KARADENİZ TEKNİK ÜNİVERSİTESİ
MÜHENDİSLİK FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

BIL3012 GÖRÜNTÜ İŞLEME DERSİ
DÖNEM PROJESİ

Adı Soyadı
Osman Can AKSOY – 394797

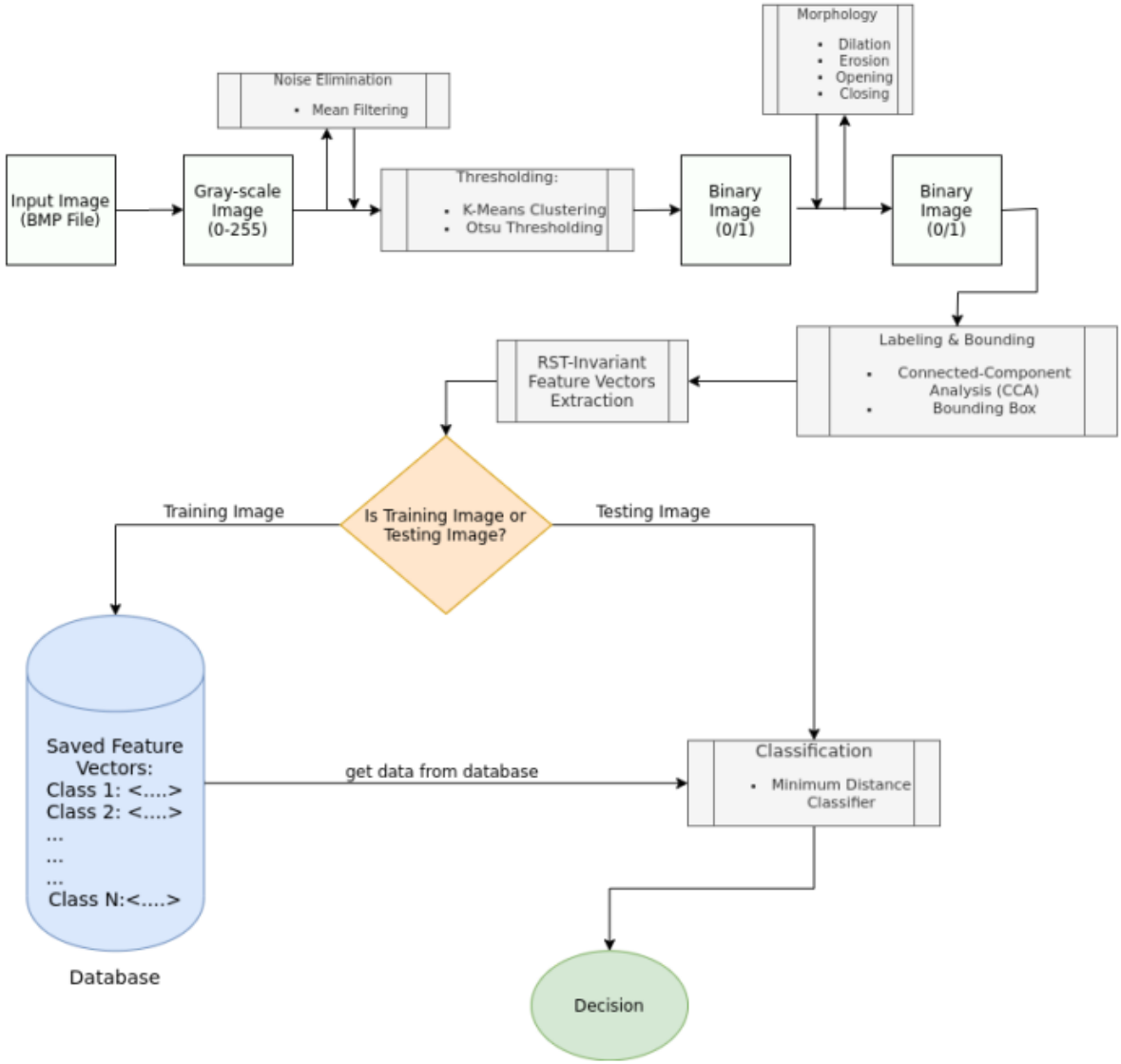
Dersin Sorumlusu
Prof. Murat EKİNCİ

2022-2023 BAHAR DÖNEMİ

Ödev 1:

Geometrik şekilleri farklı nesneleri (pirinç, mercimek, nohut, çekirdek v.b.) algılama ve sınıflandırma.

Aşağıdaki diyagram, tipik bir görüntü işleme sistemini ve ilgili alt süreçleri göstermektedir.



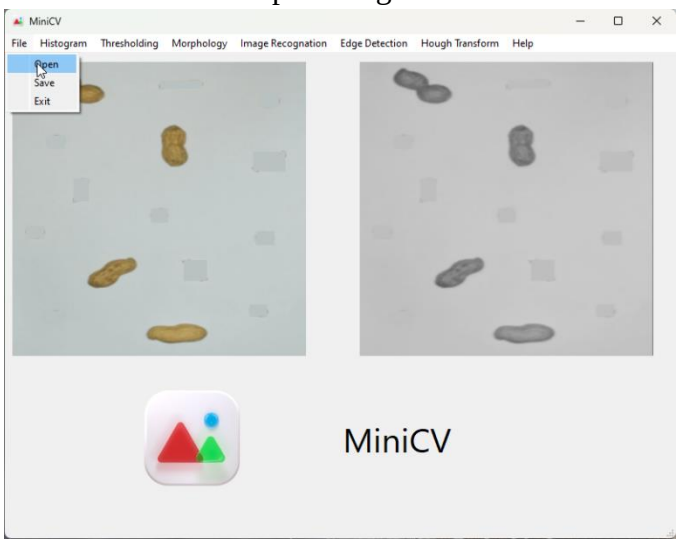
Nesne Algılama ve Sınıflandırma Uygulaması

Yukarıdaki sorunu çözmek için kullanılan Nesne Algılama ve Sınıflandırma uygulaması, Qt Framework kullanılarak C++ ile geliştirilmiştir.

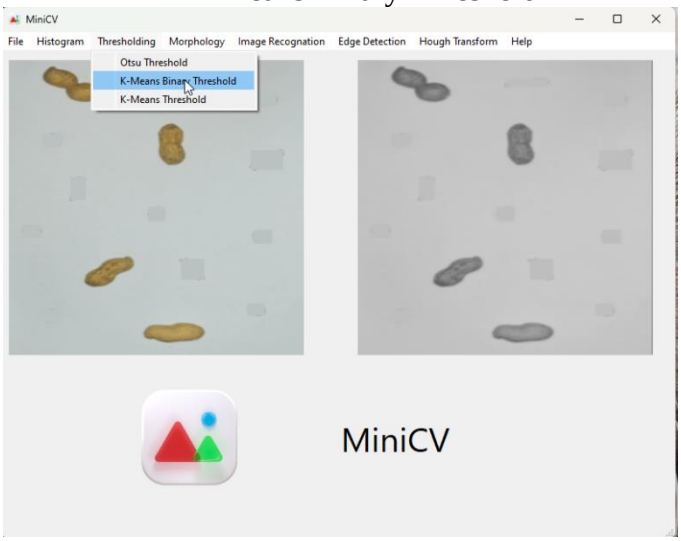
Uygulamanın ekran görüntüleri aşağıda gösterilmiştir:

1) EğitimAşaması

1 – Open Image

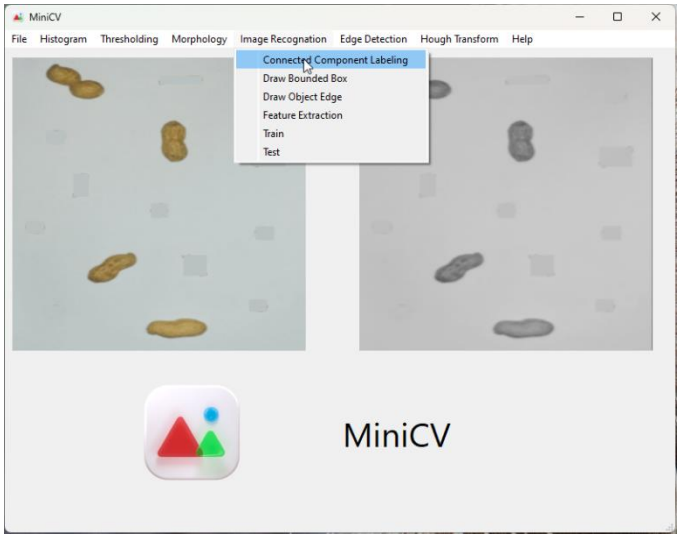


2 – K-Means Binary Threshold



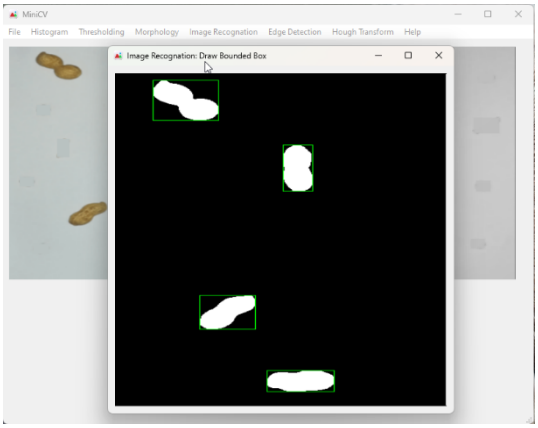
3 – Display Binary Image

4 – Connected Component Labeling

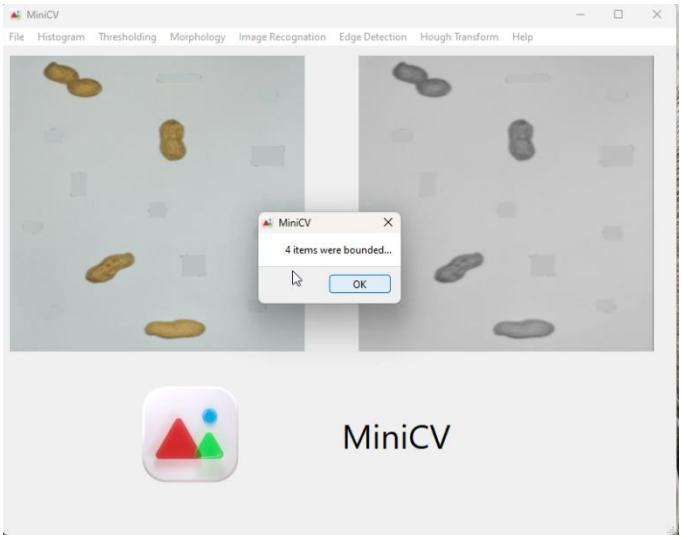


5 – Display Labeling Image

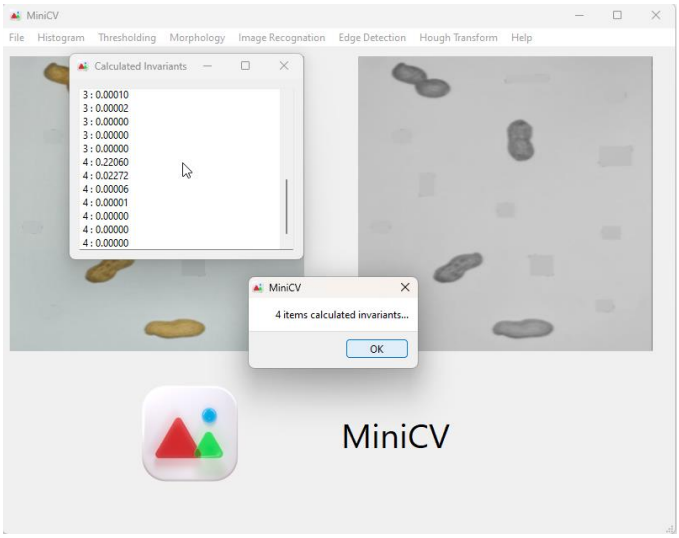
6 – Display Bounded Boxes



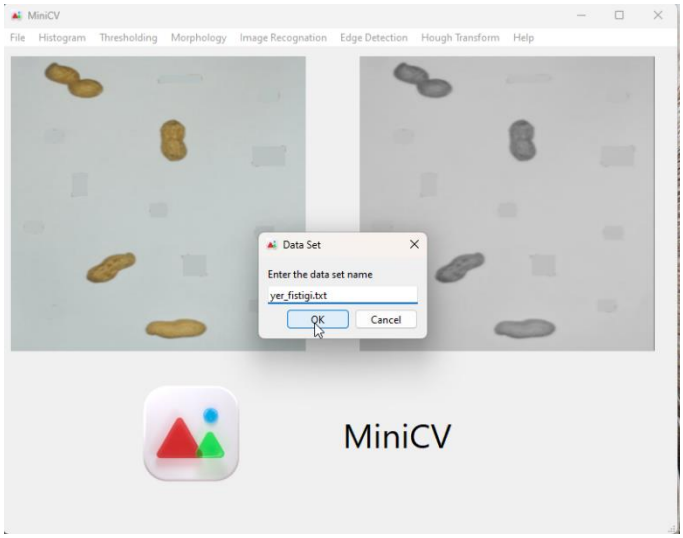
7 – Draw Each Object Boundary Box



8 – Calculate Invariants

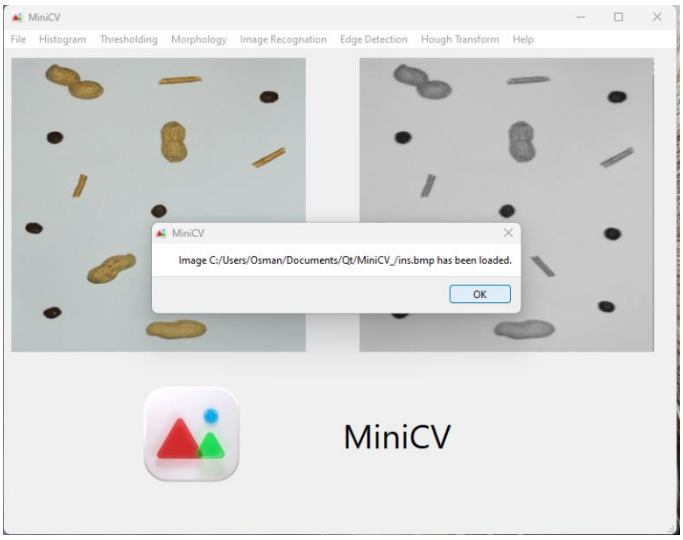


9 – Save Data Set

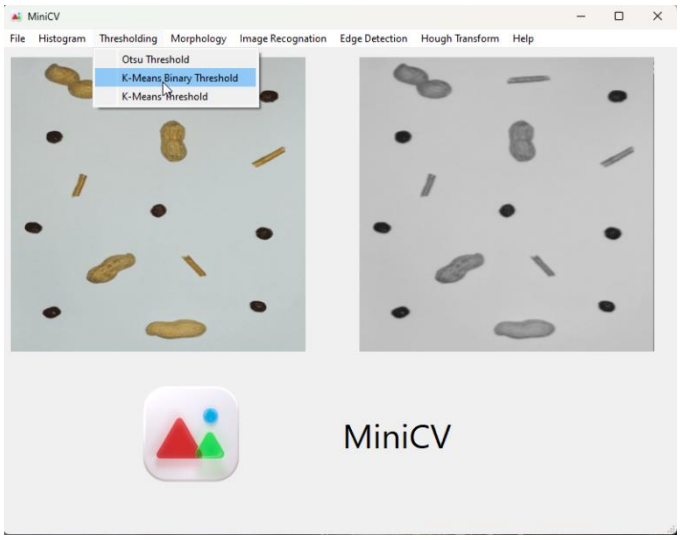


2) Test Aşaması

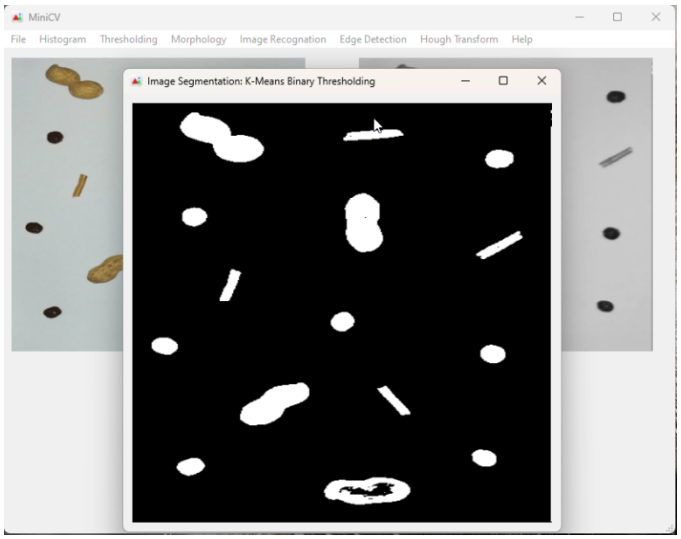
1 – Open Image



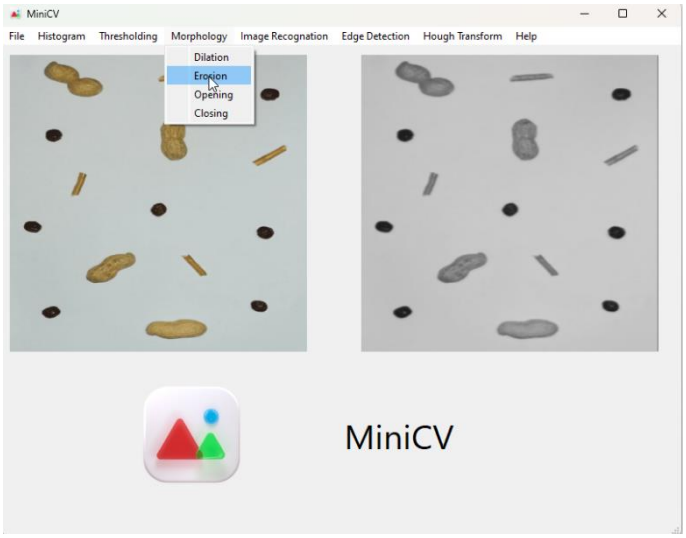
2 – K-Means Binary Threshold



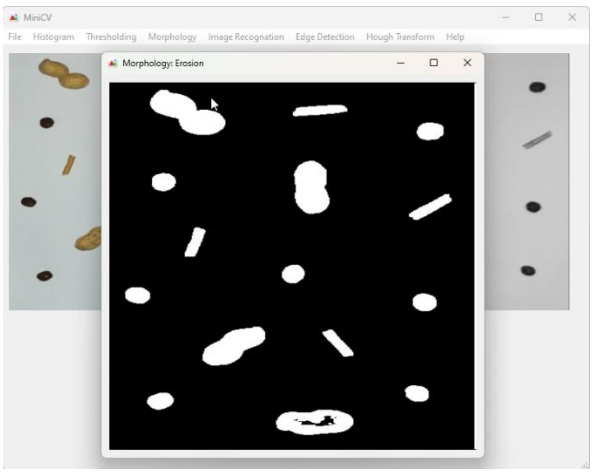
3 – Display Binary Image



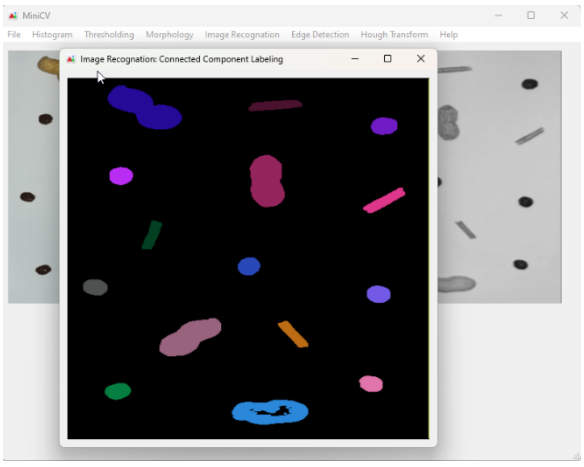
4 – Apply Erosion



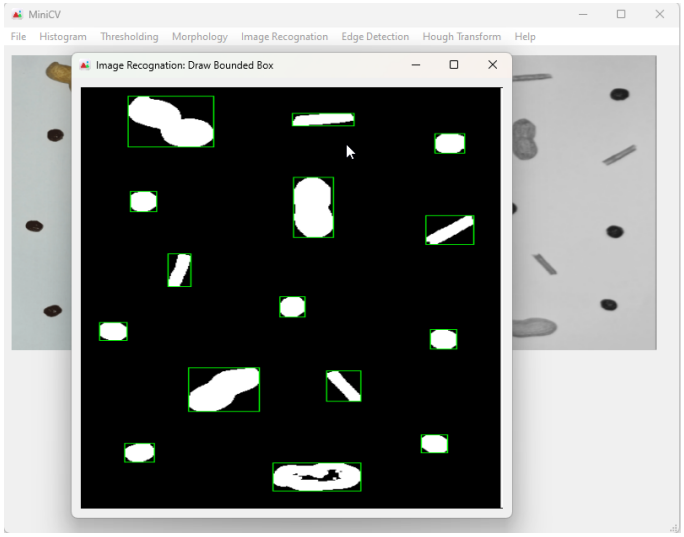
5 – Display Erosion Image



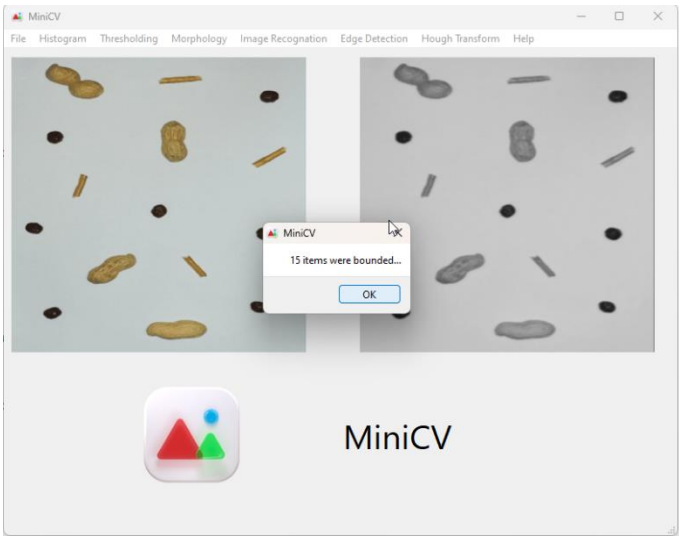
6 – Display Labeling Image



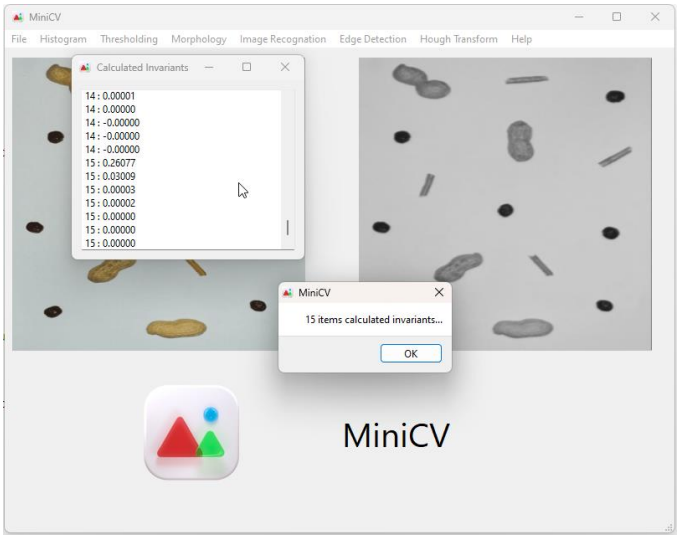
7 – Display Bounded Boxes



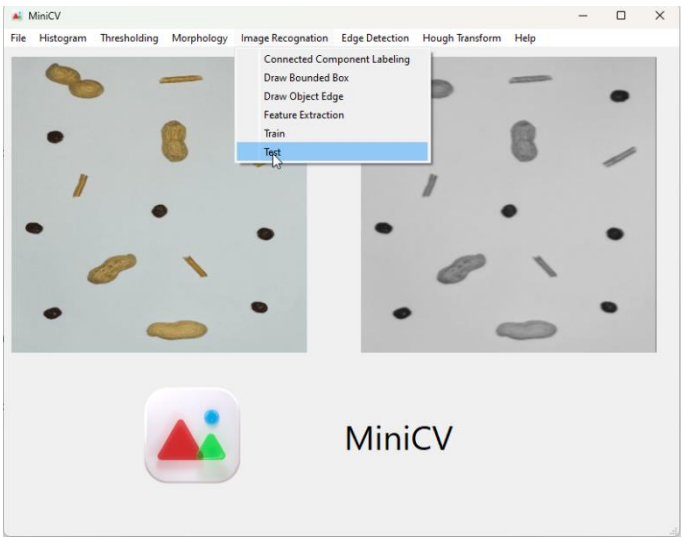
8 - Draw Each Object Boundary Box



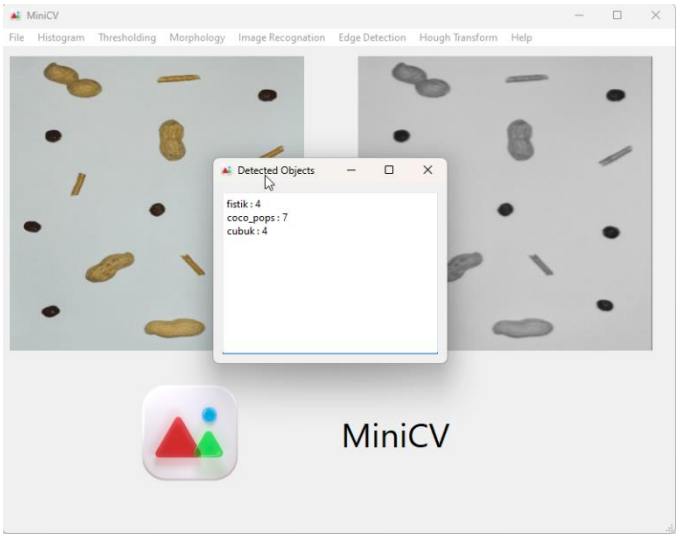
9 – Calculate Invariants



10 – Testing Phase



10 – Display Detected Objects



Uygulamada Kullanılan Sınıflar Ve Temel Fonksiyon Prototipleri:

1) Sınıflar:

- BMPHeader
- BMPReader
- BMPWriter
- Image
- Mat
- Matrix
- Object
- Pixel
- Point
- Shape

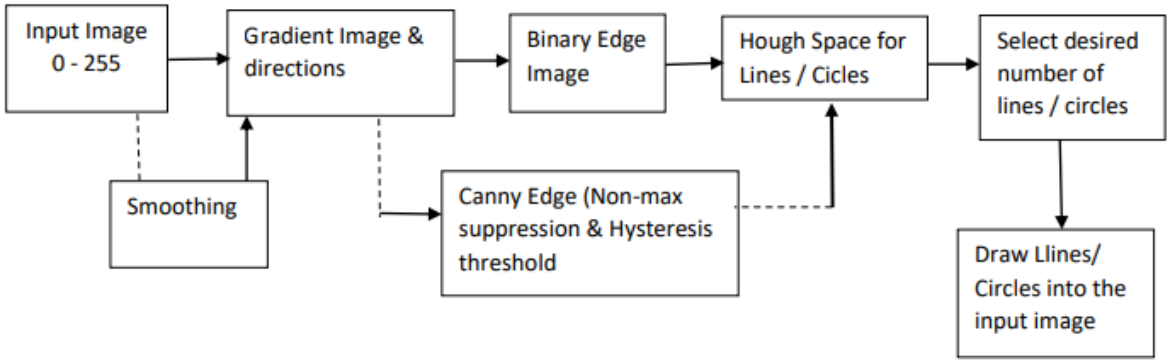
2) Fonksiyon Prototipleri:

- `Mat imread(const string& path);`
- `void imwrite(const string& path, Mat& mat);`
- `void bmp_to_grayscale(Mat& mat);`
- `void k_means_thresholding(Mat& mat);`
- `void erosion(Mat& mat);`
- `void connected_component_labeling(Mat& mat);`
- `void Object::draw_bounding_box();`
- `void Object::cut_object();`
- `void Object::calculate_invariants();`
- `void MainWindow::Train();`
- `void MainWindow::Test();`
- `long double euclidean_distance(Matrix& matrix_one, Matrix& matrix_two);`

Ödev 2:

Görüntüdeki istenen adetteki Doğrusal (Line) ve Dairesel (Circle) yapıdaki sınırların (kenarlar) ve nesnelerin konumlarının bulunması.

Aşağıdaki diyagram, tipik bir görüntü işleme sistemini ve ilgili alt süreçleri göstermektedir.



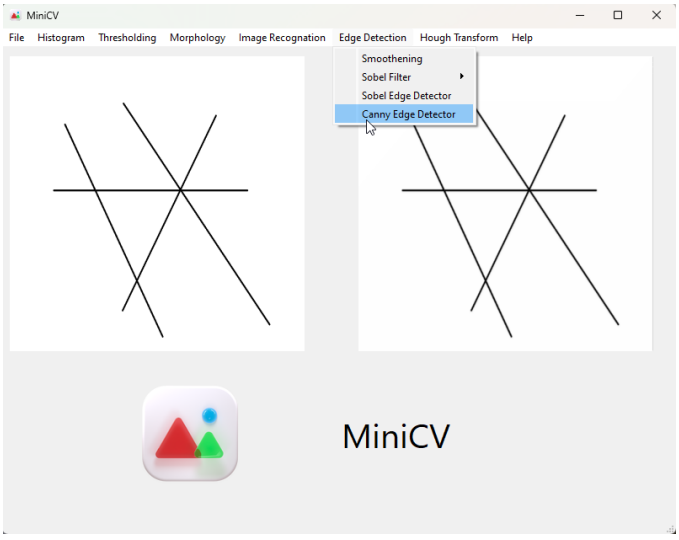
Doğrusal ve Dairesel Yapıların Sınırlarının Bulunması Uygulaması

Yukarıdaki problemi çözmek için kullanılan Doğrusal ve Dairesel Yapıların Sınırlarının Bulunması Uygulaması, Qt Framework kullanılarak C++ ile geliştirilmiştir.

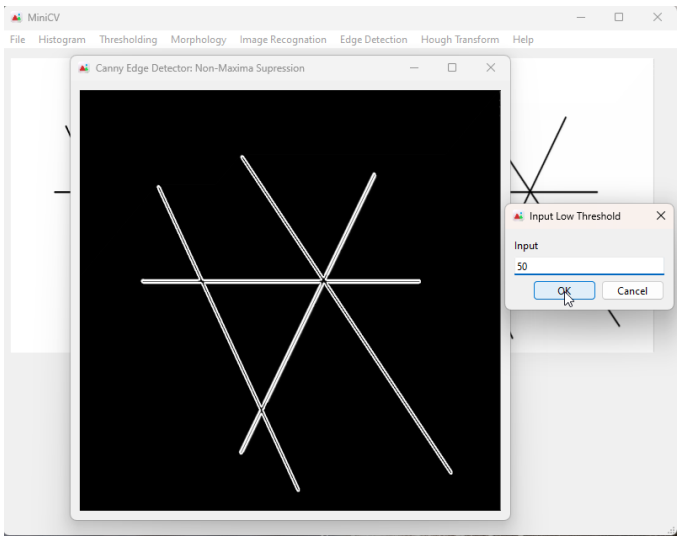
Uygulamanın ekran görüntüleri aşağıda gösterilmiştir:

1) Doğru Algılama

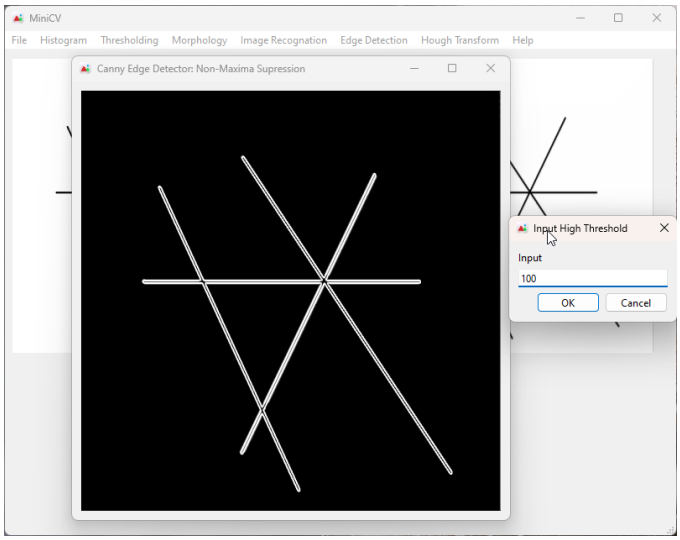
1 – Canny Edge Detector



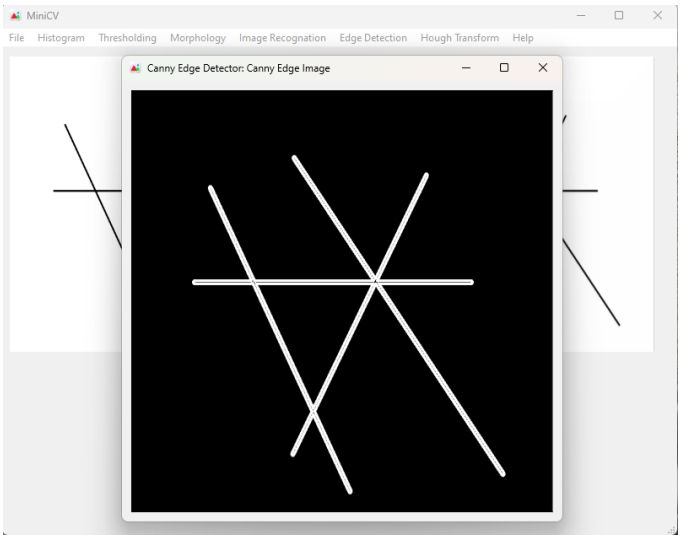
2 – Input Hysterisis Low Threshold



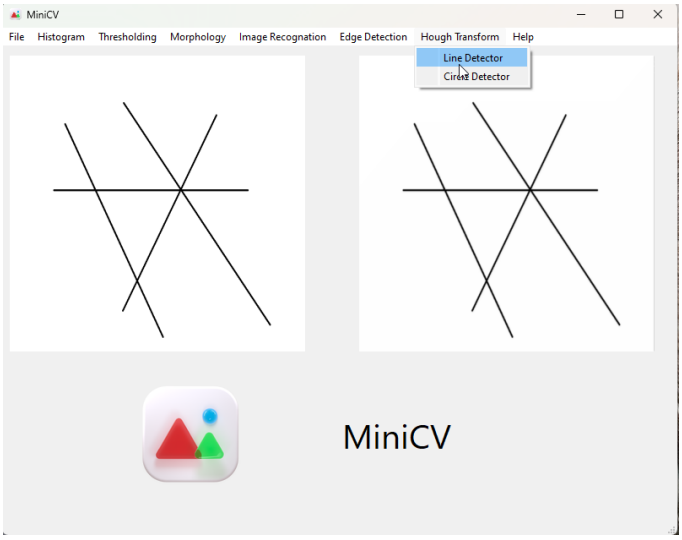
3 – Input Hysterisis High Threshold



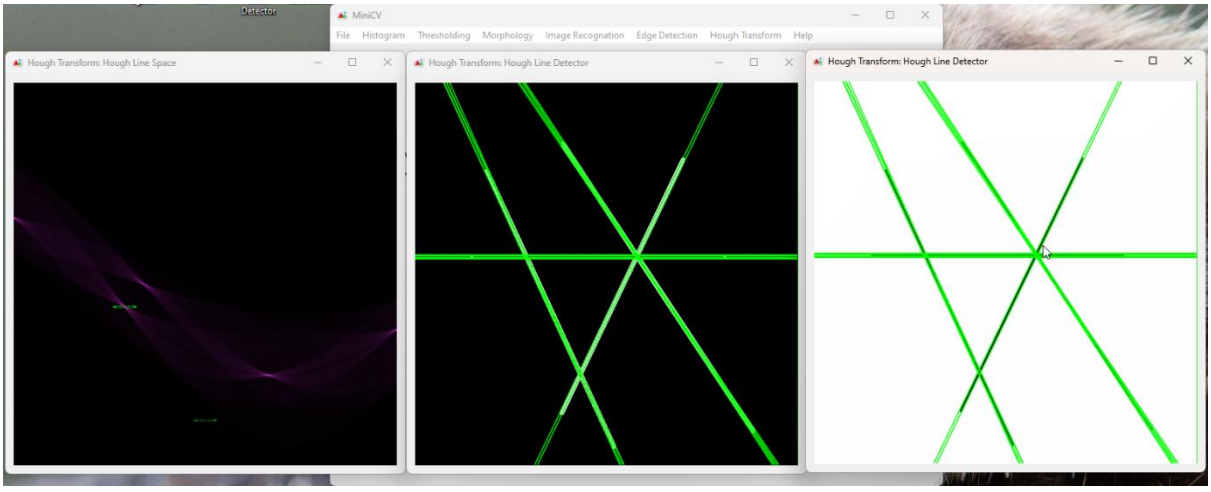
4 – Display Canny Edge Detector



5 – Apply Hough Transform(Line Detector)

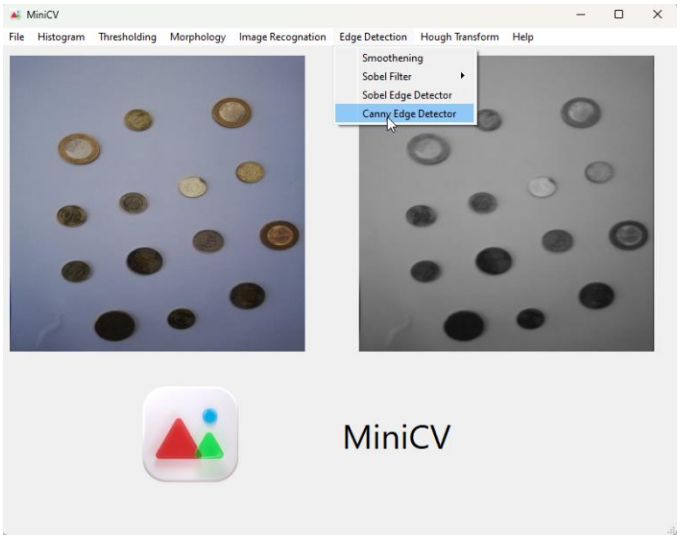


6 – Display Hough Space & Detected Lines(Binary and Gray Level)

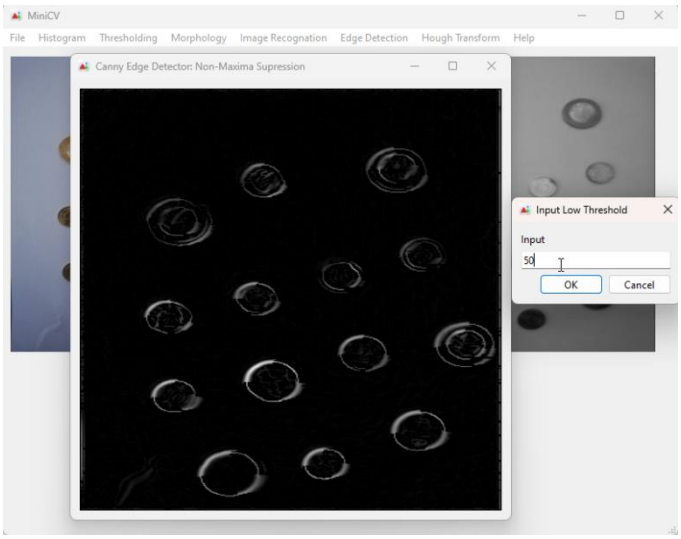


2) Daire Algılama

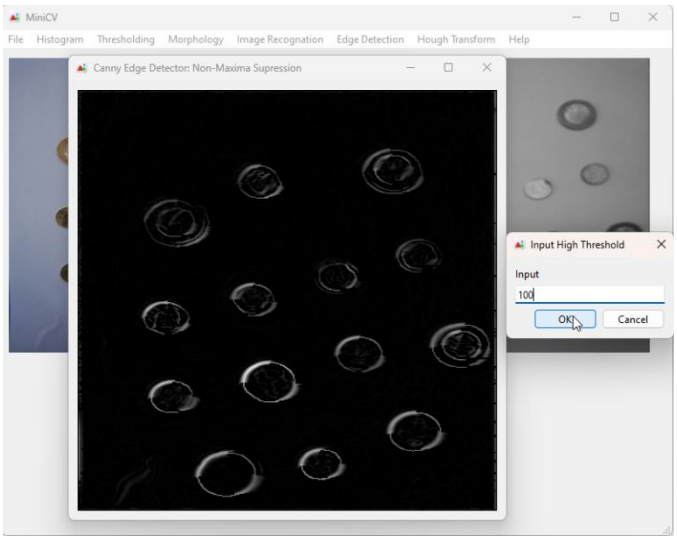
1 – Canny Edge Detector



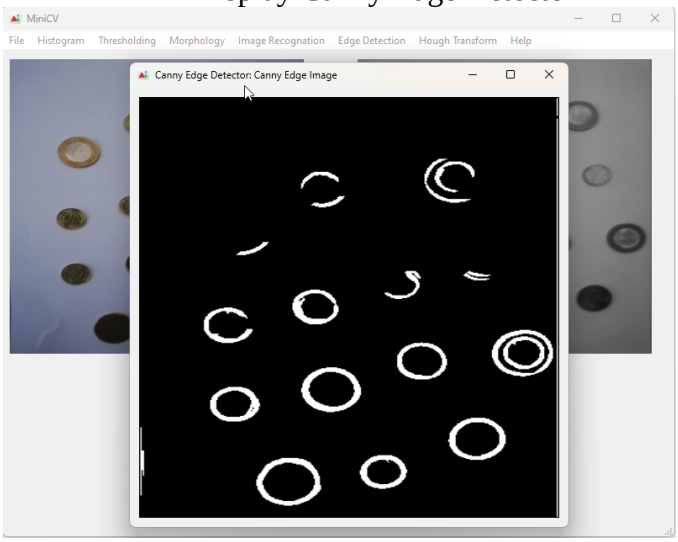
2 – Input Hysterisis Low Threshold



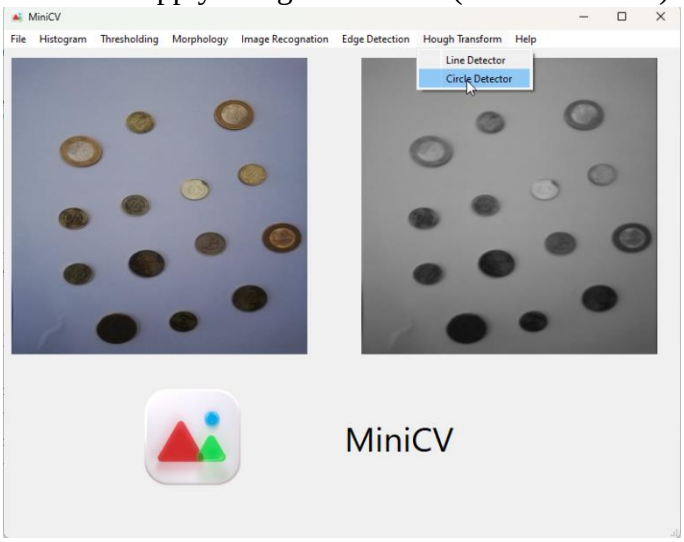
3 – Input Hysterisis High Threshold



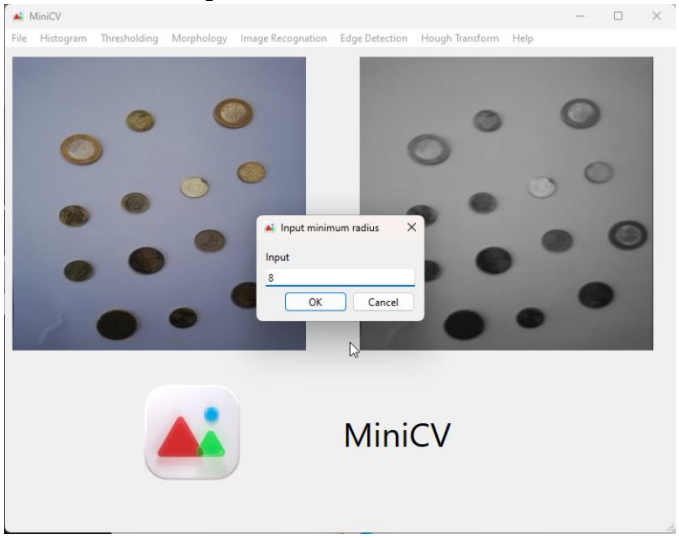
4 – Display Canny Edge Detector



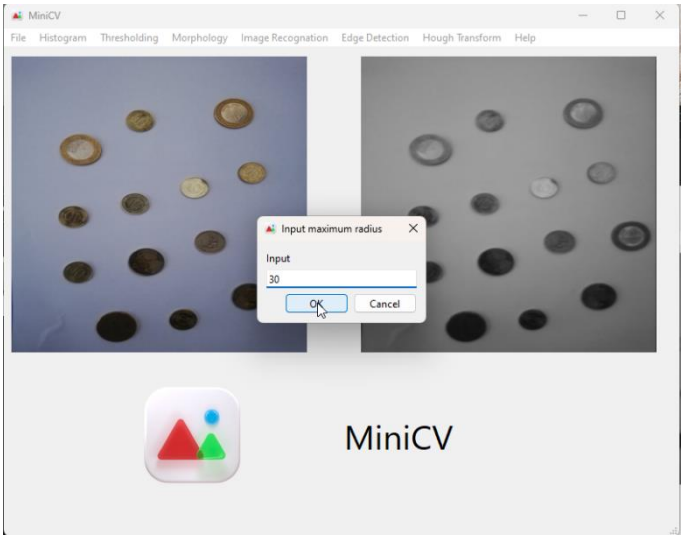
5 - Apply Hough Transform(Circle Detector)



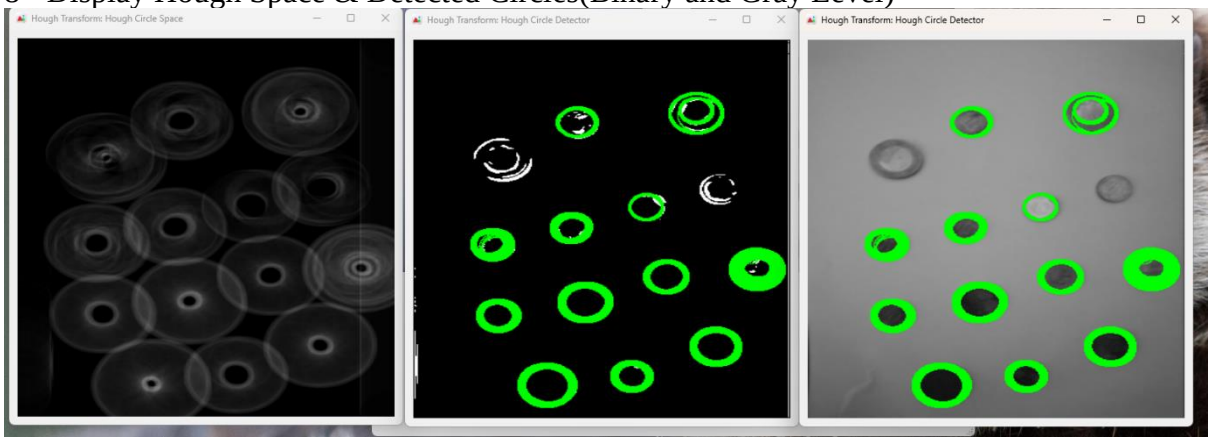
6 – Input Minimum Radius



7 – Input Maximum Radius



8 - Display Hough Space & Detected Circles(Binary and Gray Level)



Uygulamada Kullanılan Sınıflar Ve Temel Fonksiyon Prototipleri:

1) Sınıflar:

- BMPHeader
- BMPReader
- BMPWriter
- Image
- Mat
- Matrix
- Pixel

2) Fonksiyon Prototipleri:

- `Mat imread(const string& path);`
- `void imwrite(const string& path, Mat& mat);`
- `void bmp_to_grayscale(Mat& mat);`
- `void MainWindow::smoothening_gaussian_filter();`
- `void MainWindow::calc_gradient_n_direction();`
- `void MainWindow::canny_edge_detector();`
- `void hough_line_detector(Mat& mat);`
- `void draw_line(Mat& mat, int x0, int y0, int x1, int y1);`
- `void hough_circle_detector(Mat& mat);`
- `void draw_circle(Mat& mat, int center_x, int center_y, int radius, Pixel color);`