cv::Point p = _other.centerPoint - _c.centerPoint;
    if(std::abs(p.x) > CENTER_DIFF || std::abs(p.y) >
CENTER_DIFF)
        return false;

    return true;
}
...
```

Kodlamada yer alan “ $AREA_DIFF_MUL$ ” ve “ $CENTER_DIFF$ ” sabitlerinin sırasıyla “1.2” ve “22” değerleri ile ele alınması test ve gözlemler neticesinde uygun bulunmuştur.

4.4.2. Bulutların Hareket Yönlerinin ve Hızlarının Belirlenmesi

Her bir çerçeve için bulut çevritlerinin tanımlanması ve önceki çerçeveler ile bu çevritlerin eşlenmesi neticesinde bulut çevritleri hareketsetel açıdan takip edilebilir bir hal almaktadır. Takibe alınan bulut çevritlerinin programatik analizleri neticesinde çerçeveler gelmeye devam ettikçe temel olarak ağırlık merkezlerinin takibi ile hareket yönleri ve ufuktan ufuğa gökyüzünün uzaklığı ile hareket hızları kabaca belirlenebilir.

Bulut çevritlerinin tespitindeki ve ayrıştırılmalarındaki hata payı ağırlık merkezlerinin de yanıltıcı sonuçlar üretmesine sebep olabilir. Bu sebeple bulut hareket yönleri hesaplanırken bir süre örnekleme (sampling) için zaman ayırmak ve ağırlık merkezlerinin ortalamaları üzerinden hareket yönlerinin ve hareket hızlarının hesaplanması uygun bulunmuştur. Aşağıda yer alan kod parçası bu işlem yerine getirilmektedir:

```
...
double xMean = 0.0, yMean = 0.0;
if(_c->centerPointHistory.size() > 0)
{
    for(cv::vector<cv::Point>::iterator it_oh = _c-
>centerPointHistory.begin(); it_oh != _c->centerPointHistory.end();
++it_oh)
    {
        xMean += it_oh->x;
        yMean += it_oh->y;
    }
    xMean /= (double)(_c->centerPointHistory.size());
    yMean /= (double)(_c->centerPointHistory.size());

    double pixDiffBySec = cv::norm(_c->centerPoint -
cv::Point(xMean, yMean)) / (1.0 / (double)mAnalysisParams->valFPS);
    double speedMean = pixDiffBySec > 0.0 ? (pixDiffBySec *
((double)R / (double)R_KM)) : 0.0; // m/sec
    double dirMean = std::atan2(yMean - (double)_c-
>centerPoint.y, xMean - (double)_c->centerPoint.x);

    _c->movementSpeed = speedMean;
    _c->movementDirection = dirMean;
}
...
```

Şekil-22’de “ $t : 50$ ” ve “ $t : 100$ ”, Şekil-23’de “ $t : 150$ ” ve “ $t : 200$ ”, Şekil-24’de “ $t : 250$ ” ve “ $t : 300$ ” ve Şekil-25’de “ $t : 350$ ” ve “ $t : 400$ ” anlarında takibe alınmış bulutlar üzerine hesaplanan hareket yönleri bir ok işareti ile gösterilmiş ve zamana bağlı değişim gösteren ağırlık merkez değişimleri de siyah çizgiler halinde çizilerek ifade edilmiştir.

Bulutların Şekil-22, Şekil-23, Şekil-24, ve Şekil-25’de yer alan ağırlık merkezi geçmişi ve yön oklarına dair çizimler aşağıdaki fonksiyon aracılığı ile sonuç görüntü üzerine oluşturulmaktadır:

```
...
void SkyAnalyzer::performCloudTrackDrawings(CloudDefinition *_c,
cv::Mat *_resultImage)
{
    bool isMoving = _c->movementSpeed > 0;

    // draw history lines
    for(cv::vector<cv::Point>::iterator it_oh =
        _c->centerPointHistory.begin();
        it_oh != _c->centerPointHistory.end();
        ++it_oh)
    {
        if(it_oh != _c->centerPointHistory.begin()
            && mAnalysisParams->drawCloudHistories)
        {
            cv::line(*_resultImage, *(it_oh - 1), *it_oh,
                mAnalysisParams->colorCloudHistories, 1);
        }
    }

    // draw center of mass
    if(mAnalysisParams->drawCloudCenterOfMasses)
        cv::circle(*_resultImage, _c->centerPoint, 2,
            mAnalysisParams->colorCloudTexts, 2);

    // draw cloud movement direction
    if(isMoving && _c->centerPointHistory.size() > 0)
    {
        // dir arrow body
        double xC = -mAnalysisParams->valCloudDirVect
            * std::cos(_c->movementDirection);
        double yC = -mAnalysisParams->valCloudDirVect
            * std::sin(_c->movementDirection);
        cv::Point pDirEnd(_c->centerPoint.x + (int)xC,
            _c->centerPoint.y + (int)yC);
        cv::line(*_resultImage, _c->centerPoint,
            pDirEnd, mAnalysisParams->colorCloudTexts, 1);
    }
}
```

```

// dir arrow L
double arwLxC = -mAnalysisParams-> valCloudDirVect /
    2.5 * std::cos(_c->movementDirection + 10.0);
double arwLyC = -mAnalysisParams-> valCloudDirVect /
    2.5 * std::sin(_c->movementDirection + 10.0);

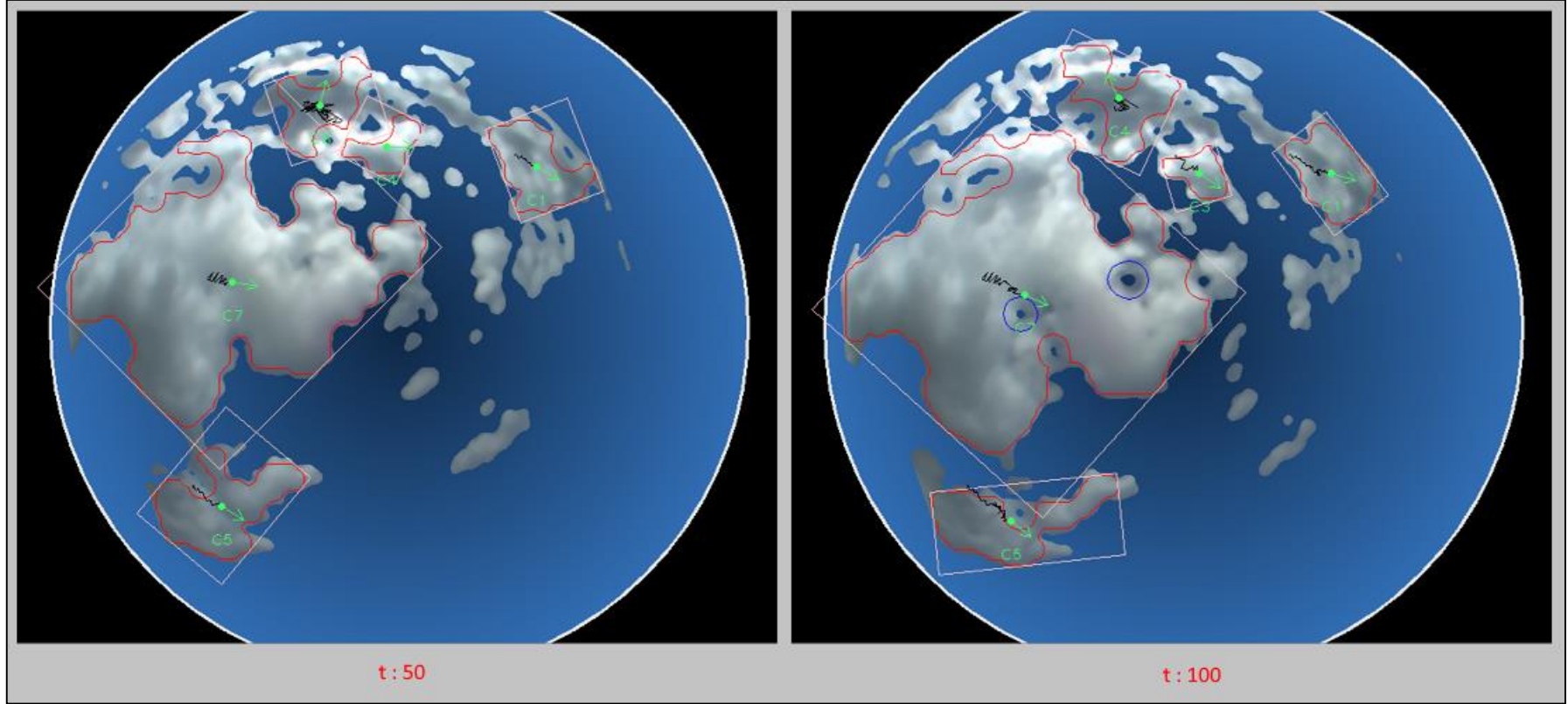
cv::line(
    *_resultImage,
    pDirEnd,
    cv::Point(pDirEnd.x + arwLxC,
        pDirEnd.y + arwLyC),
    mAnalysisParams->colorCloudTexts,
    1
);

// dir arrow R
double arwRxC = -mAnalysisParams-> valCloudDirVect /
    2.5 * std::cos(_c->movementDirection - 10.0);
double arwRyC = mAnalysisParams-> valCloudDirVect /
    2.5 * std::sin(_c->movementDirection - 10.0);

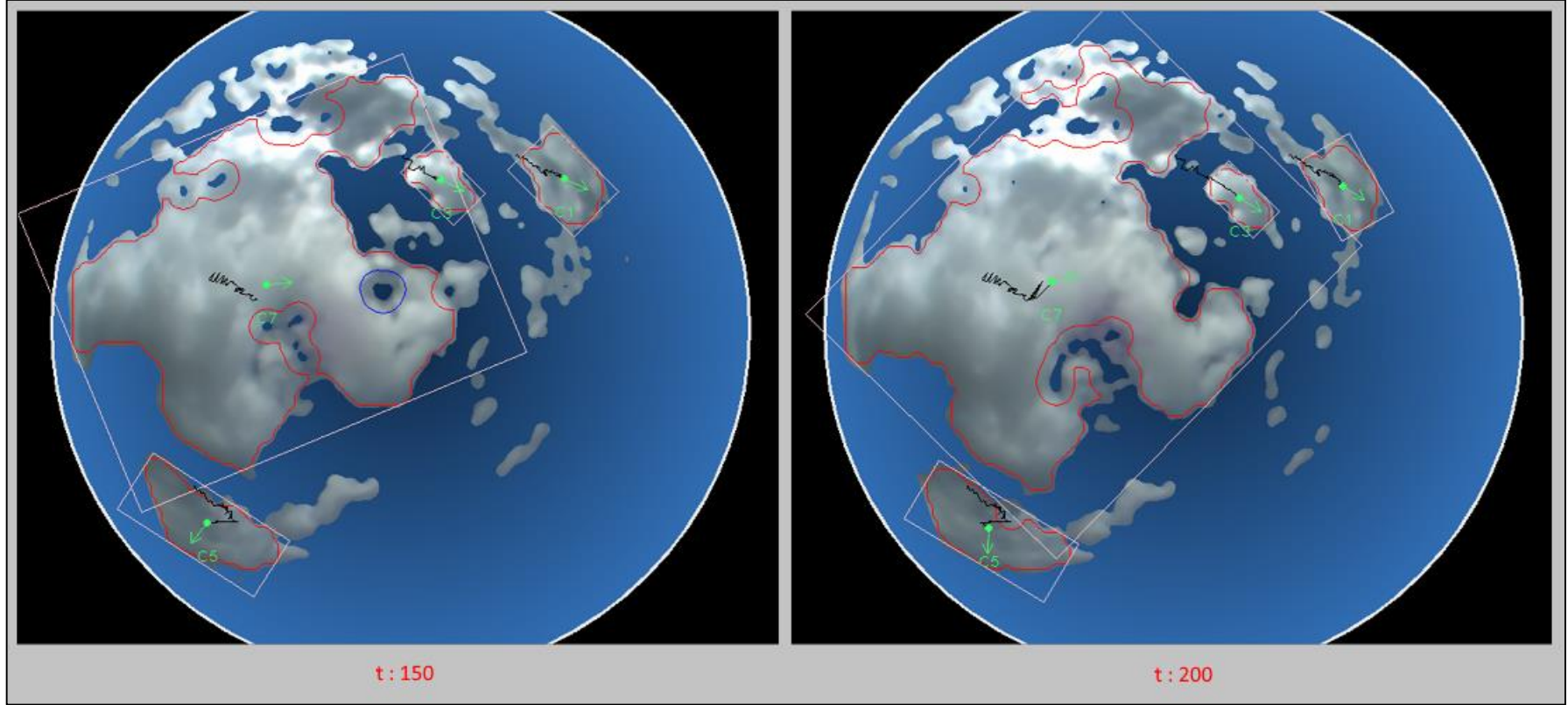
cv::line(
    *_resultImage,
    pDirEnd,
    cv::Point(pDirEnd.x + arwRxC,
        pDirEnd.y + arwRyC),
    mAnalysisParams->colorCloudTexts, 1);
}

}
...

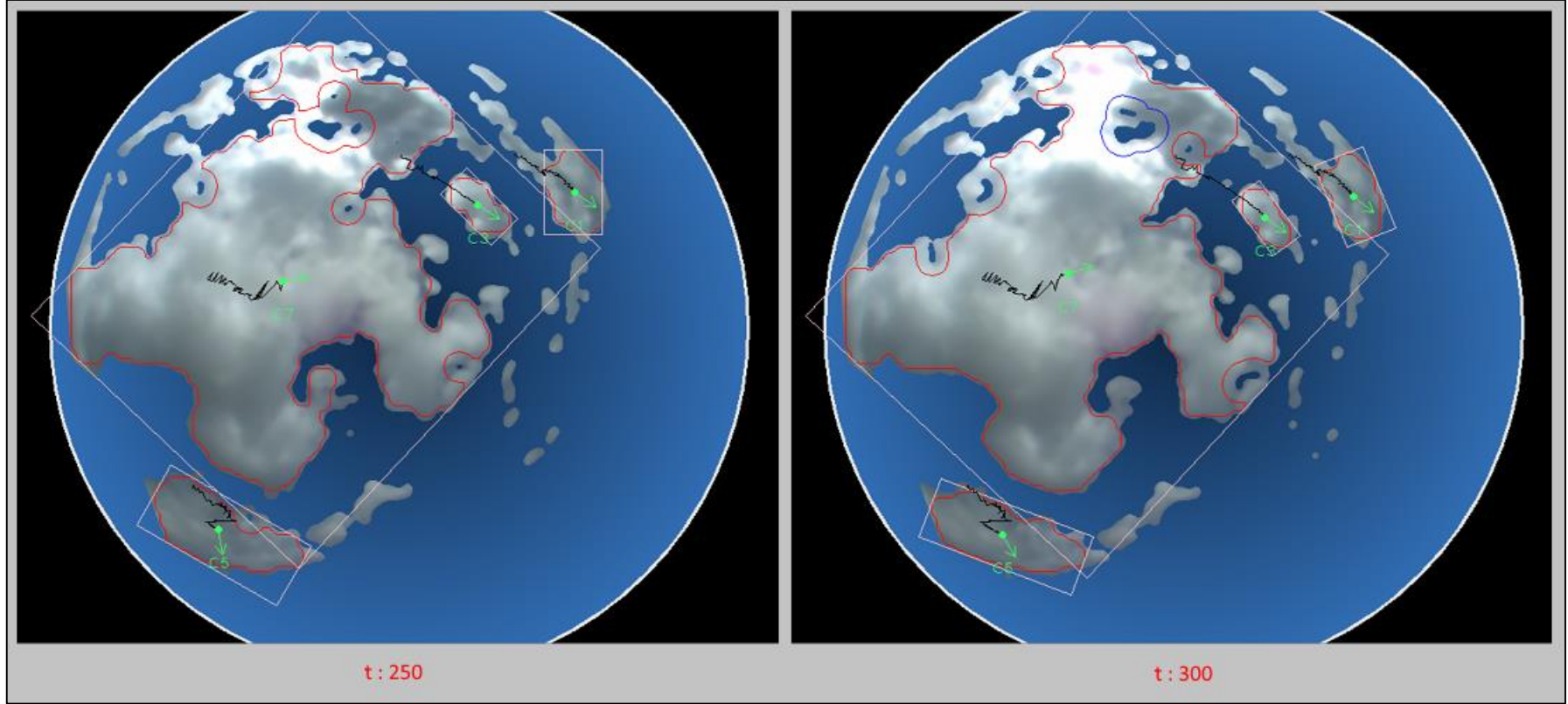
```



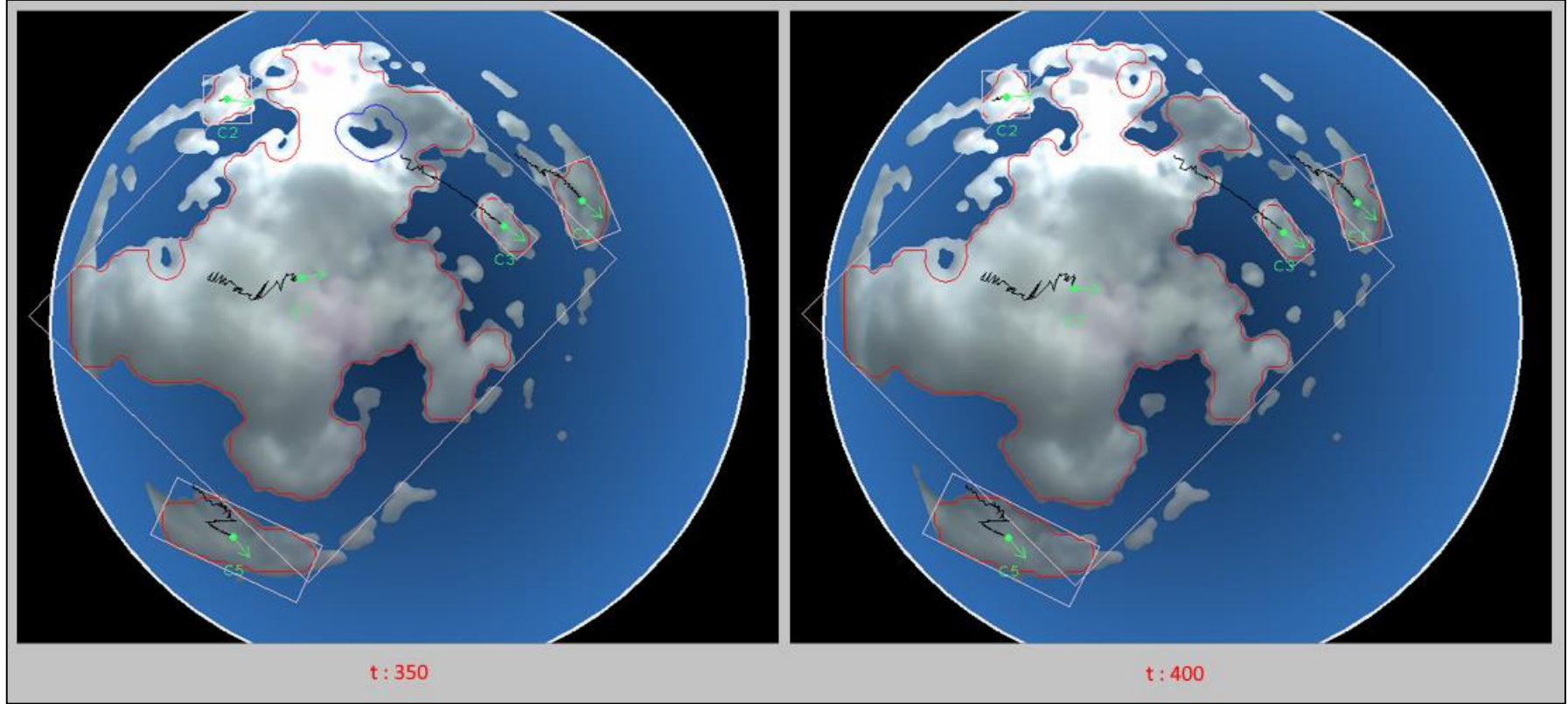
Şekil-22: Bulutların hareket yönlerinin belirlenmesi. (" $t: 50$ " - " $t: 100$ ")



Şekil-23: Bulutların hareket yönlerinin belirlenmesi. (" $t: 150$ " - " $t: 300$ ")



Şekil-24: Bulutların hareket yönlerinin belirlenmesi. (" $t: 250$ " - " $t: 300$ ")



Şekil-25: Bulutların hareket yönlerinin belirlenmesi. (“ $t: 350$ ” - “ $t: 400$ ”)

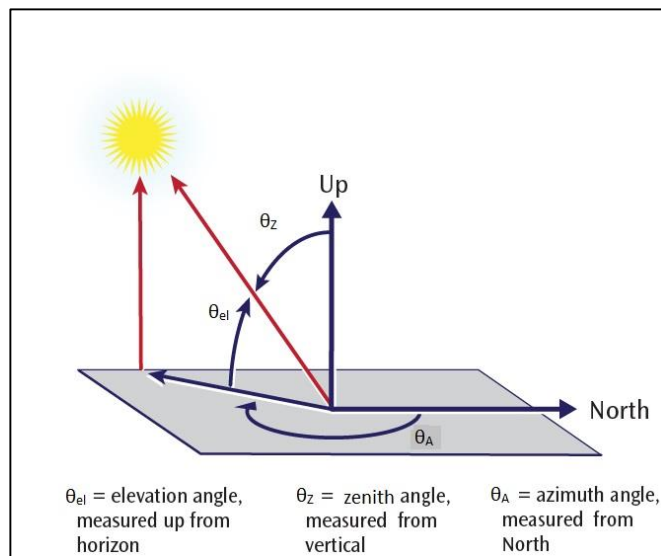
4.5. Bulutların Güneş Kapatma Analizi

Çalışmanın gelinen bu aşamasında, görüntüleme donanımı ile alınan görüntü üzerindeki bulutlar tanımlanabilir, takip edilebilir ve hareketlerine dair yön - hız bilgileri de hesaplanabilir durumdadır. Bu veriler ışığında bulutların güneş ışığını keseceğini önceden tahmin edebilmek de güneşin görüntü üzerindeki konumunun net olarak belirlenebilmesi ile mümkün kılınabilir.

4.5.1. Güneş Konumunun ve Rotasının Hesaplanması

Çalışma sırasındaki araştırmalar ile sistemin çalıştırılacağı coğrafi konumun koordinatlarına göre arzu edilen zamana dair güneşin gökyüzündeki konumunun açısıl olarak neredeyse kesin bir şekilde belirlenebilir olduğu anlaşılmıştır.

Güneşin konumunu matematiksel olarak belirlemek karmaşık birçok hesaplama ve koordinat sistemleri arasında dönüşümlerin gerçekleştirilmesi ile mümkün olmaktadır. [22] NREL (National Renewable Energy Laboratory) tarafından geliştirilen SPA'nın, açık kaynak C kodu ile yayımlanmış olması sebebiyle bu çalışmada kullanılması uygun görülmüştür. Bu algoritma ile belirli bir konuma dair enlem, boylam ve zaman değeri girdi olarak verilerek, güneşin o konuma olan ufuk (azimuth) ve yükseliş (elevation) açıları çıktı olarak alınabilir. [22] (bkz. Şekil-26)



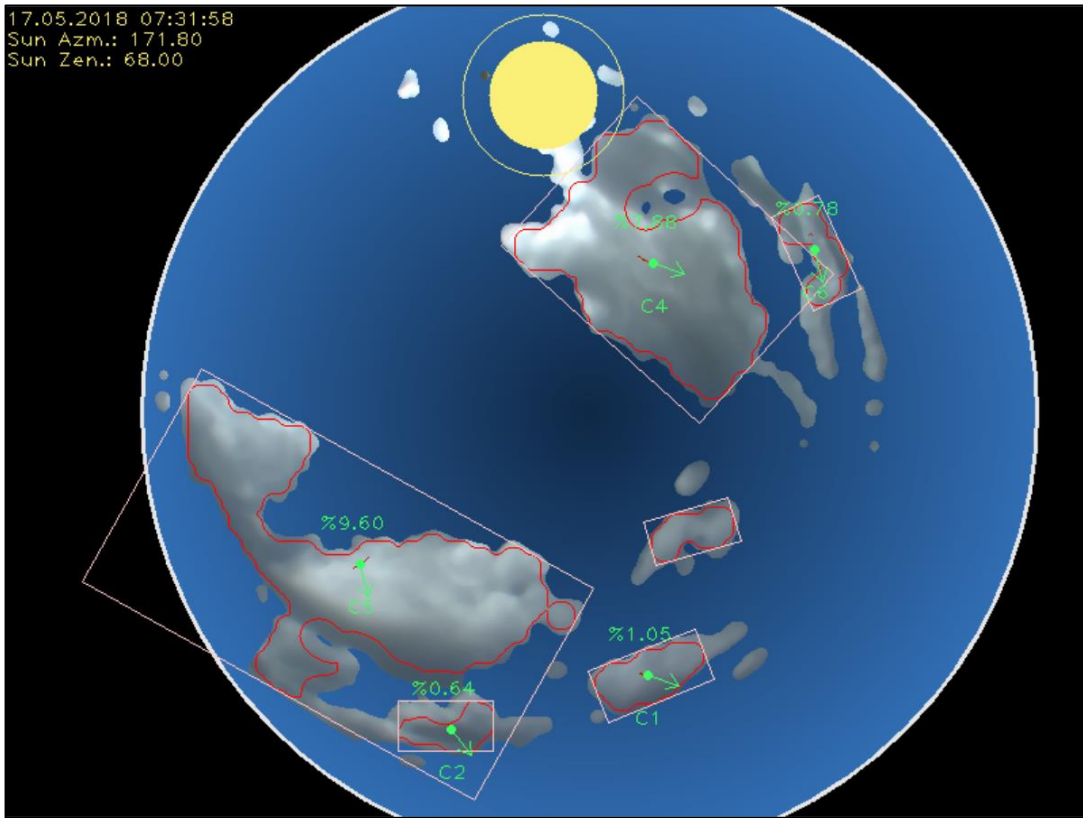
Şekil-26: Güneşin ufuk ve yükseliş açıları. [22]

Güneşin konum ve zamana bağlı olarak alınacak o anlık ufuk ve yükseliş açıları ile güneşin koordinatsal olarak görüntü üzerinde tam olarak nerede olacağı aşağıda yer alan denklemler ile kolayca hesaplanabilir.

$$Sun\ X = cX + \left(\frac{-r}{95} (90 - el) \sin\left(az \frac{\pi}{180}\right)\right)$$

$$Sun\ Y = cY + \left(\frac{r}{95} (90 - el) \cos\left(az \frac{\pi}{180}\right)\right)$$

Denklemlerde yer alan cX ile cY FOV merkezinin koordinatlarını, r FOV yarıçapını ve az ile el güneşin o anlık ufuk ile yükseliş açılarını ifade etmektedir. Şekil-27’de bu denklemler kullanılarak yapılan hesaplama ile güneş bölgesinin görüntü üzerine sarı bir daire olarak konumlandırılması gösterilmiştir.



Şekil-27: Güneşin görüntü üzerinde konumlandırılması.

Çalışmanın ilerleyen aşamalarında yapılacak olan bulutların güneşi kapatmasına dair analizi gerçekleştirebilmek için bulutların güneş bölgesine hareketi sırasında geçecek süre zarfında güneşin de hareket edecek olmasıyla gün içindeki güneş

hareket rotasının hesaplanmasına ihtiyaç duyulacaktır. Bu amaç ile gün başından başlayarak, gün sonuna kadar her bir dakika için SPA'nın işletilmesiyle zaman bazında güneş açıları elde edilerek yukarıda ifade edilen koordinat hesaplama denklemi aracılığı ile görüntü üzerinde güneşin gün içinde seyredeceği rota koordinatsal olarak hesaplanabilir. Şekil-28'de gün içindeki güneş hareket rotası sarı çizgiler ile çizdirilerek ifade edilmiştir, bu işlemi yerine getiren kodlama aşağıda yer almaktadır.

```
...
// calculate sun path
mSunPath.clear();
QDateTime pathTime = QDateTime::currentDateTime();
pathTime.setTime(QTime());
QDateTime pathTimeEnd = pathTime.addDays(1);

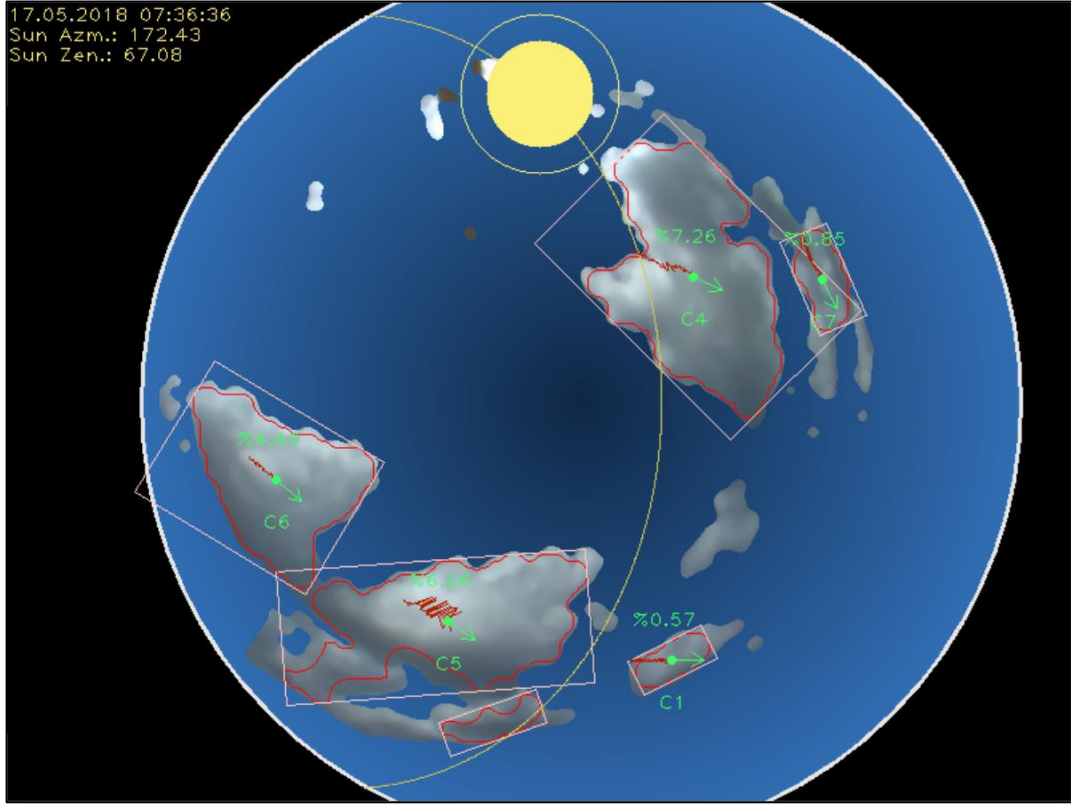
while(pathTime < pathTimeEnd)
{
    SphericalPosition spa_sun =
        SolarPosition::getSunPosition(pathTime, Location(LON,
        LAT, ALT));

    spa_sun.Azimuth += AZ_OFFSET;
    mSunPath.push_back(cv::Point(CENTER_X + (-R / 95.0 * (90.0 -
        spa_sun.Elevation) * std::sin((spa_sun.Azimuth) * M_PI
        / 180.0)), CENTER_Y + (R / 95.0 * (90.0 -
        spa_sun.Elevation) * std::cos((spa_sun.Azimuth) * M_PI
        / 180.0))));
    pathTime = pathTime.addSecs(60);
}

...

// draw sun path
for (size_t i = 0; i < mSunPath.size(); i += 2)
{
    cv::line(res.resultImage,
        mSunPath[i],
        mSunPath[i + 1],
        cv::Scalar(80, 200, 210) // yellow
        1);
}

...
```



Şekil-28: Güneş rotasının görüntü üzerinde konumlandırılması.

4.5.2. Bulut ve Güneş Çarpışmalarının Belirlenmesi

Çalışmanın bu aşamasında takibi gerçekleştirilen bulutların rotası boyunca ilerleyen güneş ile çarpışmaları belirlenebilir durumdadır. Hız ve yön bilgisi elde bulunan bulut nesnelerine ait çevritlerin hızları da dikkate alınarak görüntü üzerinde hareket yönlerince kaydırılması neticesinde bulut bölgesine olası temasları belirlenebilir.

Aşağıda yer alan kod parçası bir bulut çevritini hareket yönünce kaydırarak güneş bölgesi ile bir temas gerçekleştiğinde bunu görüntü üzerine çizdirmektedir ve bu çarpışmanın ne kadar süre sonra oluşacağını hesaplamaktadır.

```
...  
// create shift vector by movement direction  
cv::Point shiftVector(  
    (int)(xMean - ((double)_c->centerPoint.x + -mAnalysisParams->  
        valCloudDirectionVector / 2.0 * std::cos(_c-
```

```

        >movementDirection))), (int)(yMean - ((double)_c-
        >centerPoint.y + -mAnalysisParams->valCloudDirectionVector /
        2.0 * std::sin(_c->movementDirection)))
    );

// collision assumed sun parameters
int blockTimeEst = cv::norm(mSunPosition - _c->centerPoint)
    / _c->movementSpeed;
SphericalPosition spSunPositionEst = SolarPosition::getSunPosition(
    _trackStartTime.addSecs(blockTimeEst * 60),
    Location(LON, LAT, ALT));

double sunDistanceEst = R / 95.0 * (90.0 -
    getSunPositionEst.Elevation);
double sunShiftXEst = -sunDistanceEst *
    std::sin((spSunPositionEst.Azimuth) * M_PI / 180.0);
double sunShiftYEst = sunDistanceEst *
    std::cos((spSunPositionEst.Azimuth) * M_PI / 180.0);
cv::Point sunPosEst = cv::Point(CENTER_X + sunShiftXEst,
    CENTER_Y + sunShiftYEst);

// shift until the sun collided or the vision circle reached
cv::vector<cv::Point> contour(_c->contour.begin(),
    _c->contour.end()), collisionPath;
cv::Point *contCenter;
cv::Moments *contMoments;
double distance = 0;
while(distance <= R)
{
    contMoments = &cv::moments(contour);
    contCenter = &cv::Point((int)(contMoments->m10 / contMoments-
        >m00), (int)(contMoments->m01 / contMoments->m00));
    distance = std::sqrt(std::pow((double)(mAnalysisParams-
        >valVisionCircle[0] - contCenter->x), 2) +
        std::pow((double)(mAnalysisParams->valVisionCircle[1] -
        contCenter->y), 2));
    collisionPath.push_back(*contCenter);

    // shift the contour points
    for (size_t i = 0; i < contour.size(); i++)
    {
        contour[i].x -= shiftVector.x;
        contour[i].y -= shiftVector.y;
    }

    // collision check
    if (cv::pointPolygonTest(contour, sunPosEst, false) > -1)
    {
        // draw collision
        cv::line(_resultImage,
            _c->centerPoint,
            sunPosEst,

```

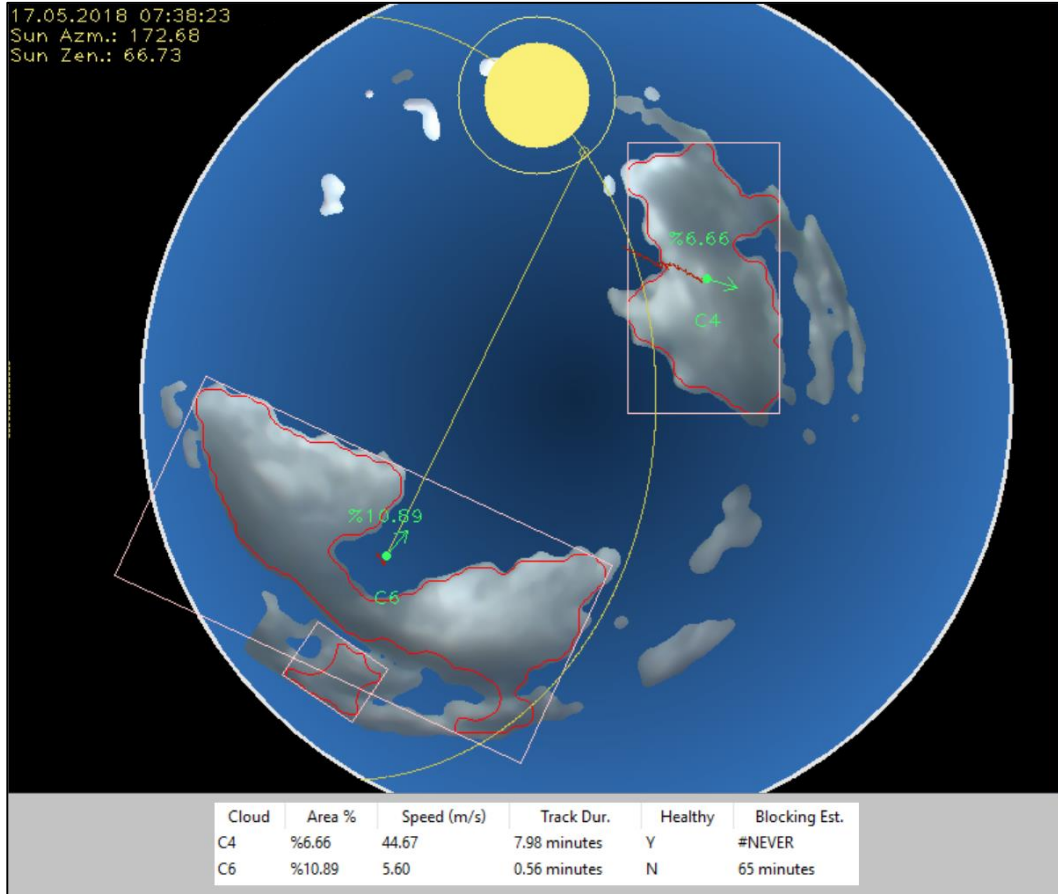
```

        mAnalysisParams->colorSunPath
    );
    cv::circle(*_resultImage,
        sunPosEst,
        4,
        mAnalysisParams->colorSunPath
    );

    _c->sunBlockEstimation = blockTimeEst;
    return; // termination
}
...

```

Şekil-29’da güneşi kapatacağı belirlenen bulutların ağırlık merkezinden, güneş rotası üzerindeki ilk temas noktasına bir doğru çizdirilerek güneşi kapatacağı ifade edilmiştir ve şeklin altında bulutların ne kadar süre sonra bu teması gerçekleştireceğinin analizine dair sonuçlar da yer almaktadır.



Şekil-29: Bulut-Güneş çarpışma analizi.

Bulutların görüntü üzerinden gerçekte ne kadar irtifada bulunduğu kesin olarak bilinemeyeceği için hızları da yaklaşık olarak hesaplanmak durumundadır. Ancak görsel olarak bir bulut çevritinin görüntü üzerinde birim zamanda ne kadar yol aldığı sağlıklı biçimde hesaplanabilir ise güneş çarpışmasına dair yapılacak süre tahmini de o denli sağlıklı olacaktır. Bu bağlamda her bir yeni çerçevenin işlenmesiyle hesaplanan bulut çevritinin güneş kapatma sürelerinin bir güven faktörüne bağlanması uygun görülmüştür. Bu güven faktörü ise aşağıdaki kod parçası ile her bir bulut çevritinin geçmiş güneş kapatma sürelerinin standart sapma dahilinde değişim göstermesine bağlanmıştır.

```
...
// update sun block estimation history
cd->sunBlockEstimationHistory.push_back(cd->sunBlockEstimation);
if(cd->sunBlockEstimationHistory.size() > 200)
    cd->sunBlockEstimationHistory.erase(
        cd->sunBlockEstimationHistory.begin());

// estimation confidence
if(cd->sunBlockEstimationHistory.size() > 1)
{
    double mean = 0.0, stdev = 0.0;
    for(cv::vector<int>::iterator it =
        cd->sunBlockEstimationHistory.begin();
        it != cd->sunBlockEstimationHistory.end(); ++it)
        mean += (*it + 2);
    mean /= (double)cd->sunBlockEstimationHistory.size();

    for(cv::vector<int>::iterator it =
        cd->sunBlockEstimationHistory.begin(); it != cd-
        >sunBlockEstimationHistory.end(); ++it)
    {
        double d = mean - (*it + 2);
        stdev += d * d;
    }
    stdev = std::sqrt(stdev /
        (double)(cd->sunBlockEstimationHistory.size() - 1));

    cd->isSunBlockEstimationConfident =
        ((double)cd->sunBlockEstimation + 2.0 >= mean - stdev)
        &&
        ((double)cd->sunBlockEstimation + 2.0 <= mean + stdev);
}
else
    cd->isSunBlockEstimationConfident = false;
...
```

5. UYGULAMA GELİŞTİRME

Gökyüzü görüntüsü girdisi ile programatik olarak bulutların tanımlanması, takip edilmesi ve güneş kapatma analizlerinin yapılabilmesi bu çalışmaya dair amaca yönelik bir uygulama geliştirilmesini mümkün kılmaktadır.

Bu çalışmanın ortaya koyduğu teknikler üzerine geliştirilecek bir uygulama ile solar enerji tesislerinde olağan veya olası güneş ışığının kesintilerine dair tesisteki diğer sistemlere entegre bir erken uyarı sistemi ihtiyacı karşılanabilir.

5.1. Uygulamanın Sistem Gereksinimleri

Görüntü işleme operasyonlarının gerçek zamanlı olarak yürütülebilmesi için uygulamanın çalıştırılacağı sisteme dair bir takım gereksinimler bulunmaktadır.

Donanımsal bakımdan günümüz işlemcilerinden “*Intel – i5*” veya üzeri niteliklerde herhangi bir işlemci gücüne sahip, görüntüleme donanımının bulunduğu ağa bağlanmak için gerekli olan ethernet kartı barındıran herhangi bir PC bu ihtiyacı karşılamada yeterli olacaktır.

Çalışma için tercih edilmiş olan görüntüleme donanımından görüntü aktarımını sağlayacak olan “*pylon SDK*” ve sürücüleri “*Windows*”, “*Linux*” ve “*OSX*” platformları üzerinde işlevini yerine getirebilir. Görüntü işleme kütüphanesi olarak çalışmada kullanımı tercih “*OpenCV*” de “*Windows*” ve “*Linux*” platformunda kullanılabilir. Bu iki önemli kütüphanenin bir arada kullanılarak geliştirilecek uygulamanın kodlama dili en iyi performansı vaad etmesi sebebiyle C++ olarak belirlenmiş ve özellikle kullanıcı arayüzü gereksinimleri için cross-platform bir framework olan “*Qt*” kullanımı tercih edilmiştir. Böylece geliştirilecek uygulamanın hedeflenen platformu da “*Windows*” veya “*Linux*” olacaktır.

5.2. Uygulama ve Kullanıcı Arayüzünün Tasarımı

Geliştirilecek örnek uygulamanın solar tesis operatörlerince kolay kullanılabilir ve özelleştirilebilir bir arayüze sahip olması, farklı bulut tipleri için analiz parametreleri üzerinde ince ayar yapabilme imkanı sağlaması ve alarm üretme

koşullarını belirleyecek bir arayüzü içermesi gerekmektedir. Şekil-30'da "SkyVision" adıyla çalışma kapsamında geliştirilen uygulamanın arayüzü gösterilmektedir. Uygulama kamera ismi ile çalıştırıldığı sistemin ağındaki görüntüleme donanımına ve üretilecek erken uyarı alarmlarının iletileceği sisteme bağlantı kurarak işlevini yerine getirmeye başlamaktadır. Sol üst kısımda görüntüleme donanımından gelen orijinal görüntü üzerine maskeleme imkanı sağlamakta, sol alt kısımda "Analysis Parameters" sekmesi altında görüntü işlemeye dair ince ayar ve sağ kısımda bulunan görsel analiz çıktısı üzerinde görsel imgeleri değiştirme imkanı vermekte, "Alarms" sekmesinde ne koşullarda alarm üreteceğini belirleme imkanı vermekte ve "Analysis" sekmesinde tespit edilen bulutların takibine dair anlık bilgileri kullanıcısına sunmaktadır.

Uygulamanın diğer sistemler ile entegrasyonunun TCP/IP ağ soketi aracılığı ile yapılması düşünülmüştür. Böylece uygulamadan alarm bekleme ihtiyacında olan bir sisteme ilgili IP ve port üzerinden ağ istemcisi olarak bağlanılarak gerekli durumlarda ilgili alarmlar iletilebilir.

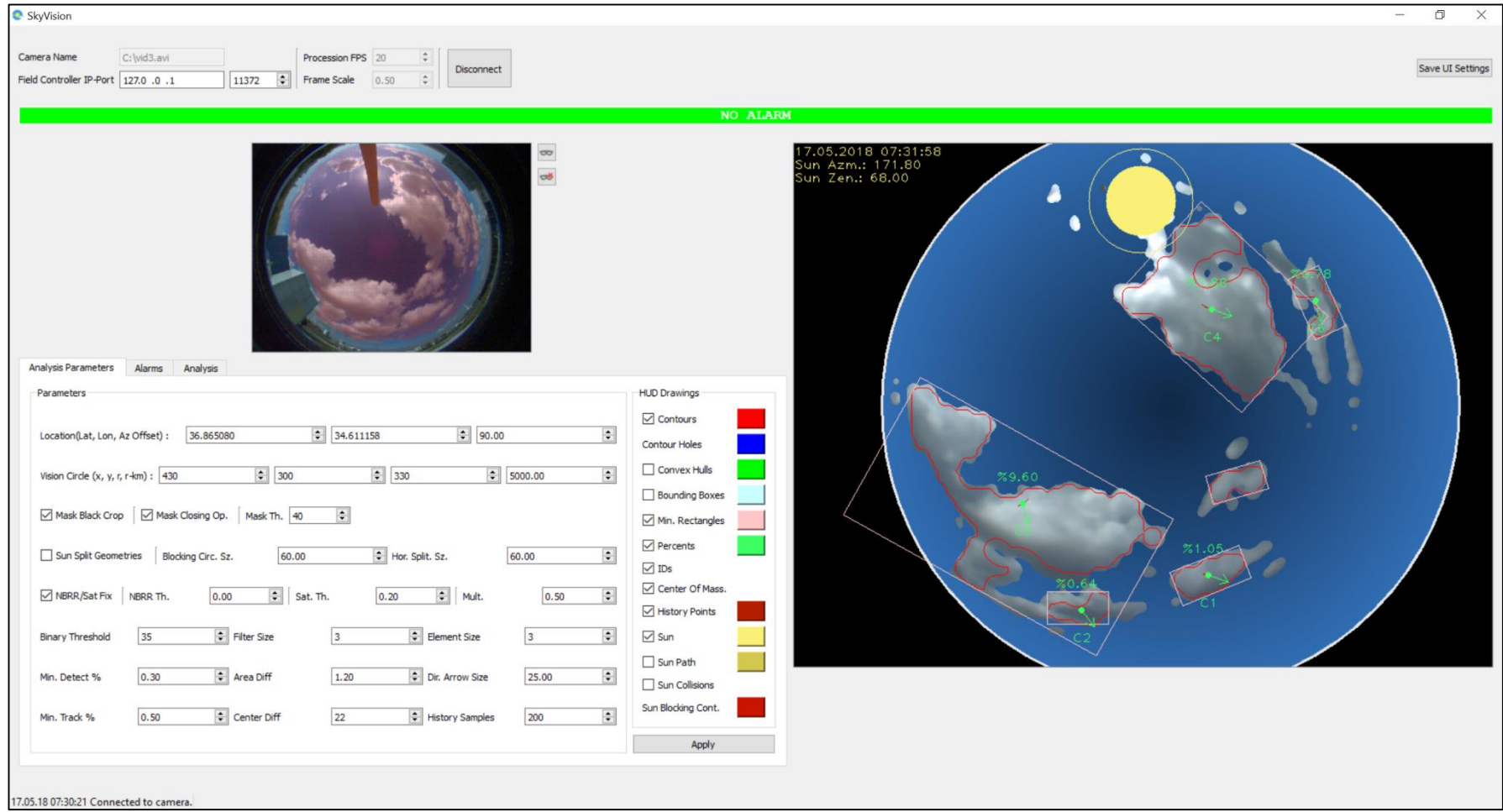
Bulutların analiz sonuçlarına göre alarm üretme koşullarını tanımlamak için de ayrıca bir arayüz geliştirilmesi düşünülmüştür. Şekil-31'de alarm koşulları tanımlama arayüzü ve bir alarm üretimi gösterilmiştir. Arayüzde entegre çalışan sistemin gereksinimlerinden ileri gelen biçimiyle "Yellow Alarm" ve "Red Alarm" olarak iki ayrı alarm türüne dair üretim koşullar bulunmaktadır. Şekilde yer alan haliyle "Yellow Alarm" üretim koşulları,

- Bulutun güneş kapatma tahmini süresi 60 dakika altında ise,
- ve bulutun alanı en az %2 kadar ise,
- ve bulut en az 10 dakikadır takip halinde ise,

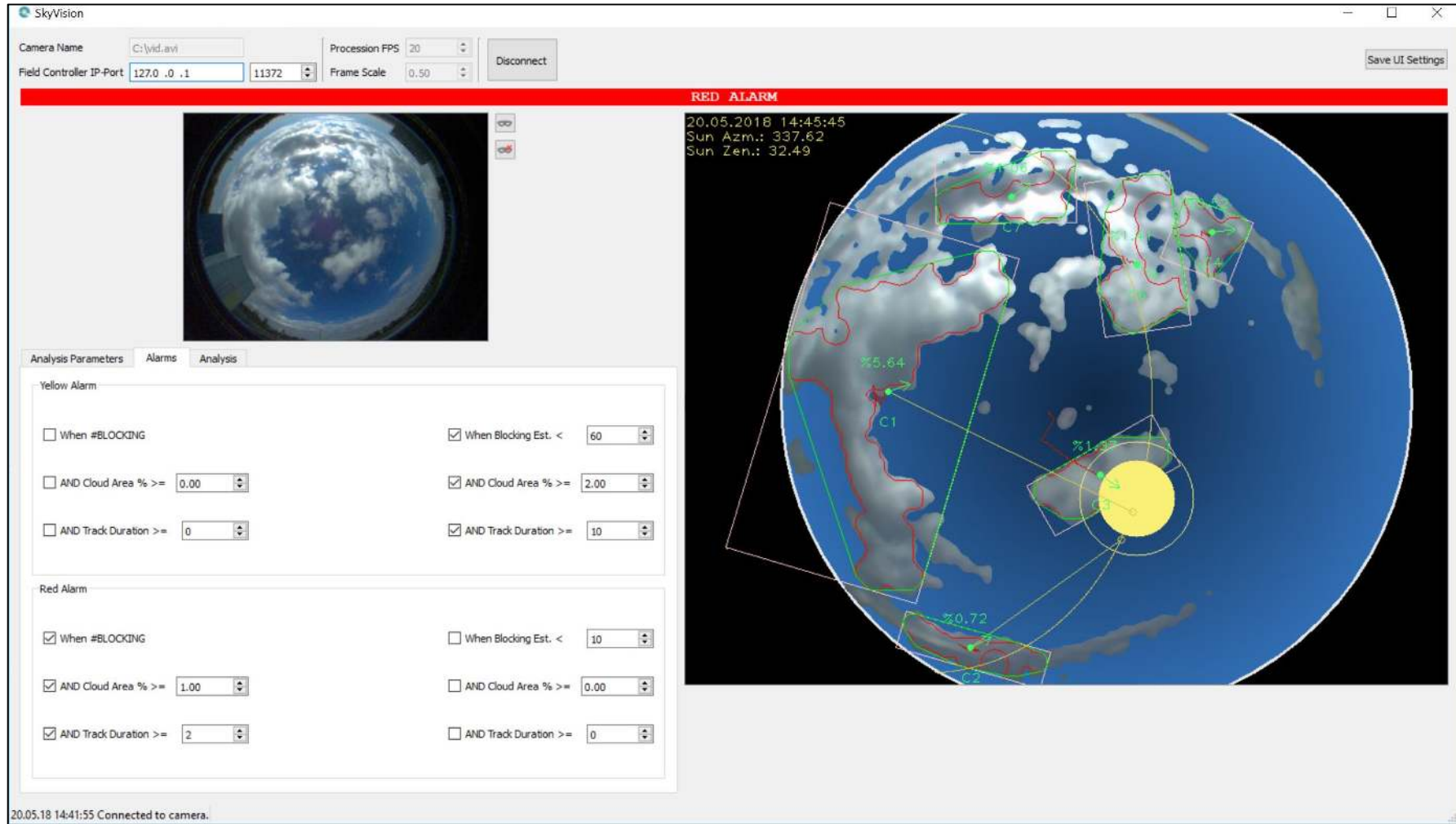
şeklinde ve "Red Alarm" üretim koşulları,

- Bulut güneşi kapatıyor ise,
- ve bulutun alanı en az %1 kadar ise,
- ve bulut en az 2 dakikadır takip halinde ise,

şeklindedir.



Şekil-30: Çalışma kapsamında geliştirilen uygulama ve arayüzü.



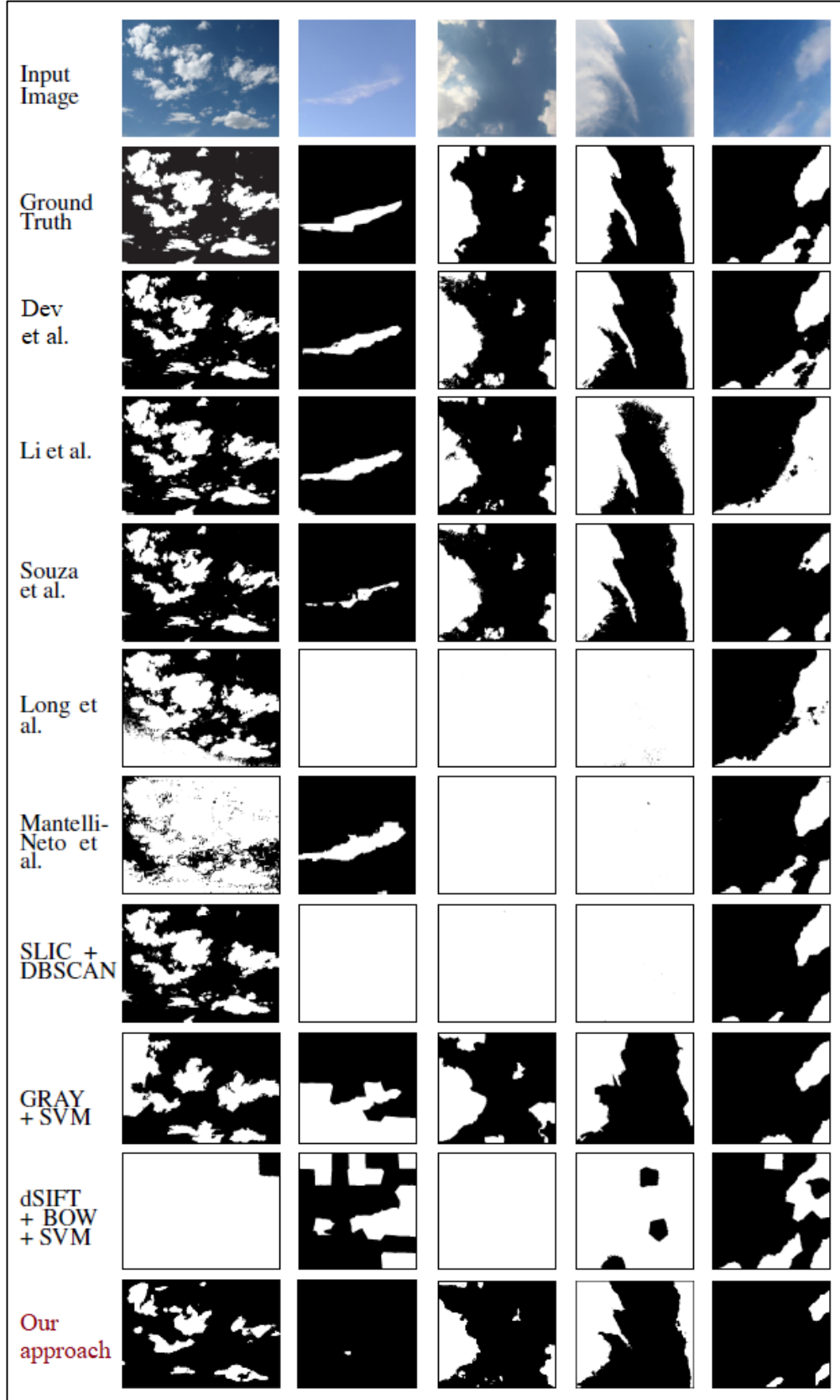
Şekil-31: Alarm koşulları tanımlama arayüzü ve “Red Alarm” durumu.

6. SONUÇ

Tez çalışması ile gökyüzü görüntülerinin görüntü işleme teknikleriyle analiz edilerek bulutların tanımlanabildiği, hareketlerinin takibinin gerçekleştirilebildiği ve bu hareketlere bağlı olarak güneş ışınlarını ne kadar süre sonra kesebileceğine dair bir erken uyarı sisteminin tasarlanabileceği ortaya konulmuştur.

Çalışmayı gerçeklemek üzere geliştirilen ve çalışmadaki bir çok şekil çıktısının elde edilmesini sağlayan “*SkyVision*” isimli örnek uygulama bir solar enerji tesisinde kullanıma sunulurken, ilgili tesis sistemlerine entegre şekilde yaklaşık altı ay boyunca test edilmiştir. Test süreci sonunda uygulama ortaya çıkarılmış haliyle işlevsel olarak yeterli bulunmuş ancak daha istikrarlı sonuçlar alınabilmesi adına kullanılmış olan yöntem ve tekniklerin geliştirilmesi amacıyla üzerinde çalışılmasına devam edilmesi faydalı görülmüştür. Testler sırasında algoritma dahilinde takibe alınan bulutların yön ve hız bilgilerinin dolayısıyla güneş kapatma sürelerinin verimli olarak hesaplanabilmesi yaklaşık 5 dakika kadar süreye ihtiyaç duyduğu ve özellikle parçalanıp tekrar birleşen bulutların her seferinde bu süreyi yeniden başlatması sebebiyle güneş kapatmasına dair süre tahmininin tutarlılığını düşürdüğü görülmüştür.

Çalışma kapsamında geliştirilen bulut tespit algoritmasının solar enerji tesisleri açısından iyi derece de optimize edilmiş olduğu sonucuna varılmıştır. Şekil-32’de benzer çalışmaların çıktılarının ve bu çalışmadaki yaklaşımın çıktısının (şekilde “Our approach” olarak ifade edilmiştir) kıyaslamaları gösterilmiştir. Bu kıyaslamaya göre bu çalışma kapsamında geliştirilen yaklaşım ile güneş kesintisi gerçekleştirmeyecek nitelikte incelikte bulutlar elemine edilmekteyken, güneşi kesme potansiyeli bulunan bulutlar benzer çalışma çıktılarına yakın sonuçlar ile ele alınmaktadır. Şekil-32’de yer alan örnek girdi görüntülerinin (şekilde “Input Image” olarak ifade edilmiştir) soldaki iki adedi HYTA’dan ve sağdaki üç adedi SWIMSEG’den elde edilmiş olup, düşük kaliteli görseller olmalarına rağmen algoritma tarafından başarıyla işlenebilmişlerdir. Öte yandan, benzer çalışmalarda sunulan yaklaşımların ağırlıklı olarak gerçek zamanlı çalışmaya uygun şekilde tasarlanmadıkları ve/veya maliyeti yükseltecek ek donanımlara (stereo kamera vb.) ihtiyaç duyduğu sonucuna varılmıştır.



Şekil-32: Bulut tanıma çıktısının benzer çalışmaların çıktıları ile karşılaştırması. [8]

KAYNAKÇA

- [1] Long C.N., Sabburg J.M., Calbó J. ve Pages D., **Retrieving cloud characteristics from ground-based daytime color all-sky images**, *Journal of Atmospheric and Oceanic Technology*, 2006, 23,5, 633–652.
- [2] Heinle A., Macke A. ve Srivastav A., **Automatic cloud classification of whole sky images**, *Atmospheric Measurement Techniques*, 2010, 3, 3, 557–567.
- [3] Liu S., Zhang L., Zhang Z., Wang C. ve Xiao B., **Automatic cloud detection for all-sky images using superpixel segmentation**, *IEEE Geoscience and Remote Sensing Letters*, 2010, 12, 2, 354–358.
- [4] Souza-Echer M. P, Pereira E.B., Bins L.S ve Andrade M.A.R, **A simple method for the assessment of the cloud cover state in high-latitude regions by a ground-based digital camera**, *Journal of Atmospheric and Oceanic Technology*, 2006, 3, 5, 437–447.
- [5] Mantelli-Neto S.L., von Wangenheim A., Pereira E.B, ve Comunello E., **The use of Euclidean Geometric Distance on RGB color space for the classification of sky and cloud patterns**, *Journal of Atmospheric and Oceanic Technology*, 2010, 27, 9, 1504–1517.
- [6] Li Q., Lu W. ve Yang J., **A hybrid thresholding algorithm for cloud detection on ground-based color images**, *Journal of Atmospheric and Oceanic Technology*, 2011, 28, 10, 1286–1296.
- [7] Yuan Y. ve Hu X., **Bag-of-words and object-based classification for cloud extraction from satellite imagery**, *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 2015, 8, 8, 4197–4205.
- [8] Dev S., Hui Y.L ve Winkler S., **Color-based Segmentation of Sky/Cloud Images From Ground-based Cameras**, *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 2016, 10, 1, 231–242.
- [9] Wood-Bradley P., Zapata J. ve Pye J., (2012), **Cloud tracking with optical flow for short-term solar forecasting**, *50th Conference of the Australian Solar Energy Society, Melbourne*.
- [10] Peng Z., Yu D., Huang D., Heiser J., Yoo S. ve Kalb P., **3D cloud detection and tracking system for solar forecast using multiple sky imagers**, *Sol Energy* 2015, 2014, 118, 496–519.
- [11] Chauvin R., Nou J., Thil S., Traoré A. ve Grieu S., (2015), **Cloud detection methodology based on a sky-imaging system**, *International Conference on Concentrating Solar Power and Chemical Energy Systems 2014*.
- [12] Reda I., Andreas A., (2003), **Solar Position Algorithm for Solar Radiation Applications**, *National Renewable Energy Laboratory*.

- [13] Nave R., (2016), **Blue Sky and Rayleigh Scattering**, <http://hyperphysics.phy-astr.gsu.edu/hbase/atmos/blusky.html>, 15 Mayıs 2018.
- [14] Meteoroloji Genel Müdürlüğü, **Bulutlar**, <https://www.mgm.gov.tr/genel/sss.aspx?s=bulutlar>, 15 Mayıs 2018
- [15] Seip W., (2012), **Can snow form "snowbows" similar to rainbows?**, <http://www.astronomy.com/magazine/ask-astro/2012/08/precipitation-phenomena>, 15 Mayıs 2018.
- [16] Fujifilm, **Fujinon FE185C057HA-1 CCTV Lens Datasheet**, http://fujifilm.com/products/optical_devices/pdf/cctv/fa/fisheye/fe185c057ha-1.pdf, 16 Mayıs 2018
- [17] Hustvedt, (2007), **CCTV Lens flare**, https://commons.wikimedia.org/wiki/File:CCTV_Lens_flare.jpg, 16 Mayıs 2018
- [18] Thryft A., (2009), **GigE Vision update heralds more changes**, <https://www.edn.com/electronics-news/4384922/GigE-Vision-update-heralds-more-changes>, 17 Mayıs 2018.
- [19] Madden-Julian Conversation, (2011), **A visit to AMF2 site**, <http://maddenjulianconversation.blogspot.com.tr/2011/10/we-are-modelers-in-field-program.html>, 17 Mayıs 2018
- [20] Niemietz R., (2008), **RGB channels separation**, http://commons.wikimedia.org/wiki/File:RGB_channels_separation.png, 17 Mayıs 2018
- [21] OpenCV.org, (2015), **Contours in OpenCV**, https://docs.opencv.org/3.1.0/d3/d05/tutorial_py_table_of_contents_contours.html, 17 Mayıs 2018
- [22] National Technology and Engineering Solutions of Sandia, **Sun Position**, <https://pvpmc.sandia.gov/modeling-steps/1-weather-design-inputs/sun-position>, 20 Mayıs 2018

ÖZGEÇMİŞ

14 Nisan 1990 tarihi, İstanbul ili Bakırköy ilçesi doğumluyum. 2008 yılında Avcılar Teknik Lisesi, Bilgisayar bölümünü, 2011 yılında Süleyman Demirel Üniversitesi, Uluborlu S.K.M.Y.O, Bilgisayar Programcılığı önlisans bölümünü ve 2013 yılında Beykent Üniversitesi, Mühendislik Mimarlık Fakültesi, Bilgisayar Mühendisliği lisans bölümünü tamamladım. 2013 yılından beri, sektörde “Yazılım Mühendisi” unvanıyla çalışmaktayım. 2015 yılında Beykent Üniversitesi, Fen Bilimleri Enstitüsü, Bilgisayar Mühendisliği Anabilim Dalında yüksek lisans eğitimine başladım.

Niteliklerim, uzman seviyede C++/Qt, C#, OpenCV, Image Processing, Socket Programming, ZigBee, SVN, Visual Studio, C++/CLI, orta seviyede Java(Android SDK), PHP, T-SQL, boost, OGRE 3D, OpenGL, Computer Vision, Embedded System Design, Microchip PIC, eBUS, MySQL, Eclipse IDE ve temel seviyede: Cryptography, Machine Learning, Database Design, Doors, Knime şeklindedir.

Yabancı dilim İngilizce olup, bekârım.

Recep ALTIN