T.C.
KASTAMONU ÜNİVERSİTESİ
MÜHENDİSLİK VE MİMARLIK FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

BİTİRME PROJESİ I

KONU

Rubik's Küpü Çözücü Program

HAZIRLAYAN

214410802 – Bashar Alkhawlani

DANIŞMAN

Dr. Öğr. Üyesi Ali Burak ÖNCÜL

Hazıran - 2025
KASTAMONU

ETİK BEYAN

Kastamonu Üniversitesi, Mühendislik ve Mimarlık Fakültesi, Bilgisayar Mühendisliği Bölümü, Mühendislik Tamamlama Programı, Tez Hazırlama Kılavuzu'nda yer alan kurallara uygun olarak hazırladığım bu çalışmada; proje içinde sunduğum verileri, bilgileri ve dokümanları akademik ve etik kurallar çerçevesinde elde ettiğimi, tüm bilgi, belge, değerlendirme ve sonuçları bilimsel etik ve ahlak kurallarına uygun olarak sunduğumu, proje çalışmasında yararlandığım eserlerin tümüne uygun atıfta bulunarak kaynak gösterdiğimi, kullanılan verilerde herhangi bir değişiklik yapmadığımı, bu projede sunduğum çalışmanın özgün olduğunu, bildirir; aksi bir durumda aleyhime doğabilecek tüm hak kayıplarını kabullendiğimi beyan ederim.

Öğrenci Numarası **:** 214410802 **İmza**

Adı Soyadı **:** Bashar Alkhawlani

ÖZET

Bitirme Projesi

Rubik's Küpü Çözücü Program

Bashar Alkhawlani

Kastamonu Üniversitesi
Bilgisayar Mühendisliği Bölümü

Bitirme Projesi Danışmanı:
Dr. Öğr. Üyesi Ali Burak ÖNCÜL
Hazıran 2025

Rubik's Küpü'nün çözümünü kolaylaştırmak amacıyla geliştirilen bu yazılım, bilgisayara bağlı bir kamera yardımıyla küpün yüzlerini tanıyıp, üzerindeki renkleri analiz ederek doğru çözüm yolunu üretmeyi amaçlamaktadır. Kamera aracılığıyla her yüzün görüntüsü alınacak, OpenCV kütüphanesi kullanılarak her karenin rengi tespit edilecektir. Daha sonra elde edilen renk dizilimi, Kociemba algoritması kullanılarak çözüm adımlarına dönüştürülecektir. Yazılım, kullanıcı dostu bir arayüzle birlikte çalışmakta ve çözüm sürecini görsel olarak sunmaktadır. Bu yazılımın geliştirilme sürecinde kullanılan yöntemler, teknikler ve araçlar detaylı şekilde açıklanmıştır. Python programlama dili, OpenCV kütüphanesi, Kociemba algoritması, renk analizi teknikleri ve arayüz tasarımı konularında çeşitli araştırmalar yapılmıştır.

Anahtar Sözcükler : Rubik's Küpü, Görüntü İşleme, Python, OpenCV, Kociemba Algoritması, Renk Tanıma, Kamera, Arayüz Tasarımı

ABSTRACT

Senior Project

Rubik's Küpü Çözücü Program

Bashar Alkhawlani

Kastamonu University
Faculty of Engineering and Architecture
Department of Computer Engineering

Project Advisor:
Dr. Öğr. Üyesi Ali Burak ÖNCÜL
June 2025

In order to facilitate the solution of the Rubik’s Cube, this software was developed to recognize the cube's faces through a camera, analyze the colors of each square, and generate the correct solution path. Using a webcam, images of all six faces of the cube are captured. Then, with the help of the OpenCV library, the color of each square is detected. The resulting color configuration is processed using the **Kociemba algorithm** to compute the optimal sequence of moves to solve the cube. The software runs through a user-friendly interface and visually presents the solution steps. This report explains in detail the techniques, methods, and tools used in the development of this application. Research was conducted in areas such as the Python programming language, OpenCV library, the Kociemba algorithm, color recognition methods, and GUI design.

Key Words : Rubik’s Cube, Image Processing, Python, OpenCV, Kociemba Algorithm, Color Detection, Camera, GUI Desi

İÇİNDEKİLER **SAYFA**

1. GİRİŞ ve TANITIM

Görüntü işleme ve yapay zeka alanlarındaki gelişmeler sayesinde, gerçek dünyadan alınan görsel verilerin analiz edilerek çeşitli problemlerin çözülebilmesi mümkün hale gelmiştir. Bu çalışmada, görsel veriler yardımıyla Rubik's Küpü'nün (Rubik Küp) otomatik olarak çözülmesi amaçlanmaktadır. Günümüzde Rubik Küp, hem eğlenceli bir zeka oyunu hem de algoritma geliştirme ve yapay zeka uygulamaları için örnek bir problem olarak kabul edilmektedir.

Projenin temel amacı, bilgisayara bağlı bir kamera aracılığıyla Rubik Küp'ün her bir yüzünün görüntüsünü almak, bu görüntülerden küpün renklerini analiz etmek ve elde edilen renk dizilimini kullanarak çözüm adımlarını otomatik olarak üretmektir. Bu amaçla görüntü işleme kütüphanelerinden biri olan OpenCV kullanılarak renk tanıma işlemi gerçekleştirilmiştir. Renk bilgileri elde edildikten sonra, dünyaca bilinen ve yaygın olarak kullanılan Kociemba algoritması kullanılarak küpün en kısa çözüm yolu hesaplanmaktadır.

Bu yazılım, sadece renk tanıma ve çözüm üretmekle kalmayıp, aynı zamanda kullanıcı dostu bir arayüzle çözüm adımlarını da görsel olarak sunmaktadır. Kullanıcı, küpün yüzlerini kameraya gösterdikten sonra sistem otomatik olarak renkleri algılar, gerekli analizleri yapar ve adım adım çözüm yolunu ekranda gösterir.

Bu proje, görüntü işleme, algoritma geliştirme, kullanıcı arayüzü tasarımı ve Python programlama dili konularında bilgi ve beceri kazanımını hedeflemekte olup, eğitimde ve uygulamalı yapay zeka projelerinde faydalı bir örnek teşkil etmektedir.

1.1. Sitemin Çalışma Mantığı

Bu projede, Rubik Küp'ün çözümünü otomatikleştirmek amacıyla görüntü işleme ve algoritma tekniklerinden yararlanılmıştır. Sistem, kullanıcının küpün her bir yüzünü kameraya göstermesiyle çalışmaya başlar. Kamera aracılığıyla alınan her görüntü, OpenCV kütüphanesi yardımıyla analiz edilerek ilgili yüzeydeki her bir karenin rengi algılanır ve sistem belleğine kaydedilir.

Renk tanıma işlemi sırasında, hem BGR renk formatı hem de HSV (Hue, Saturation, Value) renk uzayı da kullanılmıştır. HSV kullanımı, özellikle ışık değişimlerinden kaynaklı hataları azaltarak, kırmızı, yeşil, mavi, turuncu gibi küp renklerinin daha sağlıklı bir şekilde tanımlanmasına olanak tanır. Her karenin ortalama HSV değeri alınarak, önceden belirlenen eşik değerlerle karşılaştırılır ve uygun renk sınıfına atanır.

Küp altı yüzden oluştuğundan, kullanıcı tüm yüzleri tek tek sisteme tanıttıktan sonra yazılım, bu renk bilgilerini belirli bir formatta birleştirerek küpün tam konfigürasyonunu elde eder. Bu noktadan sonra sistem, küpün mevcut durumunu çözmek için Kociemba algoritmasını kullanır. Bu algoritma, minimum hamle sayısıyla çözüm üretme konusunda oldukça etkilidir ve sistemimiz bu algoritma sayesinde kullanıcıya en kısa çözüm yolunu adım adım sunar.

Ek olarak, kullanıcıya esneklik sağlamak amacıyla manuel (el ile) giriş seçeneği de sunulmuştur. Eğer kullanıcı kamera kullanmak istemezse veya görüntü alma sırasında bir sorun yaşanırsa, her bir yüzün renklerini uygulama üzerinden elle seçerek girebilir. Bu özellik, özellikle eğitim veya test amaçlı kullanımda büyük kolaylık sağlamaktadır.

Kullanıcı, tüm yüzleri başarıyla tanıttıktan sonra “Cube Solve ” butonuna bastığında, program küpün mevcut dizilimini analiz eder, çözüm adımlarını üretir ve kullanıcıya görsel olarak sunar. Ayrıca 3 boyutlu küp modeli üzerinden çözüm süreci adım adım takip edilebilir. Bu süreçlerin teknik detaylarına 3. bölümde ayrıntılı olarak yer verilmiştir.

2. PROJE KONUSU HAKKINDA GENEL BİLGİLER

Geliştirilen sistem, Rubik küpün çözümünü sağlamak amacıyla kameradan veya manuel olarak girilen görsel verileri işler ve bu verilerden elde edilen renk bilgilerine dayanarak küpün mevcut durumunu analiz eder. Sistem, herhangi bir model eğitimi veya makine öğrenmesi kullanmadan, doğrudan görüntü işleme teknikleri ve Kociemba algoritması yardımıyla küp çözümünü üretmektedir.

2.1 Renk Algılama

Sistemin ilk aşaması, Rubik küpün her yüzündeki 3x3 kare şeklindeki tüm renkli hücrelerin doğru bir şekilde tespit edilmesidir. Bu işlem aşağıdaki iki yöntemden biriyle gerçekleştirilir:

1. **Kamera ile Otomatik Giriş:**
   Kullanıcı, küpün her bir yüzünü kameraya gösterir. Sistem OpenCV kullanarak karedeki yüzü tespit eder ve her karenin ortasından renk örneği alır. Renk tanımlama işlemi **HSV (Hue, Saturation, Value)** renk uzayı kullanılarak yapılır. HSV kullanımı, ışık değişimlerinden kaynaklı renk sapmalarını minimize eder.
2. **Manuel Giriş:**
   Kullanıcı, küpün her yüzünü sistemdeki arayüz üzerinden elle girerek tanımlar. Her bir karenin rengi kullanıcı tarafından seçilir. Bu yöntem, kameranın yetersiz olduğu durumlarda alternatif olarak sunulmaktadır.

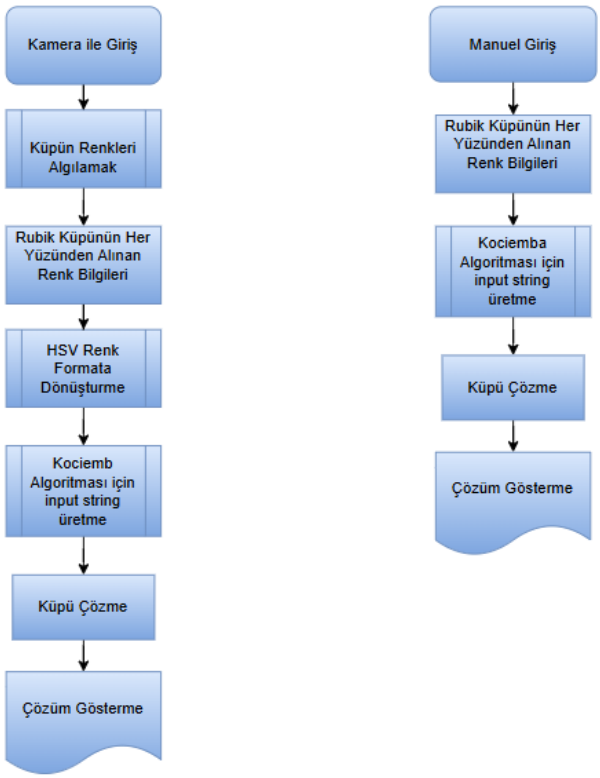
Bu iki giriş yönteminden herhangi biriyle elde edilen 54 renk bilgisi, sistem tarafından önceden tanımlanmış renk etiketlerine (örneğin: beyaz, mavi, kırmızı, yeşil, turuncu, sarı) göre sınıflandırılır ve çözüm algoritmasına uygun biçimde dizilir.

2.2 Çözüm Aşaması

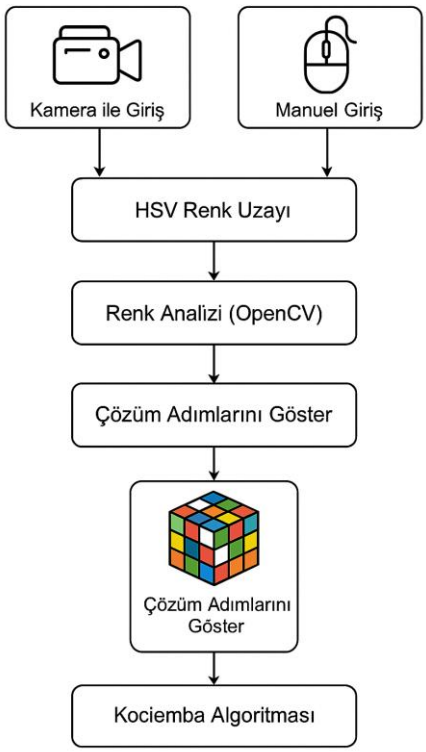
Renk bilgileri alındıktan sonra, sistem bu bilgileri **Kociemba algoritmasına uygun olacak biçimde düzenler**. Ardından, **Kociemba algoritması** kullanılarak Rubik küpün en kısa sürede çözülebilecek adımları hesaplanır. Elde edilen çözüm, kullanıcıya **adım adım metinsel olarak** sunulur. Ayrıca, sistemde yer alan **3 boyutlu küp modeli** yardımıyla kullanıcı mevcut durumu ve çözüm adımlarını **görsel olarak** takip edebilir.

Kullanım kolaylığını artırmak amacıyla, **her bir adım için yön ve dönüş bilgisini gösteren açıklayıcı görseller** de kullanıcıya sunulmaktadır. Bu sayede, kullanıcı algoritmanın önerdiği hamleleri daha rahat anlayabilir ve uygulayabilir.

2.3 Sistemin Akış Diagramlar



Şekil 2.1 Sistemin Akış Diyagrami 1



Şekil 2.3 Sistemin Akış Diyagrami 2

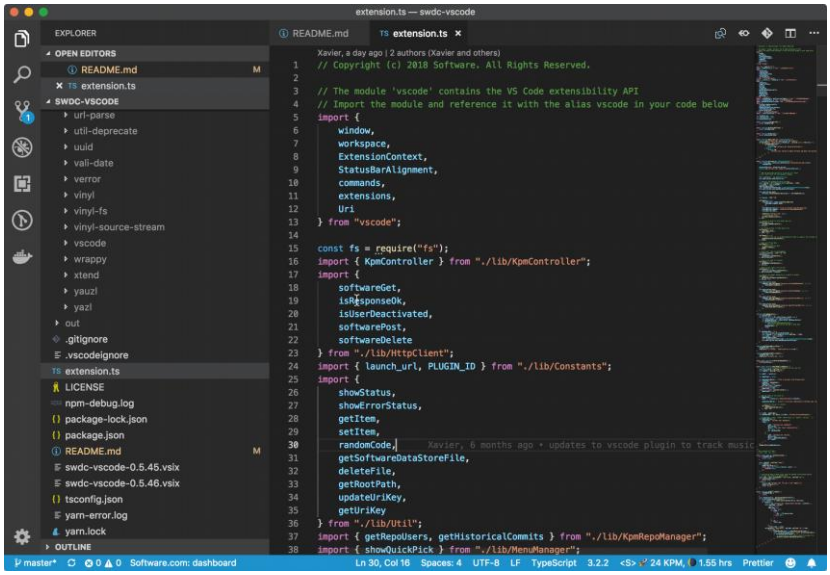
3. PROJE ÇALIŞMASINDA KULLANILAN MATERYALLER.

3.1 Visual Studio Code

Günümüzde programlama dilleri bağımsız bir alana taşınmaya başlanmıştır. Bir programlama dili ile geliştirme yapmak için her türlü işletim sistemi kullanılabilir olmalıdır. Son bir kaç yılda Microsoft bu değişime ayak uydurarak açık kaynak ürünler ve platform bağımsız teknolojiler üretmeye başlamıştır. Microsoft .net ile artık Mac OS ve Linux platformlarında geliştirme yapmak mümkün hale gelmiştir.

Visual Studio Code hızlı, hafif olmanın yanında Microsoft, Linux ve Mac işletim sistemlerinde çalışabilen bir araçtır. Kurulum yapıldığında basit bir metin editörü görüntüsü ile karşımıza çıkan ürün, eklentiler sayesinde Node JS, Ruby, Python, C/C++, C#, Javascript gibi bir çok programlama dilini desteklemektedir. Yani klasik Visual Studio ürünleri gibi her şeyi bünyesinde barındırmak yerine çekirdek bir yapı ile başlatılıp lazım olan parçaların eklentiler halinde ürüne eklenmesi sağlanarak mantıklı bir hareket yapılmıştır. Eklentilerin indirilebildiği bir market ortamı oluşturulmuştur.

Visual Studio Code, hata ayıklama özellikleriyle, gelişmiş web ve bulut uygulamaları üstünde kodları düzenlemeye, yeniden tanımlamaya ve optimize etmeye yarar. Visual Studio Code tamamen ücretsiz olup, dilediğiniz gibi kullanabilir, kodlarını inceleyebilir ve kendi ihtiyaçlarınıza göre değişim yapabilirsiniz. Uygulama, çoklu platform esteğine sahip olduğu için Linux, Mac OS X ve Windows üzerinde çalışır ve programcılar için yaklaşık 30 programlama dili desteği sunmaktadır



3.2 Python Programlama Dili

Bu projede kullanılan en temel yazılım araçlarından biri Python programlama dilidir. Python'un açık kaynaklı, esnek ve geniş kütüphane desteğine sahip yapısı, görüntü işleme, renk analizi ve algoritma uygulamaları gibi birçok karmaşık işlemi kolaylaştırmıştır. Özellikle `OpenCV`, `NumPy` ve `kociemba` gibi güçlü kütüphanelerin Python ortamında entegre bir şekilde çalışabilmesi, projenin geliştirilme sürecini hem hızlandırmış hem de daha verimli hale getirmiştir.

Python'un sadeliği sayesinde görüntülerin yakalanması, renk değerlerinin analiz edilmesi, verilerin uygun yapıya dönüştürülmesi ve çözüm algoritmasının çalıştırılması gibi işlemler az kod ile gerçekleştirilebilmiştir. Ayrıca hata ayıklama ve test süreçlerinde Python'un sunduğu interaktif çalışma ortamı, karşılaşılan problemleri daha kısa sürede tespit edip çözmeme yardımcı olmuştur.

Python'un modüler yapısı, ileride projeyi daha da genişletmek ve örneğin bir robot kol ile fiziksel olarak küpü çözebilecek bir sistem entegre etmek gibi ileri seviye uygulamalar için uygun bir temel sunmaktadır. Bu yönüyle Python, projemin hem mevcut hem de gelecekteki aşamaları için vazgeçilmez bir araç olmuştur.

3.3 OpenCV Kütüphanesi

OpenCV (Open Source Computer Vision Library), bilgisayarla görme ve görüntü işleme alanında yaygın olarak kullanılan açık kaynak kodlu bir kütüphanedir. 1999 yılında Intel tarafından geliştirilmeye başlanmış, sonrasında Itseez, Willow Garage, Nvidia, AMD ve Google gibi teknoloji devlerinin katkılarıyla gelişmeye devam etmiştir. İlk sürüm olan OpenCV Alpha, 2000 yılında yayınlanmıştır. OpenCV, BSD lisansı ile lisanslandığı için tamamen ücretsizdir ve hem akademik hem de ticari projelerde özgürce kullanılabilir.

Bu kütüphane, görüntü işleme (image processing) ve makine öğrenmesi (machine learning) alanlarında 2500'den fazla güçlü algoritma içermektedir. Bu algoritmalar sayesinde yüz tanıma, nesne algılama, hareket takibi, optik karakter tanıma (OCR), plaka tanıma, üç boyutlu görüntü işleme gibi pek çok işlem rahatlıkla yapılabilmektedir.

OpenCV; C, C++, Python, Java gibi popüler dillerle kullanılabilir ve Windows, Linux, MacOS, Android ve iOS gibi birçok platformda çalışabilir. Bu geniş destek sayesinde araştırma projelerinden ticari uygulamalara kadar birçok alanda yaygın olarak tercih edilmektedir.

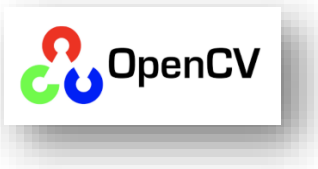
Bu projede, Rubik küpün çözümünü otomatikleştirmek amacıyla kullanıcıdan alınan görüntüler üzerinden yüzey renklerini doğru bir şekilde tespit etmek için OpenCV kütüphanesinden yoğun bir şekilde yararlanılmıştır. Sistem, kameradan alınan görüntüyü analiz ederek her yüzeydeki karelerin konumlarını ve renklerini belirler. Bu süreçte, görüntülerin HSV (Hue, Saturation, Value) renk uzayına dönüştürülmesiyle renk tanıma işlemleri daha doğru ve kararlı hale getirilmiştir. HSV formatı, renkleri insan gözüne daha yakın bir şekilde temsil ettiği için özellikle renk tespiti gereken uygulamalarda BGR formatına göre daha başarılı sonuçlar vermektedir.

Kullanıcı, dilerse küp yüzeylerinin renklerini **manuel (el ile)** girebilir. Bu özellik, kameranın algılamada zorlandığı durumlarda sistemi kullanmaya devam etmeyi mümkün kılarak kullanıcı deneyimini artırmaktadır.

Ayrıca tespit edilen yüzey renk bilgileri, **Kociemba algoritması** için uygun formata çevrilerek en kısa çözüm adımları hesaplanmaktadır. OpenCV bu aşamada yalnızca görüntüden renk tespiti yapmakla kalmaz; aynı zamanda yüzün 48x48 piksele

küçültülmesi, kontur tespiti, dikdörtgen içine alma ve görselleştirme gibi birçok ara işlemde de kritik bir rol oynar.

Sonuç olarak OpenCV, bu projenin temel yapı taşlarından biridir. Hem kamera görüntülerinin işlenmesi hem de kullanıcıya görsel olarak anlaşılır bir çıktı sunulması açısından oldukça etkili bir araç olmuştur. Projenin başarısında bu güçlü kütüphanenin katkısı büyüktür.



Şekil 3.1. OpenCV Sembolu

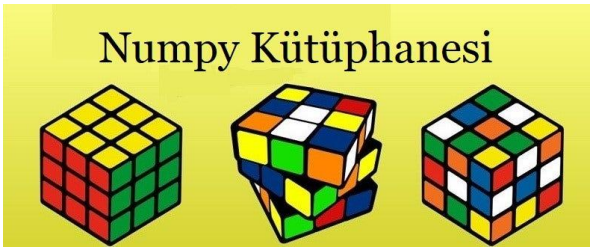
3.4 NumPy Kütüphanesi

NumPy (Numerical Python), Python dilinde bilimsel ve sayısal hesaplamalar için kullanılan temel bir kütüphanedir. Çok boyutlu diziler (array) üzerinde yüksek performanslı matematiksel işlemler yapılmasına olanak tanır. OpenCV ve diğer görüntü işleme kütüphaneleri ile birlikte çalışarak görüntülerin piksel verilerini hızlı bir şekilde işlemek için kullanılır.

Bu projede NumPy'nin kullanım alanı, HSV (Hue, Saturation, Value) renk uzayına dönüştürülen Rubik küp yüzey görüntülerindeki renk analizidir. Her bir yüzün 3x3 renk hücresinden elde edilen HSV değerleri NumPy dizileri içerisine alınır ve bu dizilerdeki renk değerlerinin ortalaması alınarak temsil edici (ortalama) bir renk belirlenir.

Bu işlem, ortam ışığı ve küçük renk varyasyonlarına karşı daha stabil ve doğru bir sınıflandırma yapılmasını sağlar. HSV renk uzayı kullanmak, RGB'ye göre renk tonlarını daha etkili bir şekilde ayırt etmeye olanak tanır ve böylece her yüzeye ait doğru rengi tahmin etmek mümkün olur.

NumPy sayesinde bu işlemler hızlı ve verimli bir şekilde gerçekleştirilmektedir. Böylece sistem, her yüzün gerçek rengine daha yakın bir renk tahmini yaparak, Kociemba algoritmasına doğru veri aktarımını sağlamaktadır.



Şekil 3.2

3.5 Pillow (PIL) Kütüphanesi

Pillow, Python programlama dili için geliştirilmiş popüler bir görüntü işleme kütüphanesidir. Orijinal **PIL (Python Imaging Library)** kütüphanesinin geliştirilmiş ve sürdürülen bir versiyonudur. Pillow; görüntüleri açmak, düzenlemek, yeniden boyutlandırmak, üzerine yazı veya şekiller çizmek, farklı formatlara dönüştürmek gibi birçok işlevi destekler.

Bu projede Pillow kütüphanesi, Rubik küpün çözüm adımlarını kullanıcıya **görsel olarak** sunmak amacıyla kullanılmıştır. Sistem, Kociemba algoritmasıyla üretilen çözüm adımlarını sadece metin olarak göstermekle kalmaz, aynı zamanda bu adımların her birine karşılık gelen görselleri de kullanıcıya gösterir. Bu sayede, kullanıcı her adımı kolayca anlayabilir ve fiziksel küp üzerinde uygulayabilir.

Pillow kütüphanesi yardımıyla:

- Adım adım çözüm resimleri ekranda gösterilir.
- Görüntüler Tkinter arayüzüyle uyumlu hale getirilerek kullanıcı arayüzüne entegre edilir.
- Kullanıcının bir sonraki adıma geçmesini kolaylaştıracak şekilde düzenli ve anlaşılır bir görsel akış sağlanır.

Bu görsel destek, kullanıcı deneyimini artırarak sistemin daha sezgisel kullanılmasını sağlar. Ayrıca çözümün sadece teknik değil, **kullanıcı dostu** bir şekilde sunulmasına katkıda bulunur.

3.6 Tkinter Kütüphanesi

Tkinter, Python programlama dili ile grafiksel kullanıcı arayüzleri (GUI – Graphical User Interface) geliştirmek amacıyla kullanılan en temel ve yerleşik kütüphanelerden biridir. Bu kütüphane, Python'un standart kütüphanesi ile birlikte gelmekte olup, kullanıcıların görsel olarak etkileşimli uygulamalar geliştirmelerine olanak sağlar. Tkinter, temel olarak Tcl/Tk adı verilen daha eski ve güçlü bir GUI araç takımına dayanır. İlk olarak 1990'lı yılların başında geliştirilen Tkinter, zamanla Python geliştiricileri arasında yaygınlaşarak, birçok masaüstü uygulamasının temel aracı hâline gelmiştir. Bugün hala sadeliği, taşınabilirliği ve düşük sistem gereksinimi nedeniyle geniş bir kullanıcı kitlesi tarafından tercih edilmektedir.

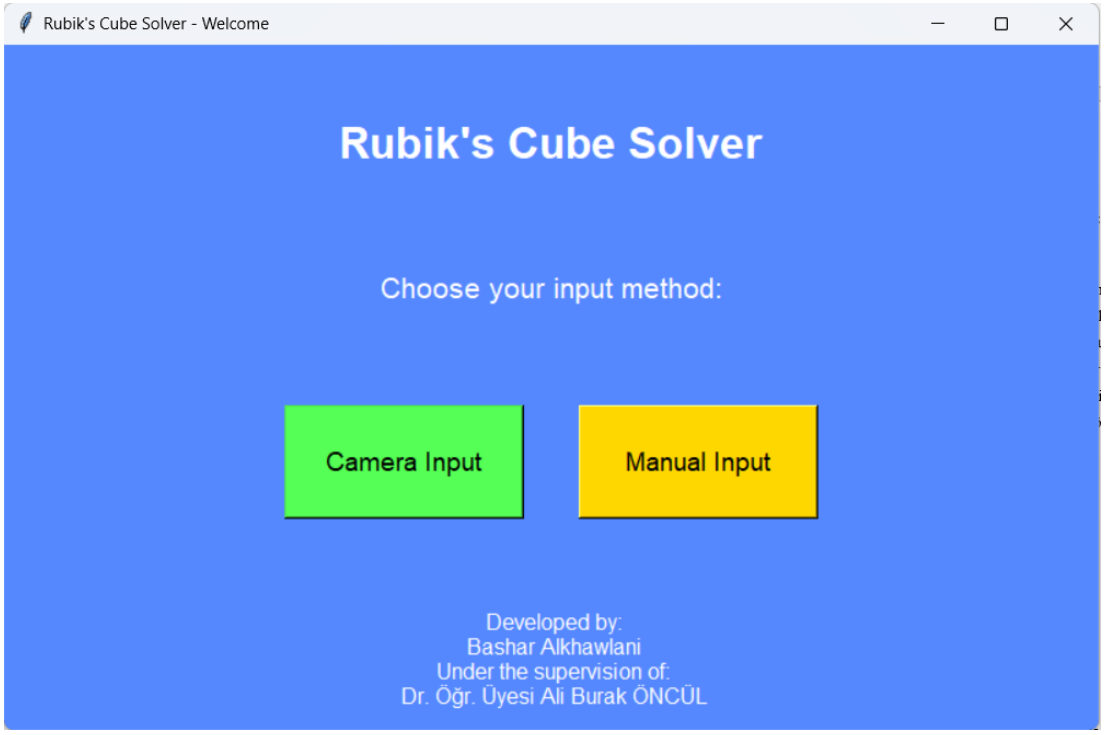
Tkinter, bir uygulama içinde pencereler oluşturmak, bu pencerelere düğmeler, etiketler, giriş kutuları, menüler, tuval alanları gibi öğeler eklemek için kolay ve etkili yollar sunar. Geliştiricilere görsel arayüzler oluştururken Python'un sadeliğini koruyarak hızlı çözümler üretme imkânı tanır. Bu kütüphane, Windows, Linux ve macOS gibi çeşitli işletim sistemleriyle uyumlu çalışabilmekte, herhangi bir harici kütüphane kurulumuna ihtiyaç duymadan doğrudan Python ortamında kullanılabilmektedir.

Bu projede Tkinter kütüphanesi, geliştirilen sistemin görsel arayüzünü oluşturmak için tercih edilmiştir. Kullanıcı ile sistem arasında doğrudan ve anlaşılır bir iletişim sağlamak amacıyla, tüm etkileşimsel yapı bu kütüphane aracılığıyla tasarlanmıştır. Kullanıcının kameradan alınan görüntülerle etkileşime geçmesini sağlamak, küp yüzeylerindeki renkleri kaydetmek, çözüm sürecini başlatmak ve nihai çözüm adımlarını takip etmek gibi işlevlerin tamamı Tkinter üzerinden gerçekleştirilmiştir. Tkinter'ın sunduğu pencere ve bileşen yönetimi sayesinde, sistemdeki işlem adımları kullanıcıya sade ve yönlendirici bir arayüzle sunulmuştur.

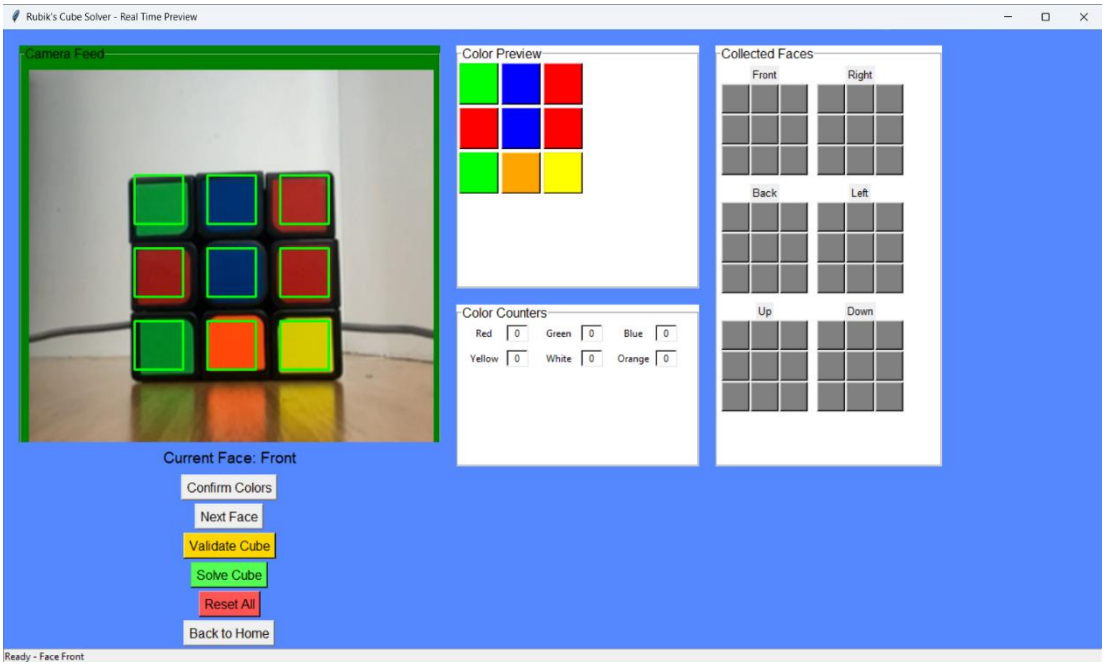
Ayrıca kullanıcı deneyimini daha da kolaylaştırmak amacıyla, çözüm adımlarının yalnızca metin olarak sunulmasıyla yetinilmemiş, aynı zamanda her adımı temsil eden görseller de Tkinter penceresi üzerinde gösterilmiştir. Bu görseller sayesinde kullanıcı, Rubik küpün hangi yönde ve hangi yüzeyinin döndürülmesi gerektiğini görsel olarak kolayca anlayabilmekte ve çözüm sürecini daha rahat bir şekilde takip edebilmektedir. Özetle, Tkinter bu projede yalnızca bir arayüz aracı değil, aynı zamanda kullanıcıyla sistem arasında köprü görevi gören temel bir bileşen olmuştur.

Özetle, Tkinter bu projede yalnızca bir arayüz aracı değil, aynı zamanda kullanıcıyla sistem arasında köprü görevi gören temel bir bileşen olmuştur. Bu kapsamda geliştirilen arayüz tasarımının daha iyi anlaşılması adına, çalışmanın eklerinde arayüzün farklı aşamalarına ait ekran görüntüleri de paylaşılacaktır. Bu görseller, arayüz bileşenlerinin nasıl yerleştirildiğini, kullanıcı etkileşimlerinin nasıl sağlandığını ve çözüm sürecinin görsel olarak nasıl aktarıldığını göstermesi açısından önem arz etmektedir.

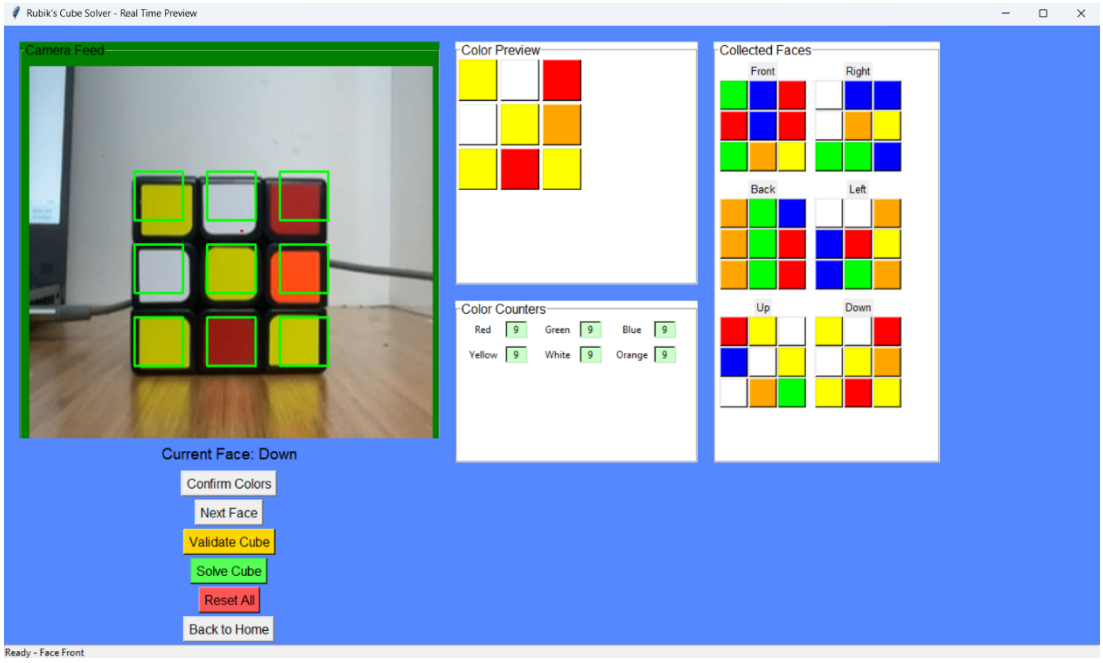
- **Uygulamanın Kullanıcı Arayüzü**



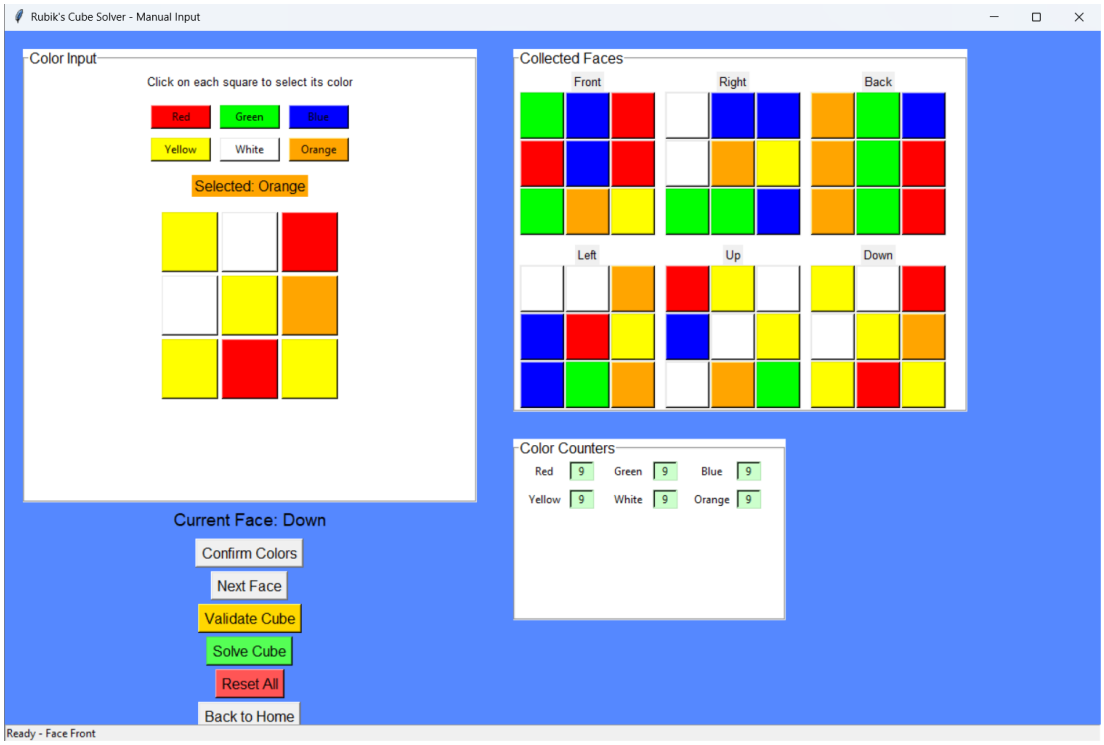
Şekil 3.3 Welcome Sayfası



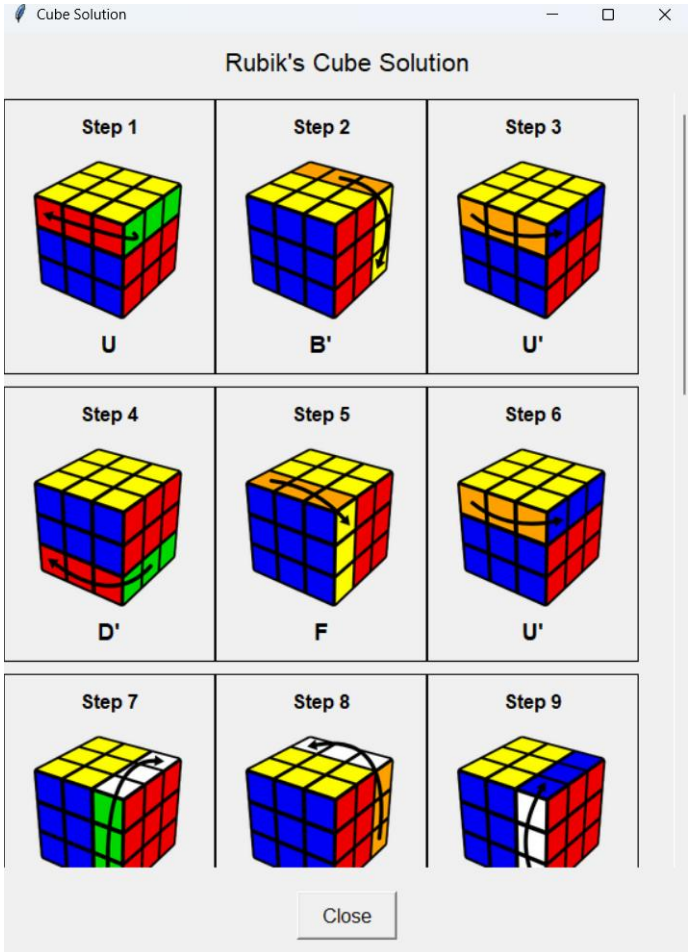
Şekil 3.4



Şekil 3.3 Camera ile giriş Sayfası



Şekil 3.4 Manuel Giriş Sayfası



Şekil 3.5 Çözüm Göstrtim Sayfası

3.7 Rubik Küp Problemi ve Kociemba Algoritması

Rubik Küp, 1974 yılında Macar heykeltıraş ve mimar **Ernő Rubik** tarafından, öğrencilerine üç boyutlu objelerin nasıl hareket ettiğini öğretmek amacıyla icat edilmiştir. Başlangıçta eğitimsel bir araç olarak tasarlansa da, kısa sürede tüm dünyada popüler hale gelmiş ve en çok satılan zeka oyunlarından biri olmuştur. Küp, her biri dönebilen 3x3 boyutundaki 6 yüzeyden oluşur ve her yüzeyde 9 adet küçük kare bulunur. Bu yüzeyler 6 farklı renkten oluşur ve küp karıştırıldığında amaç, her yüzeyi tekrar tek renkli hale getirmektir.

Rubik Küp'ün çözümü yüzeyde basit bir oyun gibi görünse de, arkasında oldukça karmaşık bir kombinatorik yapı barındırır. Standart bir 3x3x3 Rubik Küp'ün **43.252.003.274.489.856.000** farklı konfigürasyonu vardır. Yani bu kadar olasılık içerisinden doğru adımları bularak başlangıç durumuna ulaşmak, sıradan bir kullanıcı için son derece zordur. Bu nedenle, Rubik Küp'ün en kısa sürede ve en az hamlede çözülmesi için birçok algoritma geliştirilmiştir.

Bu algoritmalar arasında en yaygın ve etkili olanlardan biri, **Herbert Kociemba** tarafından 1992 yılında geliştirilen **Kociemba Algoritması**dır. Bu algoritma, küpü iki aşamada çözmeye yönelik akıllı bir stratejiye dayanır. İlk aşamada, küp belirli bir alt gruba (G1) indirgenir. Bu alt grup, bazı hamlelerin sınırlandığı bir durumdur ve küpün çözümüne yaklaşmayı sağlar. İkinci aşamada ise, bu özel gruptan (G1) küpün tamamen çözülmüş olduğu G2 durumuna geçilir. Bu yapı sayesinde Kociemba algoritması, çözümü ortalama olarak **20 hamle veya daha az** sayıda adımda sağlayabilir.

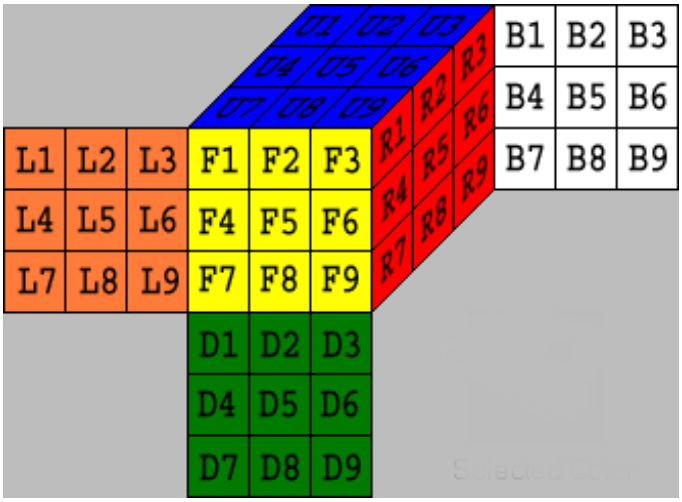
Kociemba algoritması, yalnızca teorik bir çözüm değil, aynı zamanda yazılım projelerinde de pratik olarak kullanılabilir. Python programlama dili için hazırlanmış açık kaynaklı kütüphaneler (örneğin `kociemba` modülü) sayesinde, algoritmanın entegrasyonu oldukça kolaydır. Bu kütüphaneler, kullanıcıdan alınan renk dizilimini çözüm hamlelerine çevirerek çıktılar üretir.

Bu proje kapsamında, kullanıcıdan kamerayla alınan Rubik Küp görüntüleri OpenCV kütüphanesi aracılığıyla işlenir. Her bir yüzeyin renkleri tanımlanır ve elde edilen verilerden küpün tüm durumunu temsil eden bir renk matrisi oluşturulur. Bu matris daha sonra Kociemba algoritmasına giriş olarak verilir. Algoritma, bu karmaşık renk

dizilimini analiz ederek, kullanıcıya küpü çözmek için gereken adımları üretir. Bu adımlar kullanıcıya hem metinsel olarak, hem de projede geliştirilen grafiksel arayüzde görsel olarak sunulur.

Kociemba algoritmasının kullanılmasının başlıca avantajı, kullanıcıdan yalnızca doğru şekilde yüzey renklerini girmesini veya göstermesini istemesidir. Gerisini algoritma otomatik olarak çözerek kullanıcıyı yönlendirir. Bu yaklaşım, sadece oyun amaçlı değil, aynı zamanda bilgisayarla görme (computer vision), algoritmik düşünme, yapay zekâ ve yazılım mühendisliği gibi alanlara da gerçek dünya örneği oluşturur.

Sonuç olarak, Kociemba algoritması bu projenin temel taşlarından biridir. Sunduğu yüksek doğruluk oranı, hızlı çözüm kabiliyeti ve yazılıma kolay entegre edilebilir yapısı sayesinde, Rubik Küp'ün çözümünü sade ve kullanıcı dostu bir hale getirerek projeye büyük katkı sağlamaktadır.



- **Kociemba Algoritmasında Kullanılan Notasyon Sistemi ve Hamleler**

Kociemba algoritması tarafından üretilen çözümler, Rubik Küp'te yapılması gereken dönüşleri standart bir **notasyon sistemi** ile ifade eder. Bu sistem, küpün yüzeylerini ve her bir yüzeye yapılacak dönüşlerin yönünü temsil eden harf ve sembollerle çalışır. Kullanıcının bu notasyonları doğru bir şekilde anlayabilmesi ve uygulayabilmesi, çözümün fiziksel olarak başarıyla gerçekleştirilmesi için kritik öneme sahiptir.

Rubik Küp'ün altı yüzeyi şu harflerle temsil edilir:

- **U** (Up): Üst yüzey
- **D** (Down): Alt yüzey
- **L** (Left): Sol yüzey
- **R** (Right): Sağ yüzey
- **F** (Front): Ön yüzey
- **B** (Back): Arka yüzey

Her bir harf tek başına geldiğinde, saat yönünde 90 derece döndürmeyi ifade eder. Harflerin sonuna ' (kesme işareti) gelirse, bu dönüşün saat yönünün tersine (yani saat yönünün tersi) olduğunu belirtir. Eğer harfin yanında **2** varsa, bu yüzeyin 180 derece döndürüleceğini ifade eder.

Örnek Notasyonlar ve Anlamları:

- **R**: Sağ yüzeyi saat yönünde 90 derece döndür
- **R'**: Sağ yüzeyi saat yönünün tersine 90 derece döndür
- **R2**: Sağ yüzeyi 180 derece döndür
- **U**: Üst yüzeyi saat yönünde 90 derece döndür
- **F'**: Ön yüzeyi saat yönünün tersine 90 derece döndür
- **D2**: Alt yüzeyi 180 derece döndür
- **L'**: Sol yüzeyi saat yönünün tersine döndür
- **B**: Arka yüzeyi saat yönünde döndür

Bu notasyon sistemi, hem uluslararası yarışmalarda (WCA kuralları) hem de algoritmik çözüm yazılımlarında yaygın olarak kullanılır. Kociemba algoritması çıktısı, bu standart notasyonla üretilir ve böylece çözümler, kullanıcı tarafından kolayca takip edilip fiziksel olarak uygulanabilir hale gelir.

Projemde, bu notasyon sisteminin görsel bir arayüz ile desteklenmesi planlanmıştır. Kullanıcı, algoritmanın ürettiği çözüm adımlarını adım adım okuyabilir ya da ekranda animasyonlar aracılığıyla hangi yüzeyi nasıl çevirmesi gerektiğini doğrudan görebilir. Böylece sadece teknik bilgiye sahip kişiler değil, ilk kez Rubik Küp deneyen kullanıcılar bile algoritma çıktısını rahatlıkla anlayabilir ve uygulayabilir

4. SONUÇLAR VE Gelecek Çalışmalar

Bu proje kapsamında, Rubik Küp'ün kamerayla algılanması, yüzey renklerinin dijital olarak tespit edilmesi ve bu renk bilgilerinin çözüm algoritmasına dönüştürülerek küpün adım adım çözülmesi başarıyla gerçekleştirilmiştir. Görüntü işleme için **OpenCV**, matematiksel işlemler için **NumPy** kütüphaneleri kullanılmış; çözüm algoritması olarak ise dünya çapında yaygın şekilde kullanılan **Kociemba algoritması** entegre edilmiştir.

Proje, gerçek dünya problemlerine yönelik bir mühendislik çözümü sunmakta ve kullanıcıya görsel destekli, etkileşimli bir deneyim sağlamaktadır. Uygulama, kullanıcıdan gelen renk verilerini alır, bu verileri Kociemba formatına çevirir ve algoritmanın çıktısı olan çözüm adımlarını kullanıcıya sunar.

Bu çalışmanın sonucunda, görüntü işleme, algoritmik çözümleme ve kullanıcı arayüzü geliştirme alanlarında ileri düzey bilgi ve deneyim kazanılmıştır. Proje, sadece akademik bir uygulama değil, aynı zamanda gelecekteki daha büyük sistemlerin temelini oluşturabilecek bir platformdur.

Gelecek Geliştirme Planları

Projeyi bir üst seviyeye taşımak amacıyla aşağıdaki geliştirmeler planlanmaktadır:

- **Gerçek zamanlı 3D küp görselleştirme** ile kullanıcıya, hangi renklerin tanındığı ve hangi yüzlerin eksik olduğu hakkında daha sezgisel bir rehber sağlanacaktır.
- Renk tanıma algoritması, değişken ışık koşullarında daha kararlı çalışacak şekilde geliştirilecektir (örneğin: HSV renk uzayı, renk düzeltme teknikleri).
- Kullanıcı dostu bir mobil arayüz ile sistem Android/iOS platformlarında erişilebilir hale getirilecektir.
- Farklı dillerde (örneğin: Türkçe, İngilizce, Arapça) çok dilli destek sağlanacaktır.

Bunlara ek olarak, bu projenin doğal bir devamı olarak **mekanik bir robot kol** tasarlanması ve geliştirilmesi hedeflenmektedir. Bu robot, kamera aracılığıyla tanımlanan küp durumunu alacak ve Kociemba algoritmasının verdiği çözüm adımlarını fiziksel olarak uygulayarak küpü gerçek dünyada çözecektir. Böyle bir

sistem; bilgisayarla görme, yapay zekâ, robotik ve mekatronik mühendisliğinin entegrasyonuna güzel bir örnek teşkil edecektir.

Sonuç olarak, bu proje yalnızca bir Rubik Küp çözme aracı değil; aynı zamanda görüntü işleme ve algoritmik çözümleme yöntemlerinin pratikte nasıl uygulanabileceğini gösteren yenilikçi ve geliştirilebilir bir çalışmadır.

5. KAYNAKLAR

1. Kociemba, H. (2012). "Cube Explorer: Optimal Solver for Rubik's Cube." https://kociemba.org/cube.htm
2. Bradski, G. (2000). "The OpenCV Library." Dr. Dobb’s Journal of Software Tools.
3. Oliphant, T. E. (2006). "A guide to NumPy." Trelgol Publishing.
4. Brown, D. J. (2008). "The complete guide to the Rubik's Cube." Carlton Books Ltd.
5. Korf, R. E. (1997). "Finding optimal solutions to Rubik’s Cube using pattern databases." In AAAI/IAAI, Vol. 97, pp. 700-705.
6. İnternet: GitHub, "kociemba/python solver for Rubik’s Cube", https://github.com/muodov/kociemba
7. İnternet: TutorialsPoint, "OpenCV Python Tutorial", https://www.tutorialspoint.com/opencv/index.htm
8. İnternet: Rubik’s Official Website, "How to solve the Rubik’s Cube", https://rubiks.com
9. Zuse, M. (2018). "Color detection using OpenCV in Python." Journal of Computer Vision Applications, 5(2), 34–41.
10. İnternet: PyImageSearch, "How to detect colors using OpenCV", https://pyimagesearch.com/2021/04/28/opencv-color-detection