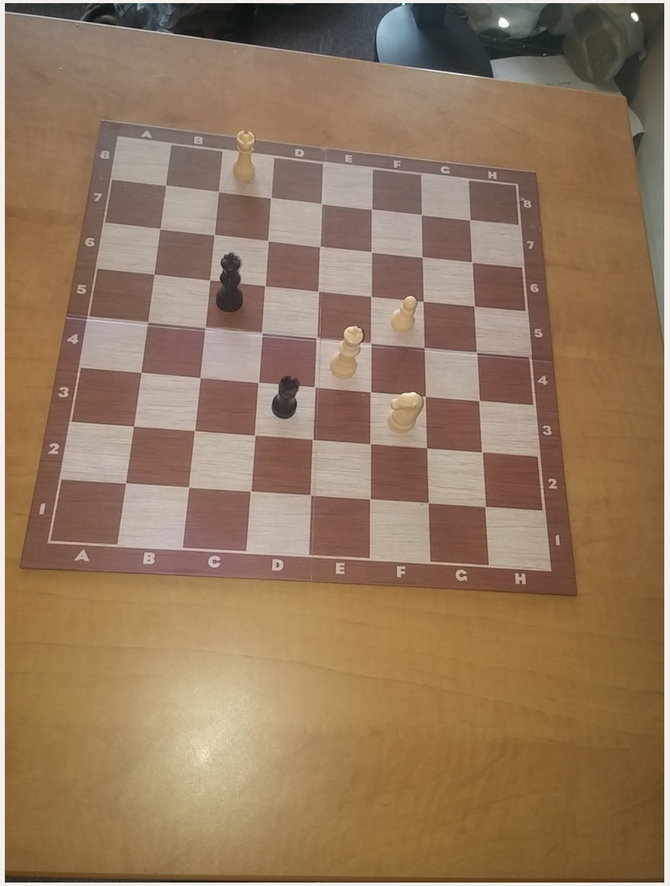
Staj Raporu

ZEYNEP BERDA AKKUŞ

10-14 Temmuz

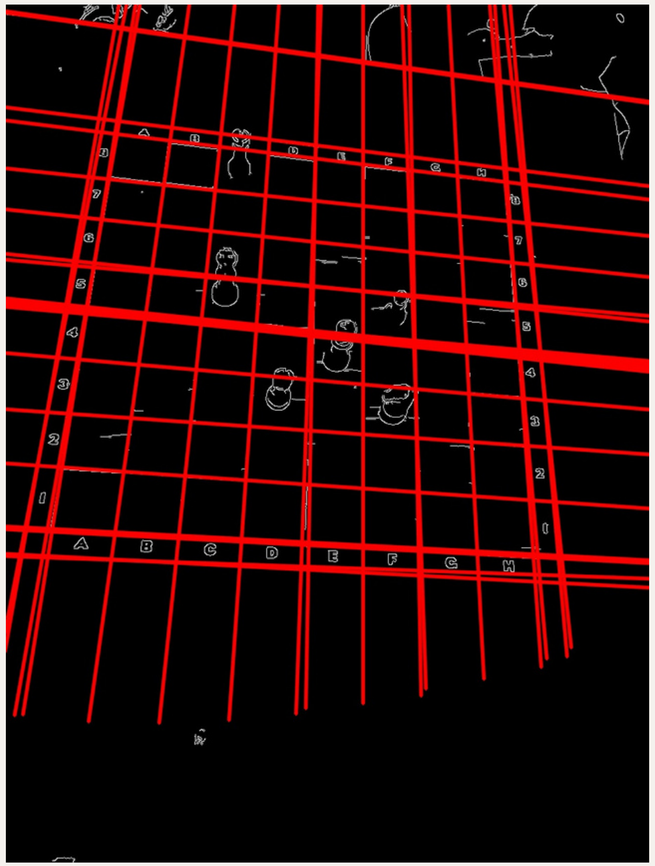
AMAÇ: COMPUTER VİSİON KISMINDA İLERLEMEK



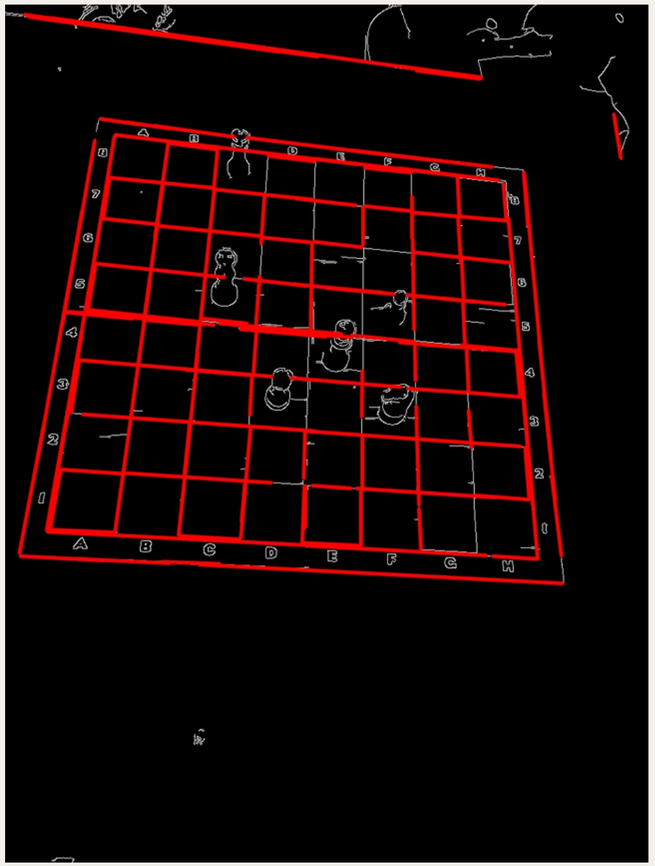
Örnek Fotoğraf

- Hough line detection
- Canny edge detection

HOUGH LINE DETECTİON

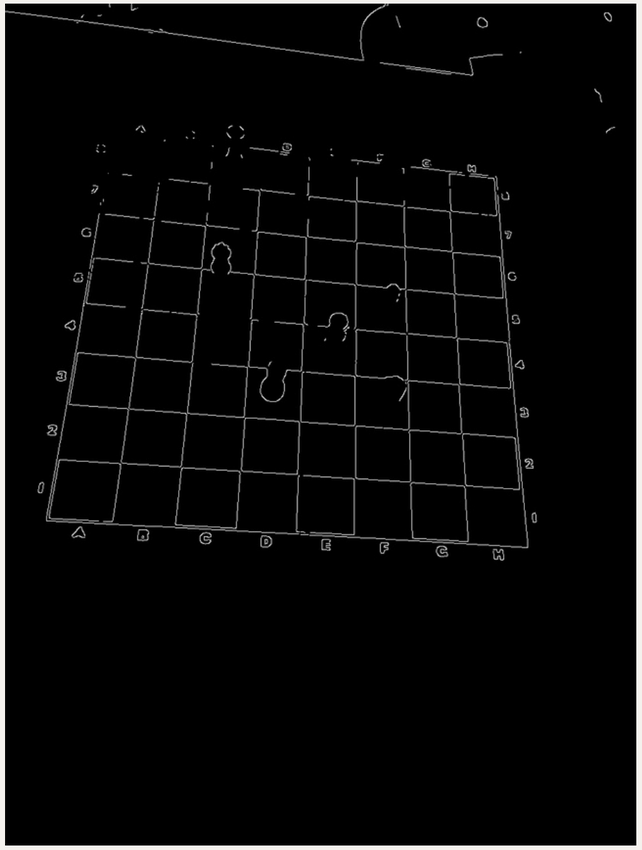


Standard Hough Line Transform

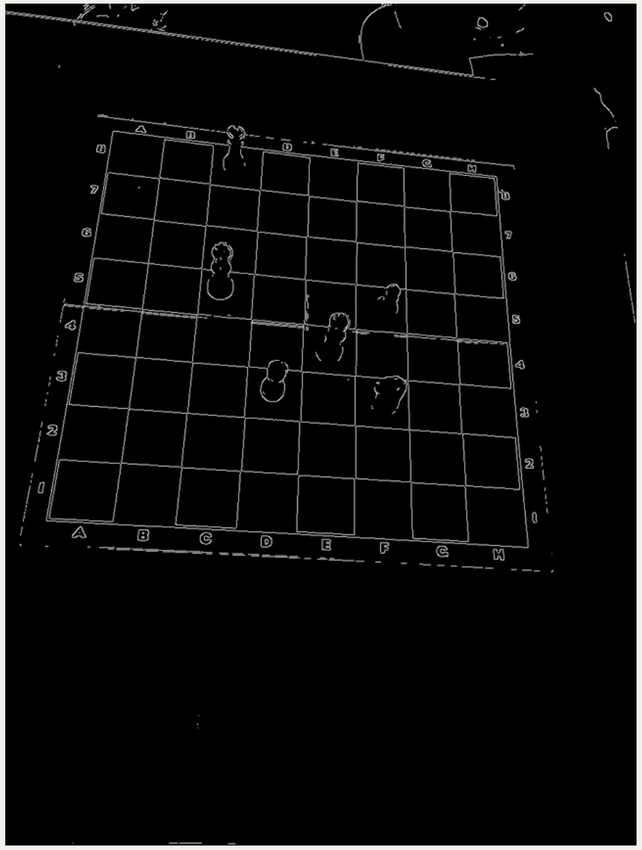


Probabilistic Hough Line Transform

CANNY EDGE DETECTİON

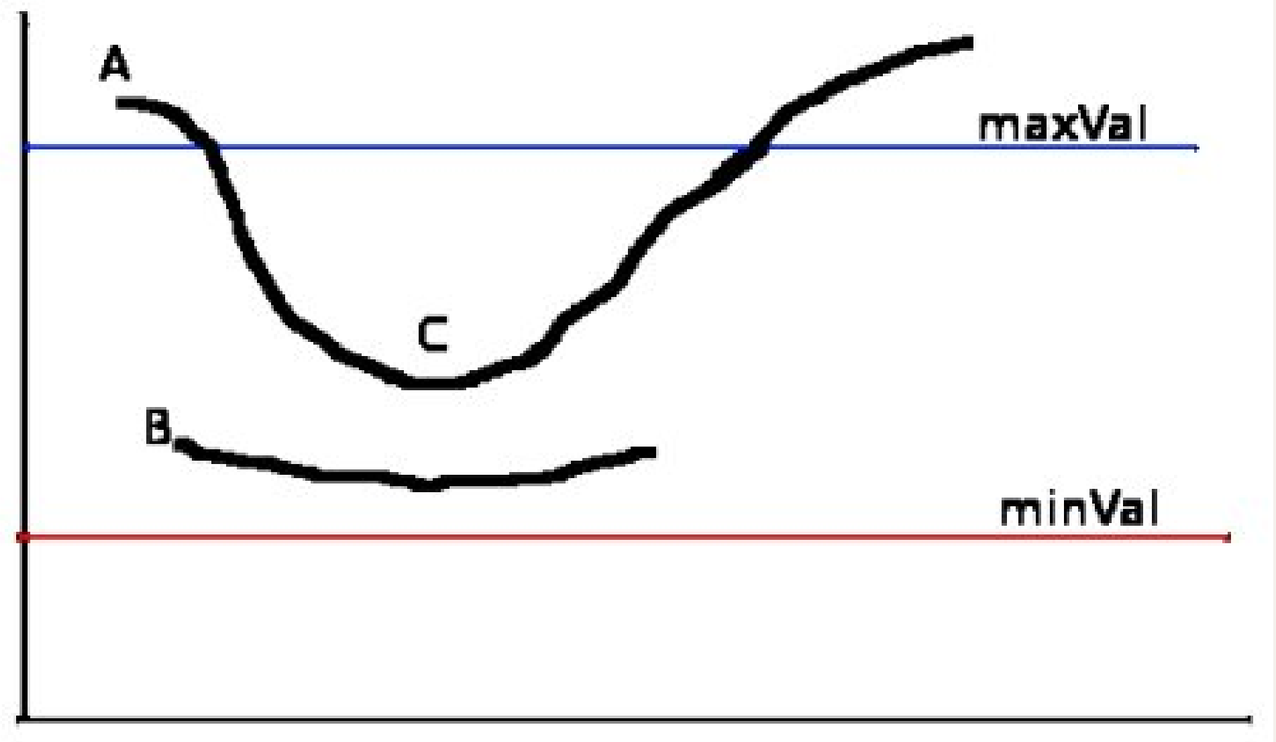


With Gaussian Blur function



Without Blur Canny Edge detection

The edge A is above the maxVal, so considered as "sure-edge". Although edge C is below maxVal, it is connected to edge A, so that also considered as valid edge and we get that full curve. But edge B, although it is above minVal and is in same region as that of edge C, it is not connected to any "sure-edge", so that is discarded. So it is very important that we have to select minVal and maxVal accordingly to get the correct result.

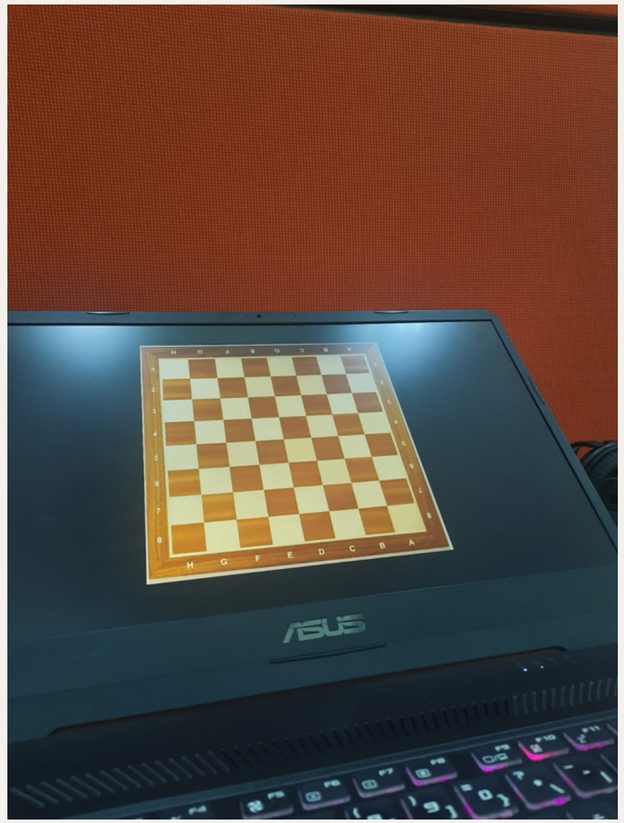


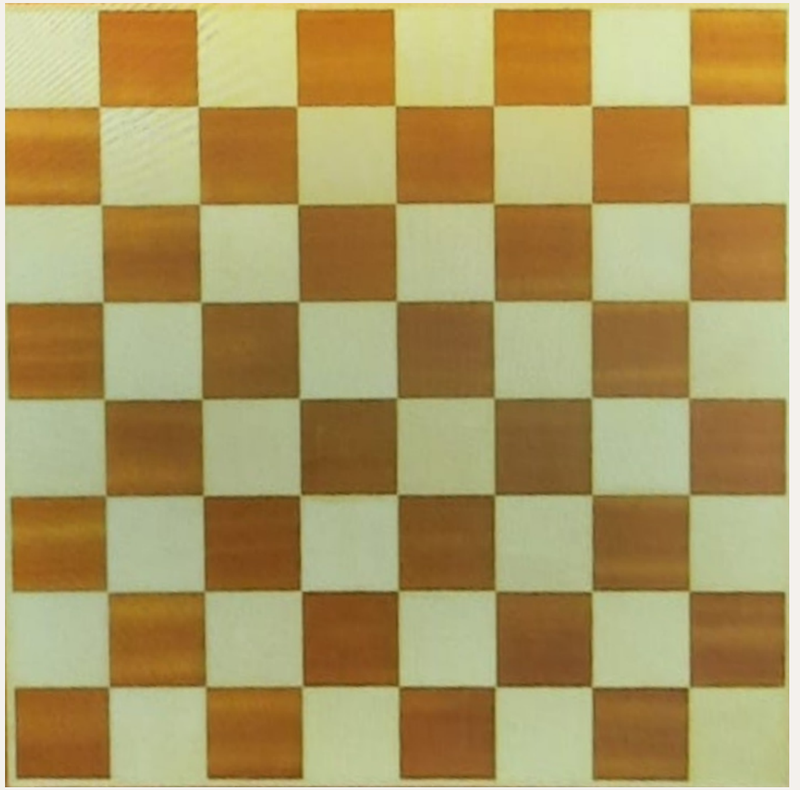
Tresholds

COMPUTER VISION - ÇIKARILAN SONUÇLAR:

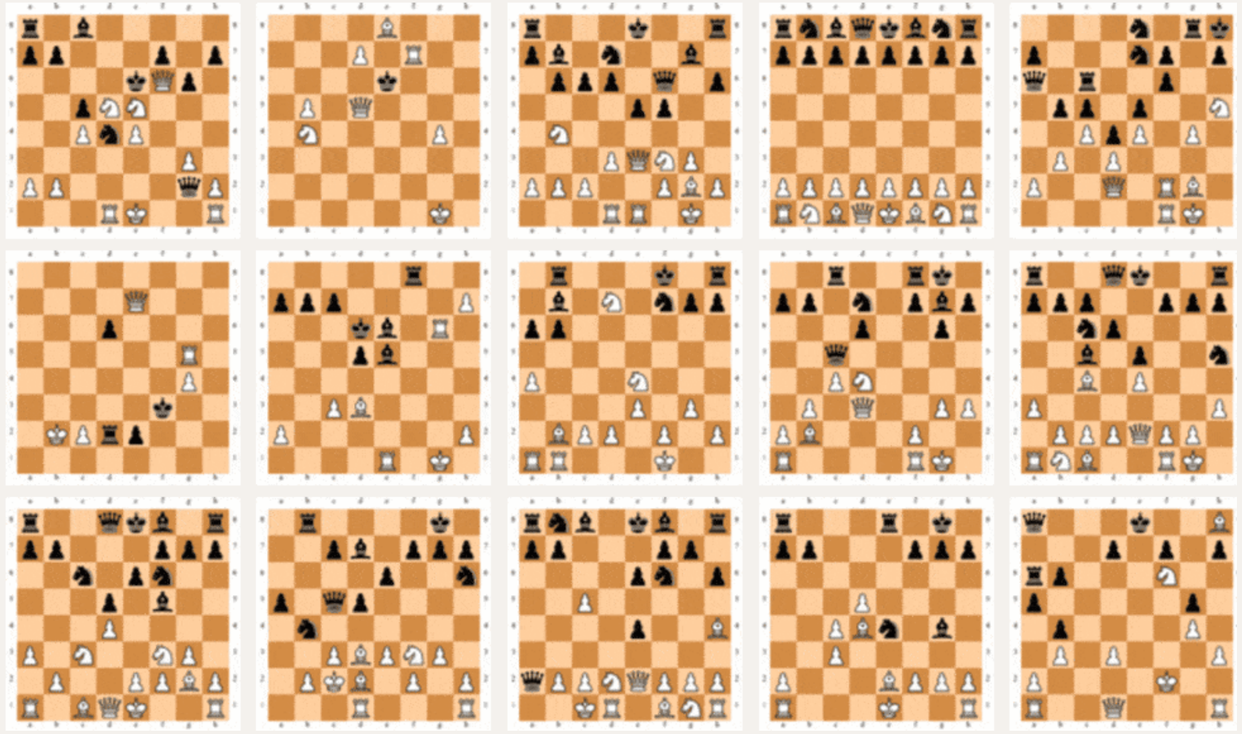
- Fotoğrafların parlaklığına ve kontrasına göre computer vision uygulamalarının sonuç başarısı değişmektedir.
- Parlaklık, kontrast ve ortama bağlı olarak yapılması gereken ayarlar şu şekilde sıralanmıştır:
    a. Gaussian filtresinin kullanılıp kullanılmayacağına karar verilmesi (noise reduction)
    b. Threshold değerlerine karar vermek
- Bu ayarların her fotoğraf için ayrı ayrı yapılma imkanı olmaması ve ortamın sabitlenmesinin istenmemesi üzerine deep learning yöntemleri denemek istenmiştir.

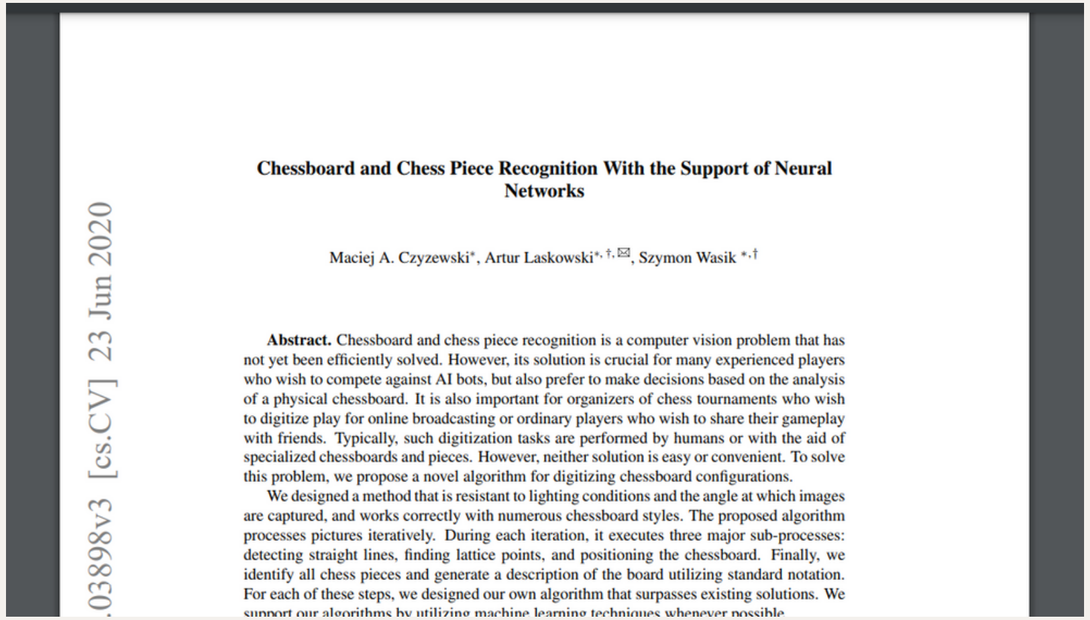
DEEP LEARNING - ILK HEDEF : BOARD DETECTION





YÖNTEM:





Chessboard and Chess Piece Recognition With the Support of Neural Networks

Maciej A. Czyzewski*, Artur Laskowski*,†,✉, Szymon Wasik *,†

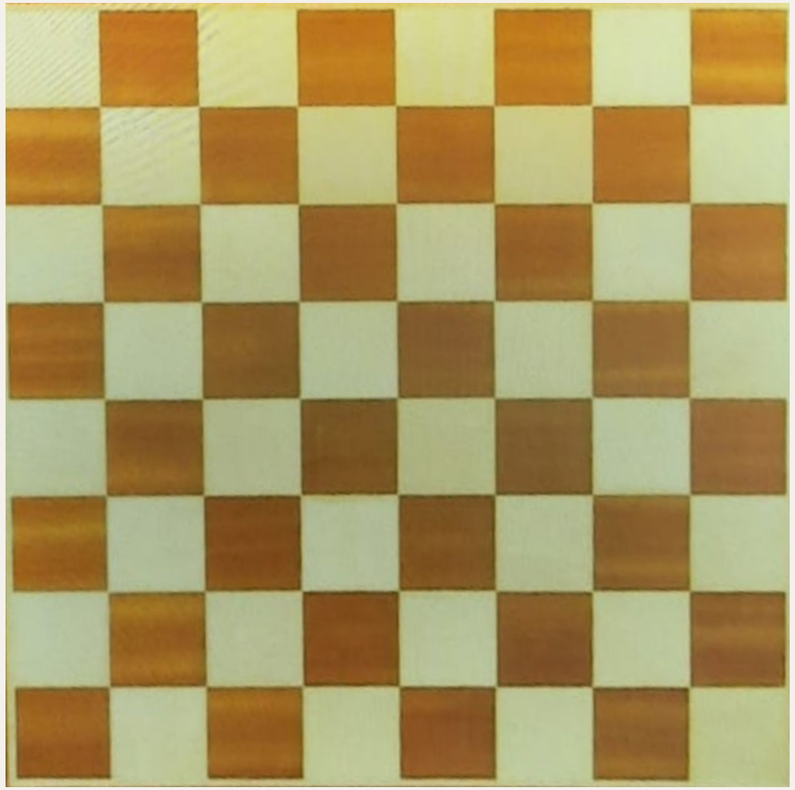
Abstract. Chessboard and chess piece recognition is a computer vision problem that has not yet been efficiently solved. However, its solution is crucial for many experienced players who wish to compete against AI bots, but also prefer to make decisions based on the analysis of a physical chessboard. It is also important for organizers of chess tournaments who wish to digitize play for online broadcasting or ordinary players who wish to share their gameplay with friends. Typically, such digitization tasks are performed by humans or with the aid of specialized chessboards and pieces. However, neither solution is easy or convenient. To solve this problem, we propose a novel algorithm for digitizing chessboard configurations.

We designed a method that is resistant to lighting conditions and the angle at which images are captured, and works correctly with numerous chessboard styles. The proposed algorithm processes pictures iteratively. During each iteration, it executes three major sub-processes: detecting straight lines, finding lattice points, and positioning the chessboard. Finally, we identify all chess pieces and generate a description of the board utilizing standard notation. For each of these steps, we designed our own algorithm that surpasses existing solutions. We support our algorithms by utilizing machine learning techniques whenever possible

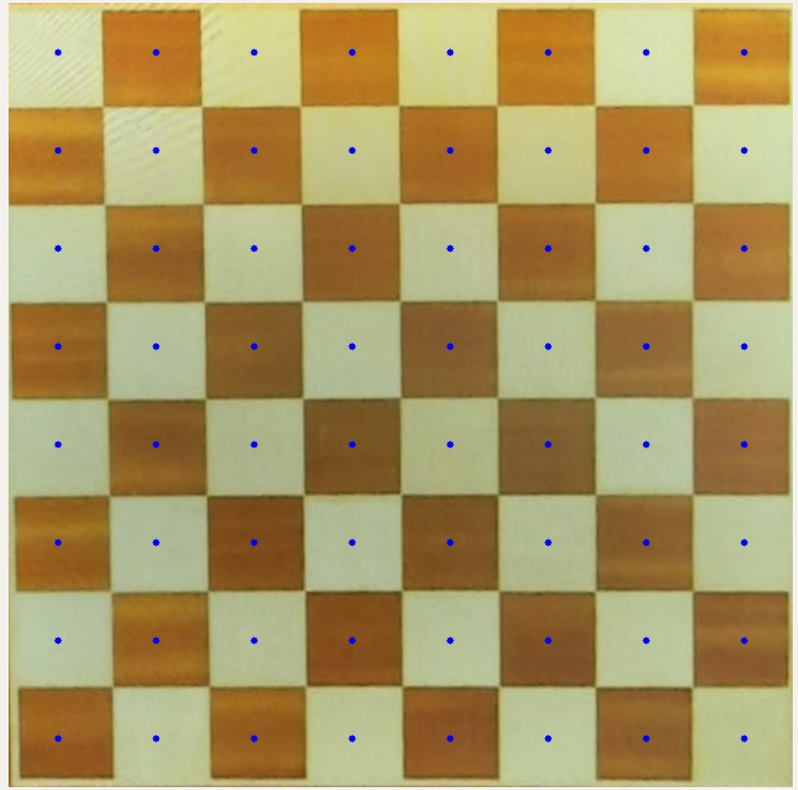
https://github.com/maciejczyzewski/neural-chessboard/tree/draft

17-21 Temmuz

DEEP LEARNING - ILK HEDEF : BOARD DETECTION



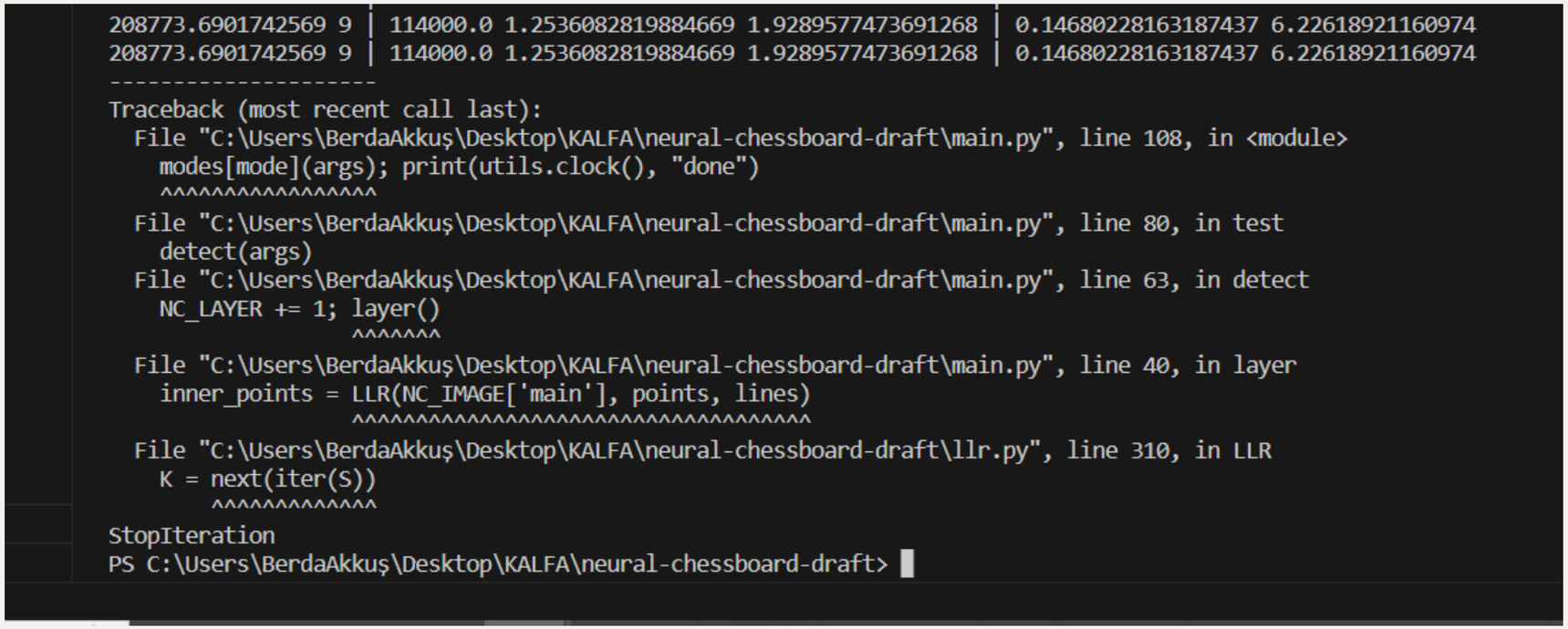
find middle points



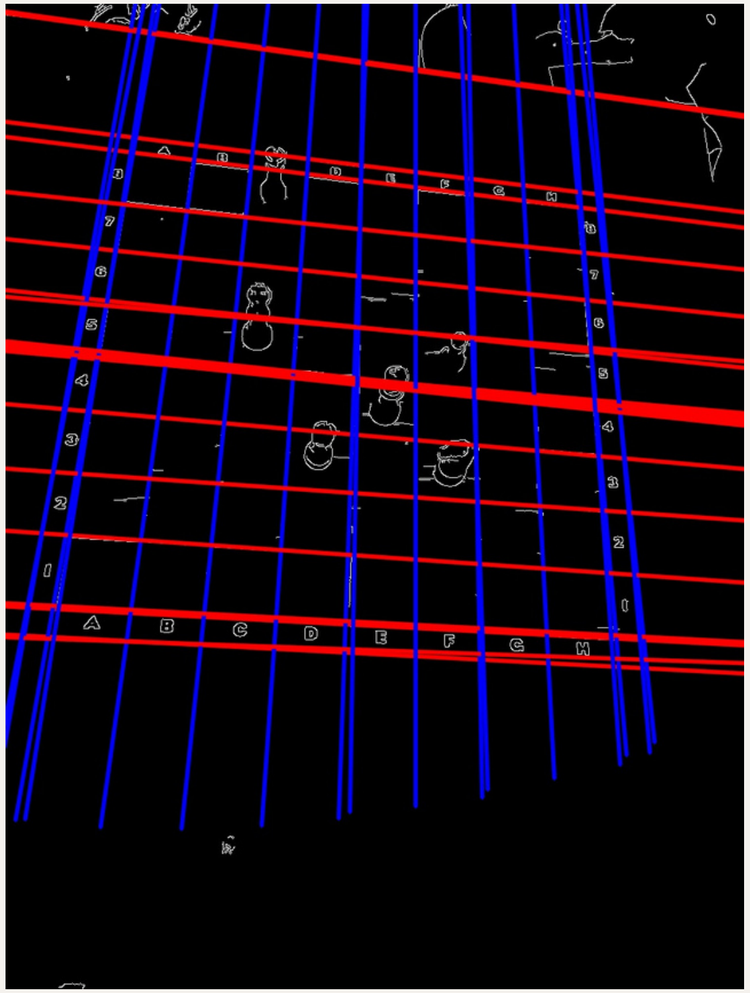
output: pixel array : [(x1,y1),(x2,y2),...]

KARŞILAŞTIĞIMIZ PROBLEMLER:

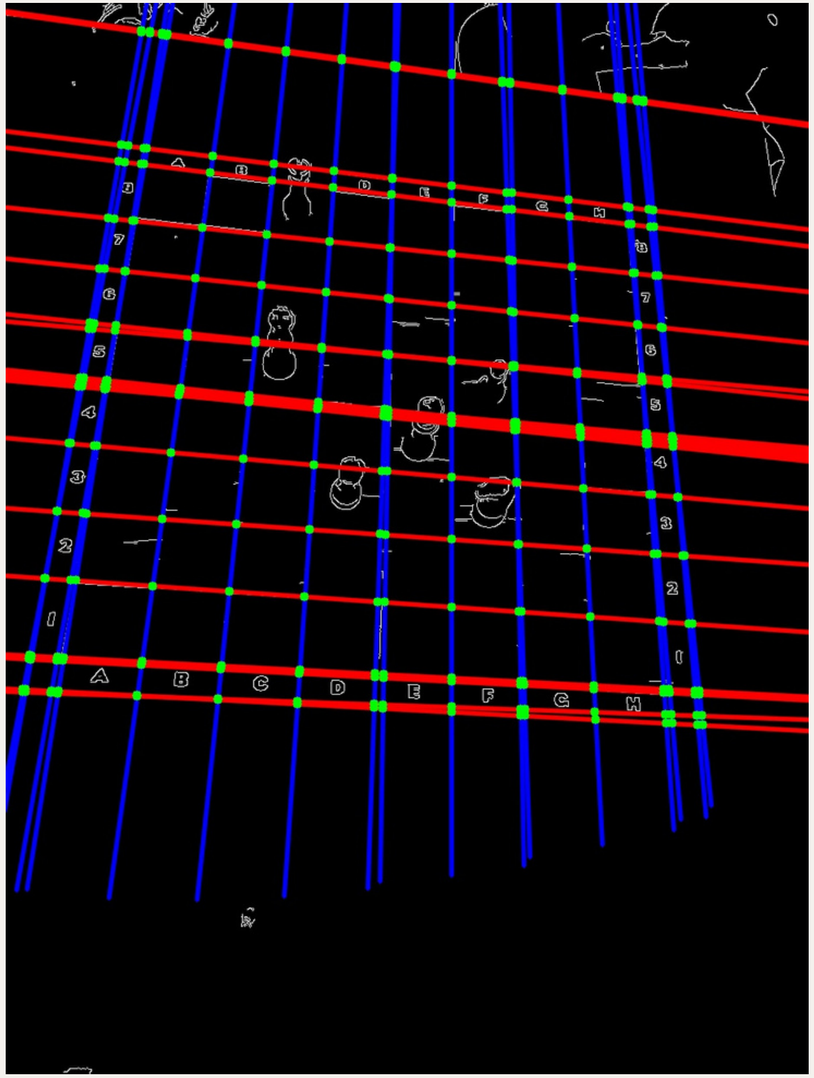
- Kullandığımız neural network kodu 2017 de yazılmış bir kod olması sebebiyle (tahminen) bazı fotoğraflarda hata vermeye başladı. ör:



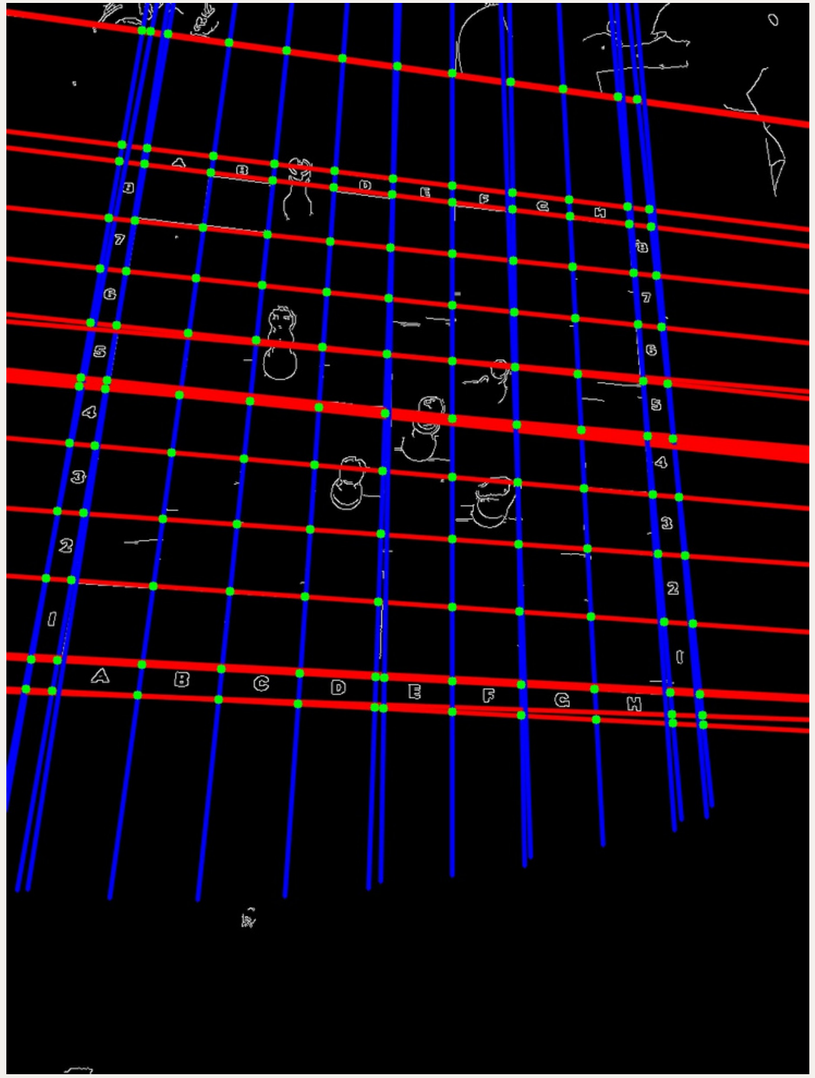
Kamera kalibrasyonu sayesinde chessboard'un orta noktasının x,y,z değerlerinin bulunabildiği ROS ekibi tarafınan tespit edilmiş olup bu noktadan itibaren vision ekibinin amacı chessboard'un en küçük karelerinin birinin kenar boyutunu hesaplamak olarak değişmiştir.



- Karşılaşılan problemler neticesinde hazı kod kullanmaktan vazgeçilmiştir. Klasik vision yöntemleriyle hesap yapılmaya çalışılacaktır.
- Bu doğrultuda hough line tranform kullanılarak bulunan doğrular dikey ve yatay olacak şekilde sınıflandırılmıştır. Burdaki amaç en küçük karelerden birinin kenar uzunluğunu bulabilmektir.

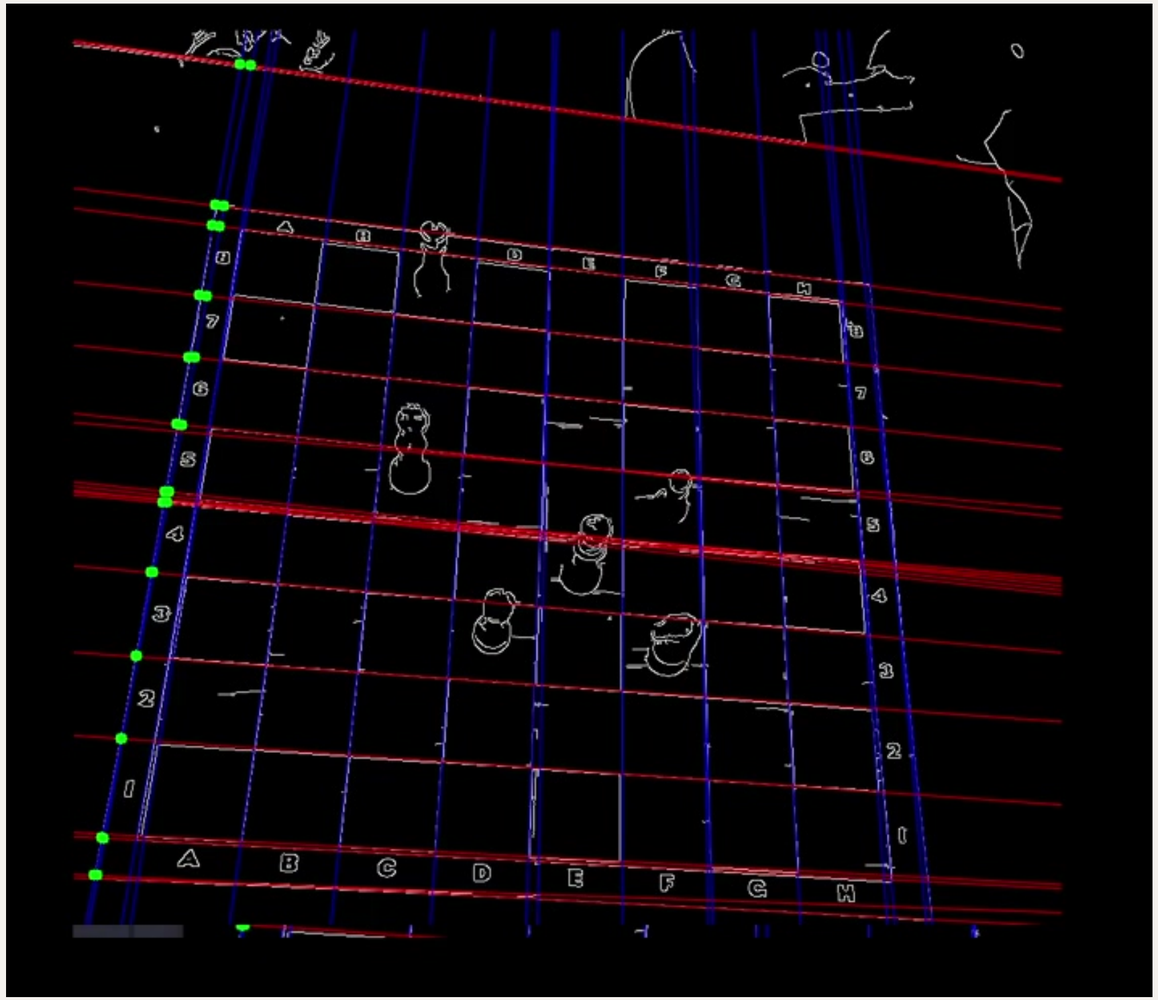


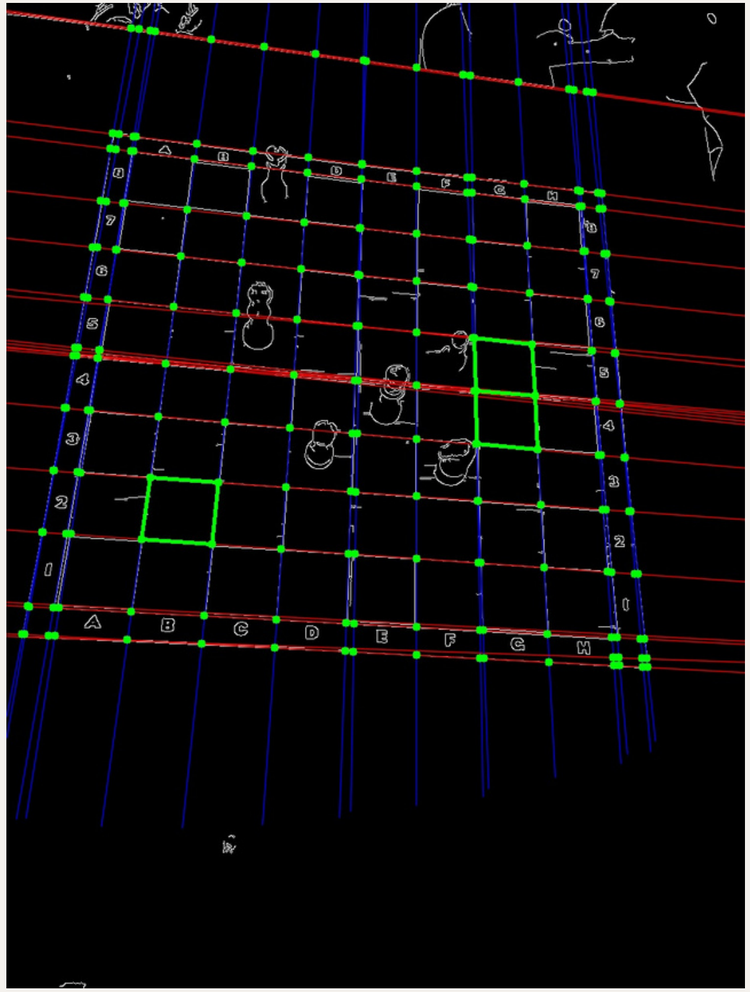
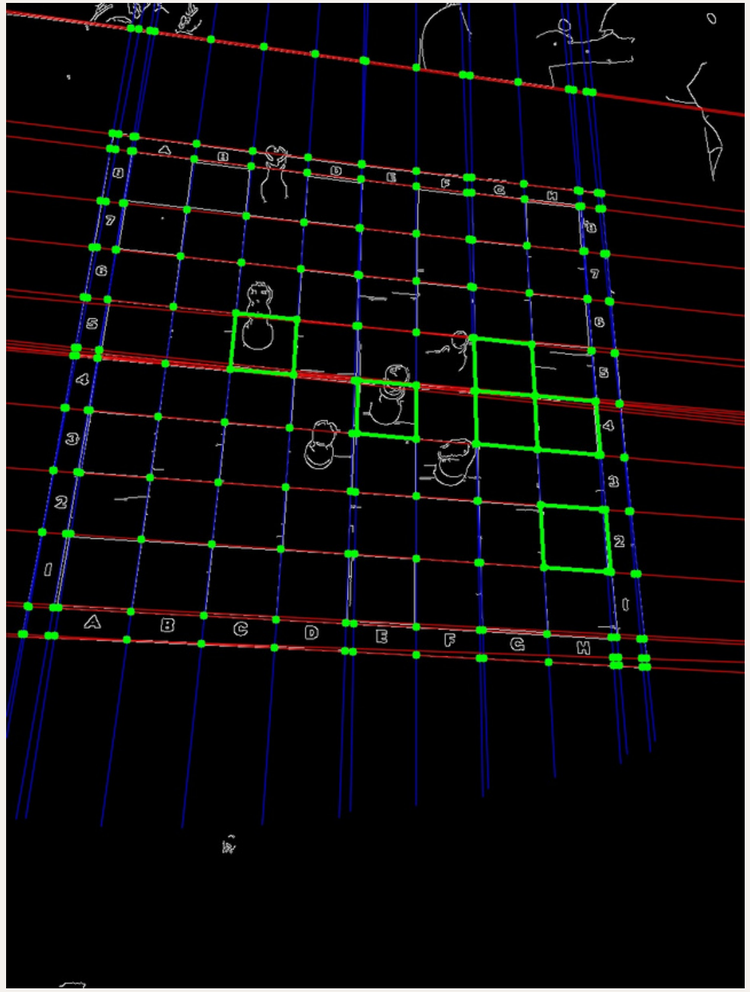
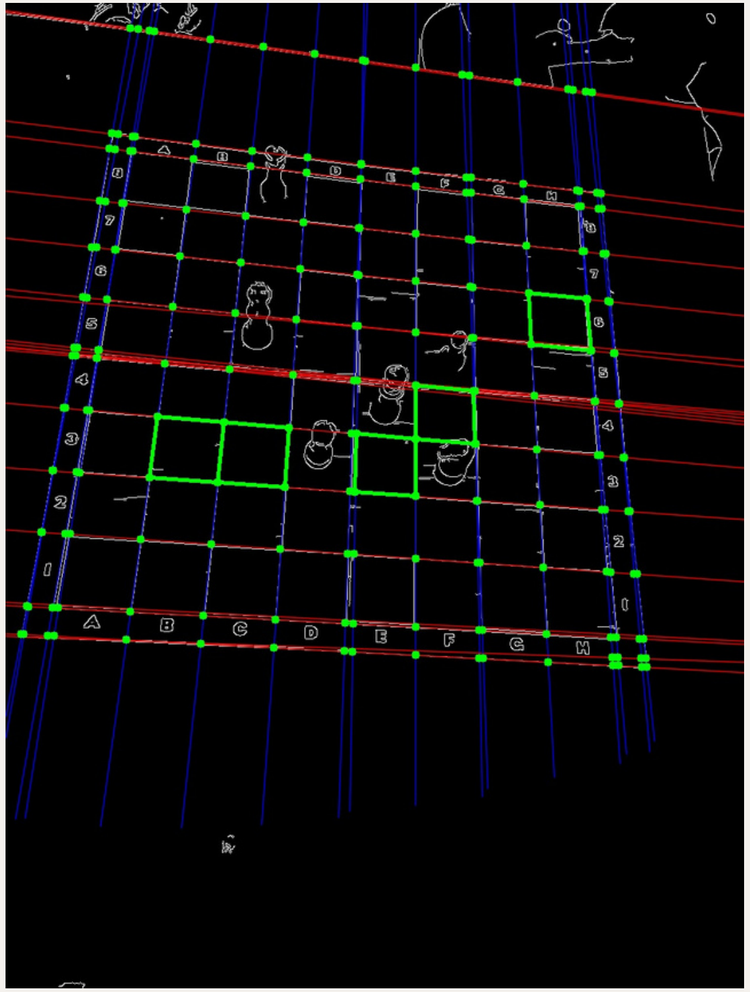
- Line segmentation sonrası kesişim noktaları belirlenmiştir.



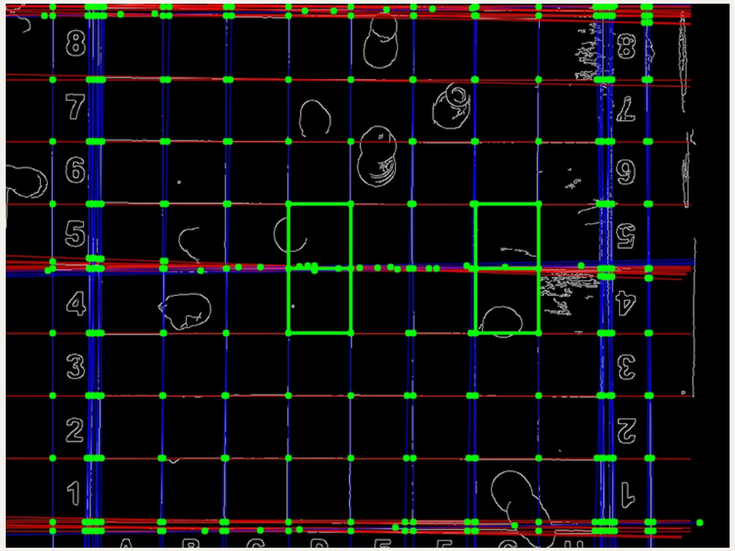
- Tekrar eden kesişim noktaları silinmiştir.

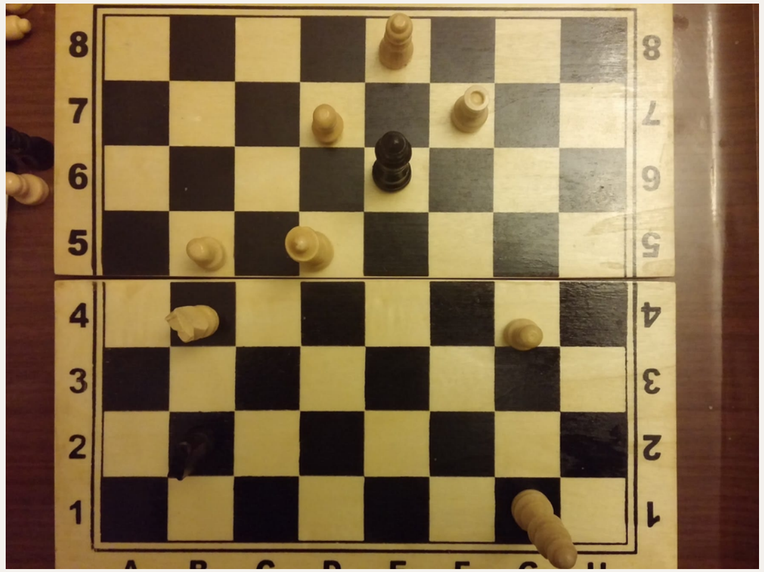
Gruplandırılmış Kesişimler:

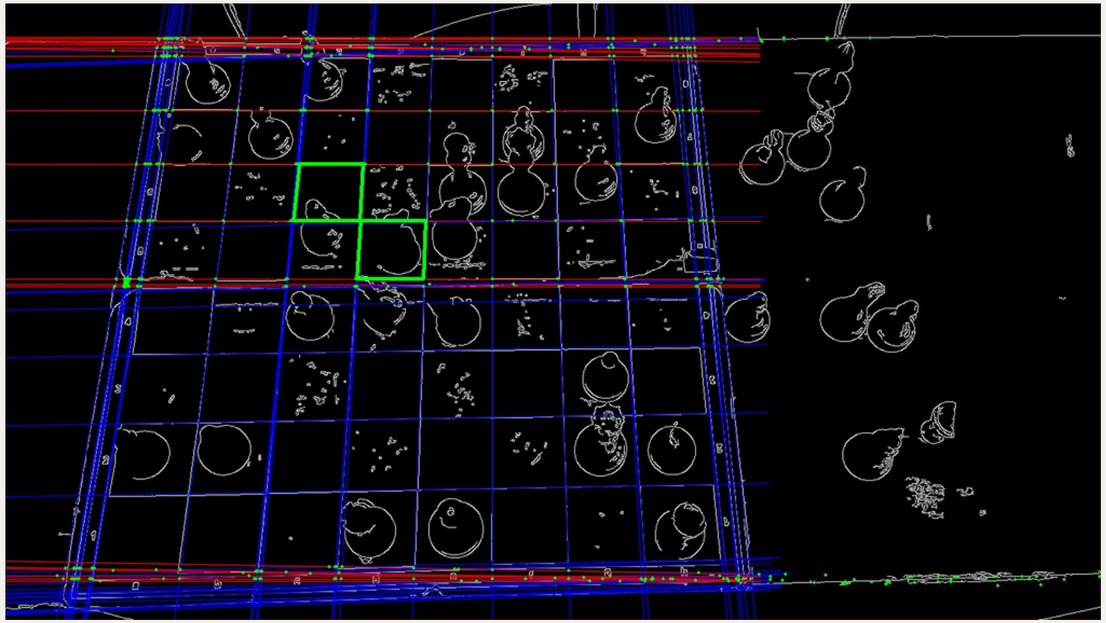


non_deterministic

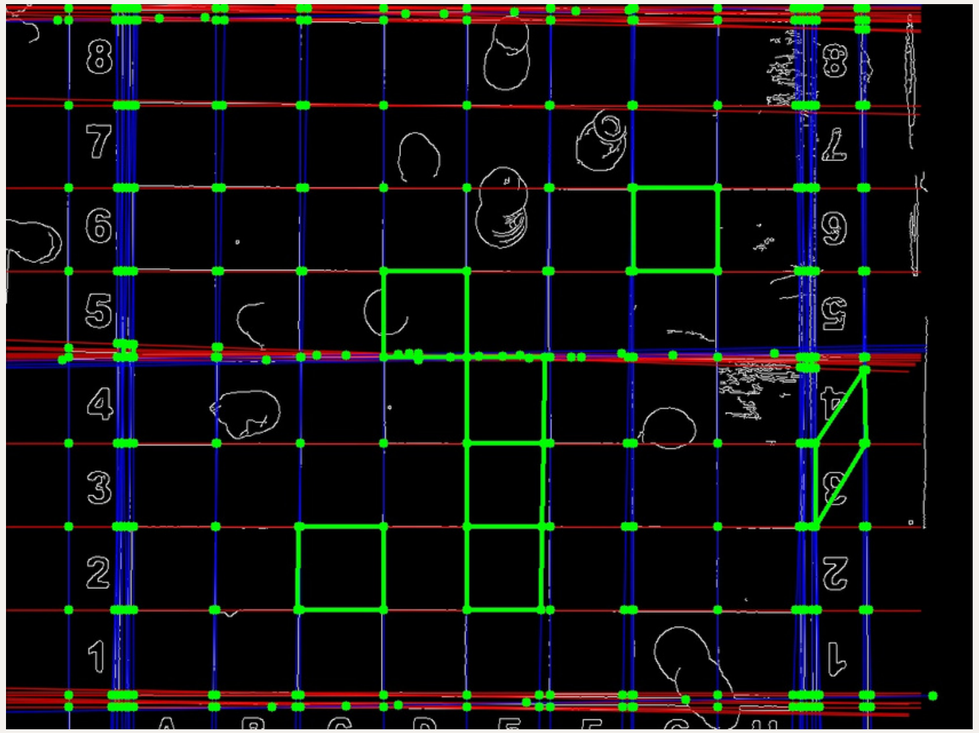




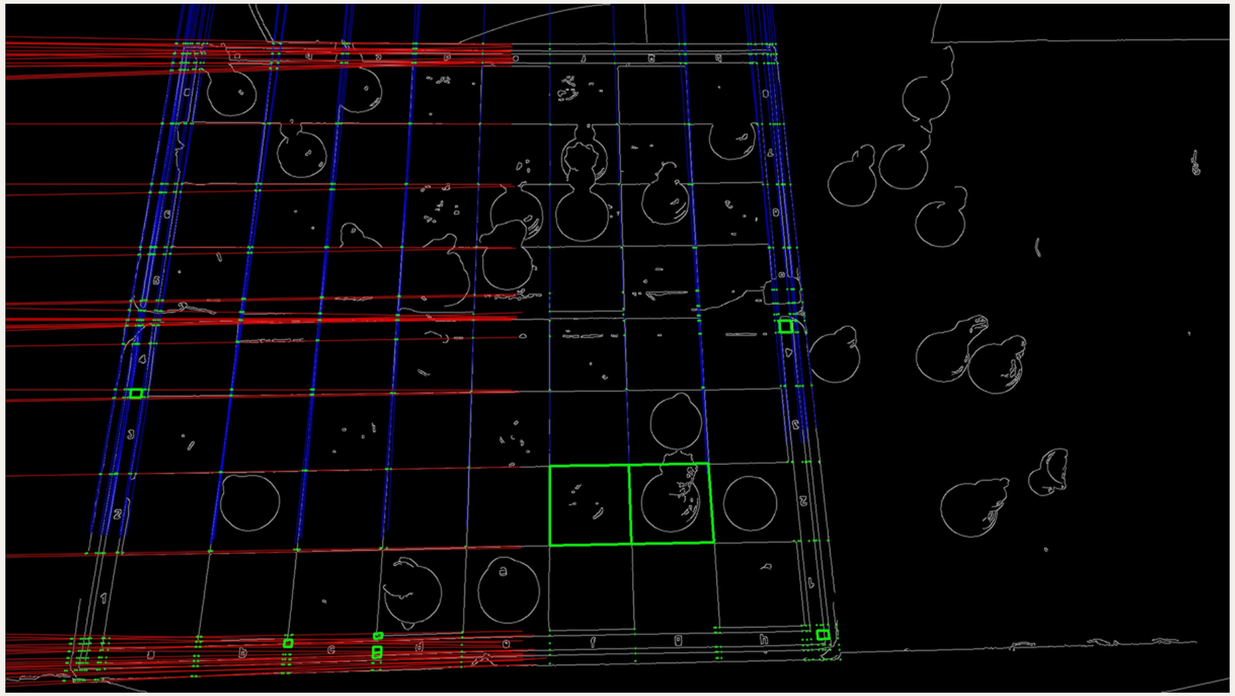


Karşılaşılan hatalar:

```
ect_hazır.py
hello
[89.0, 8.0, 91.0, 103.83159442096611]
PS C:\Users\BerdaAkkuş\Desktop\KALFA\neural-chessboard-draft> & C:/Users/BerdaAkkuş/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/Users/BerdaAkkuş/Desktop/KALFA/comp_v/chessdet
ect_hazır.py
hello
[96.0, 103.83159442096611, 87.0, 93.0, 89.0, 91.0, 91.0, 91.0]
PS C:\Users\BerdaAkkuş\Desktop\KALFA\neural-chessboard-draft> & C:/Users/BerdaAkkuş/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/Users/BerdaAkkuş/Desktop/KALFA/comp_v/chessdet
ect_hazır.py
hello
Traceback (most recent call last):
  File "c:\Users\BerdaAkkuş\Desktop\KALFA\comp_v\chessdetect_hazır.py", line 184, in <module>
    if mesafe_bul(intersections[Dikey][yatay],intersections[Dikey+1][yatay])-mesafe_bul(intersections[Dikey][yatay],intersections[Dikey][yatay+1])<10 \
                                              ~~~~~~~~~~~~~~~~~~~~~~~~^^^^^^^
IndexError: list index out of range
PS C:\Users\BerdaAkkuş\Desktop\KALFA\neural-chessboard-draft>
```

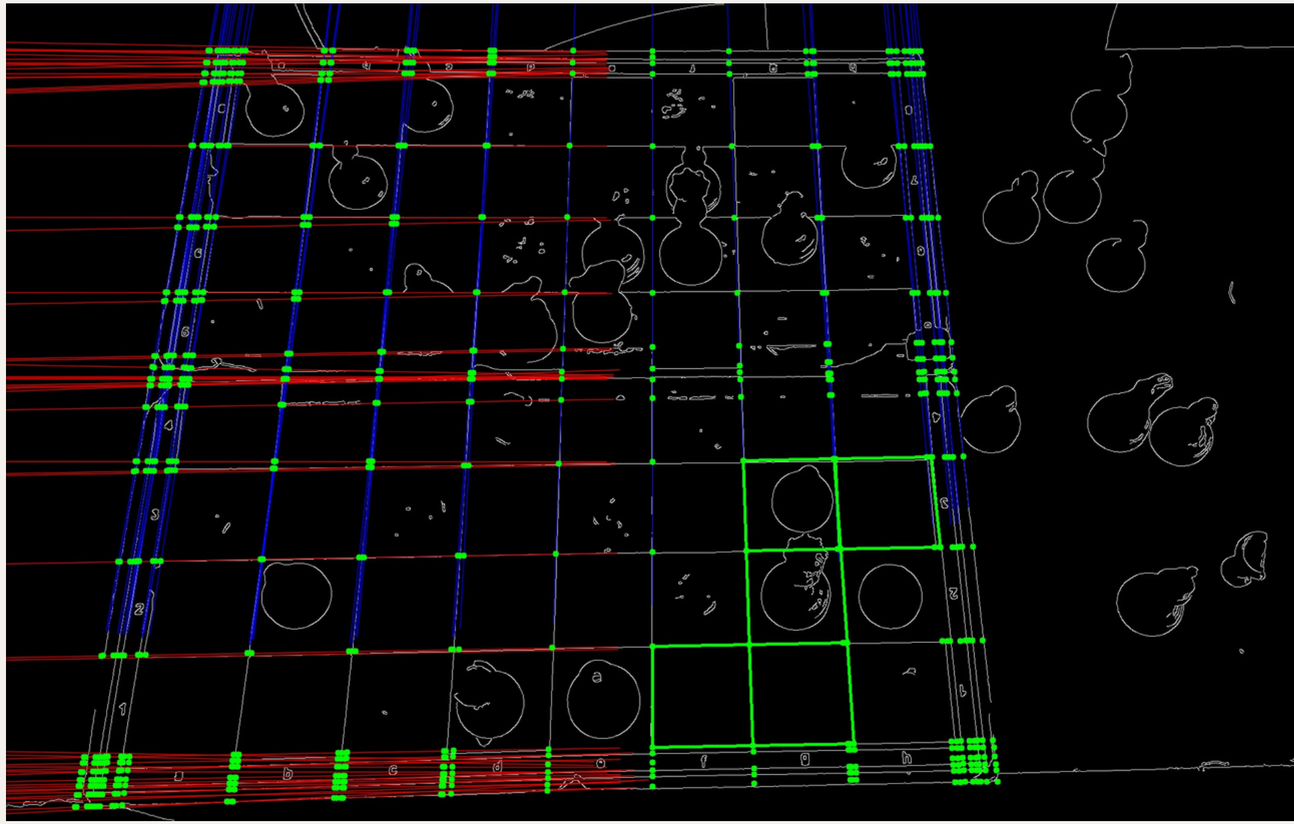


1. Boş küme, kare bulunamaması durumu
2. Eşkenar dörtgen, kenarların eşit olması ancak dik açı oluşmaması durumu
3. Kenar ölçüsü alt sınırı olmadığı için minnacık kayerleri kare sanma durumu



4. Hiç kesişim bulamaması durumu,
5. Kare oluşturamayacağı kenar noktaları seçme durumu
6.Line gruplamada yanlışlıklar

```
Sonucun:  158.02847958644557
PS C:\Users\BerdaAkkuş\Desktop\KALFA\comp_v> python3 .\chessdetect_hazır.py
Merhaba! Chessboarddaki en kücük karelerin ortalama kenar uzunluğunu bulma koduna hoş geldin.
Bulunan kare sayisi: 5
bulunan kenar uzunluklar:
 [159.02829936838285, 161.02794788483146, 163.02760502442524, 157.0286598045083, 151.02979838429235]
TEBRİKLER!!
Sonucun:  159.02829936838285
PS C:\Users\BerdaAkkuş\Desktop\KALFA\comp_v>
```



Sonuç:

```
TEBRİKLER!!
PS C:\Users\BerdaAkkuş\Desktop\KALFA\comp_v> python3 .\chessdetect_hazır.py
Merhaba! Chessboarddaki en kücük karelerin ortalama kenar uzunluğunu bulma koduna hoş geldin.
Bulunan kare sayisi: 3
bulunan kenar uzunluklar:
 [109.65856099730654, 135.36986370680884, 114.6298390472568]
TEBRİKLER!!
Sonucun:  114.6298390472568
PS C:\Users\BerdaAkkuş\Desktop\KALFA\comp_v>
```