Introduction to Computational and Biological Vision

Checkerboard Recognition

Fledel Yuval

Khait Vitaly

Introduction

- Allowing a computer to play checkers against a human player on a physical board.
- The first step is to “see” the board.
  - □ Find the board in the picture.
  - □ Find the pieces on the board.
  - □ Identify the pieces.

Assumptions

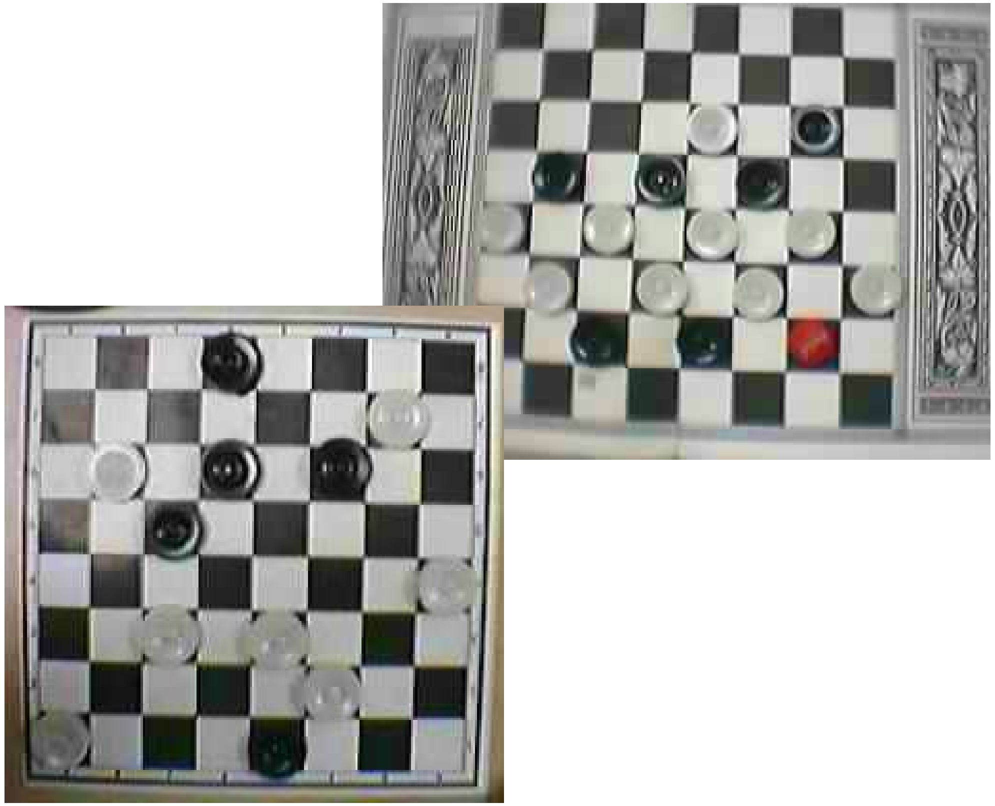
- Environment: A computer, a player, a checkers board, a camera.
- The checkers board is entirely contained in the image.
- The pieces are smaller than the cells.
- The image is taken close to zenith over the board.
- The background of the image has a relatively small number of lines.
- Playable cells are dark.

Goals

- Identify the board in the picture
  - Orientation
  - Location
- Identify the pieces in the pictures.
  - Color
  - Location
- Emphasis on low false positives.

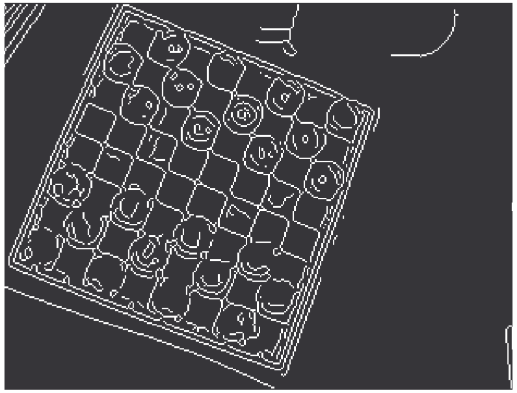
Step 0 - Acquiring

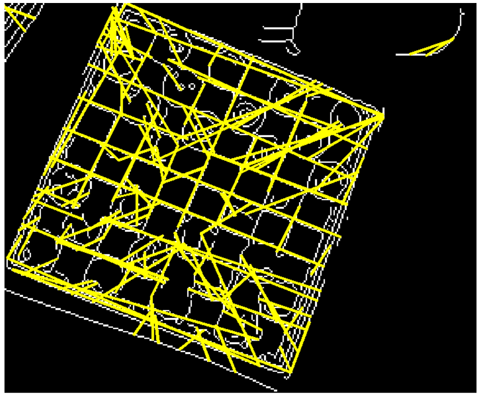
- Acquiring the Image using a cheap webcam.
  - Low resolution
  - Barely focused
  - Noisy
  - Bad Lighting
  - Old boards.

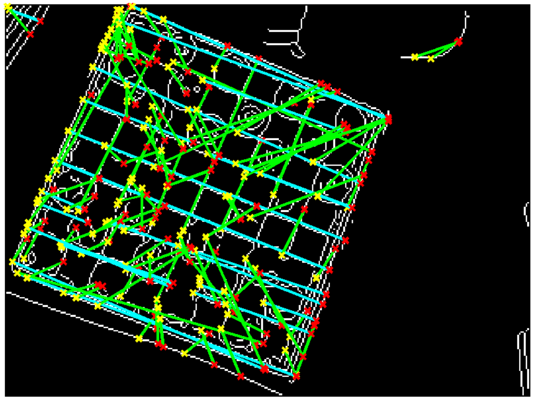


Step 1 – Board Orientation

- Detect the board orientation:
  - Canny edge detection.
  - Hough transform for lines.
  - Voting for orientations.

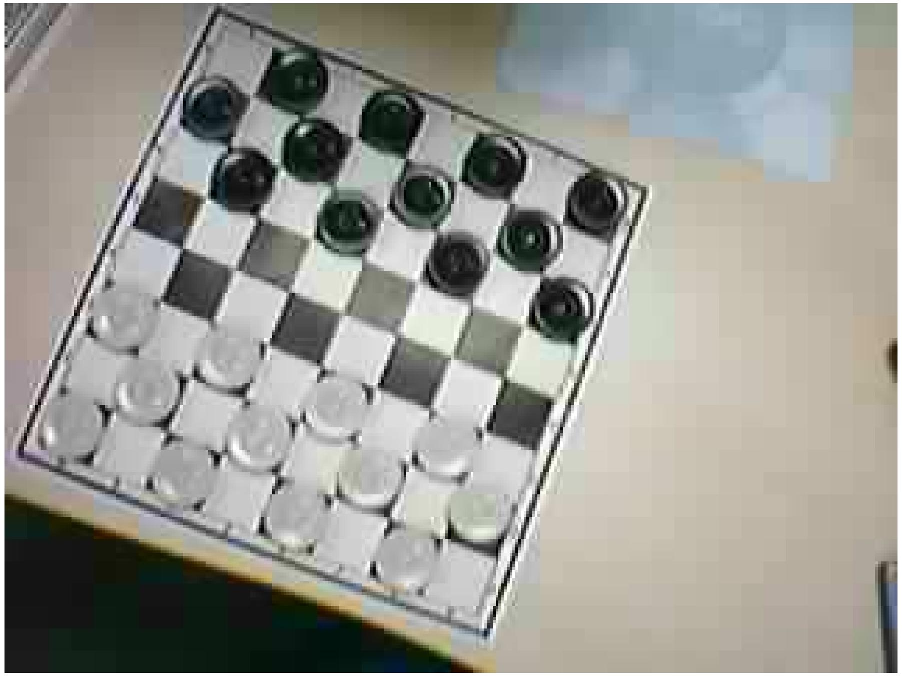




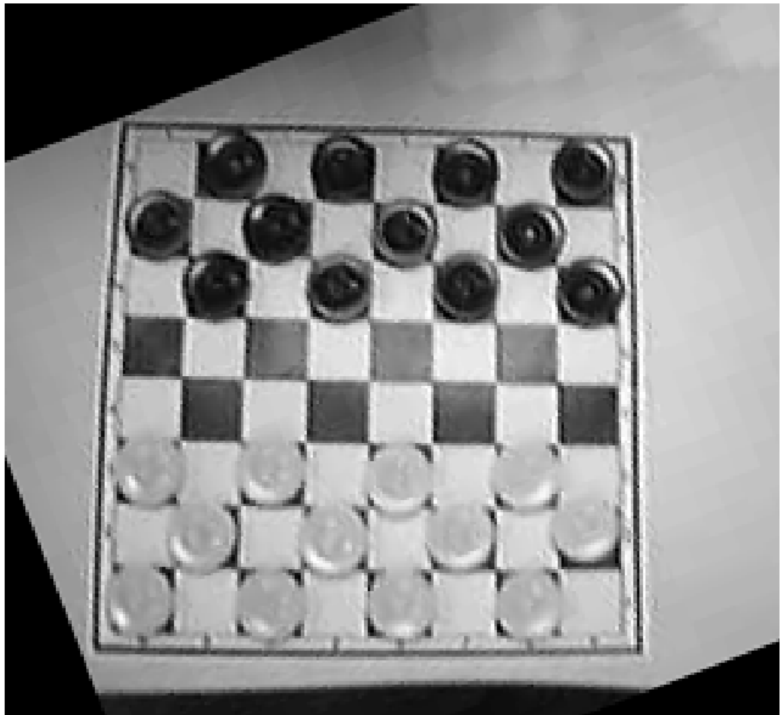


Step 1 - Board Orientation

- Aligning with the axis



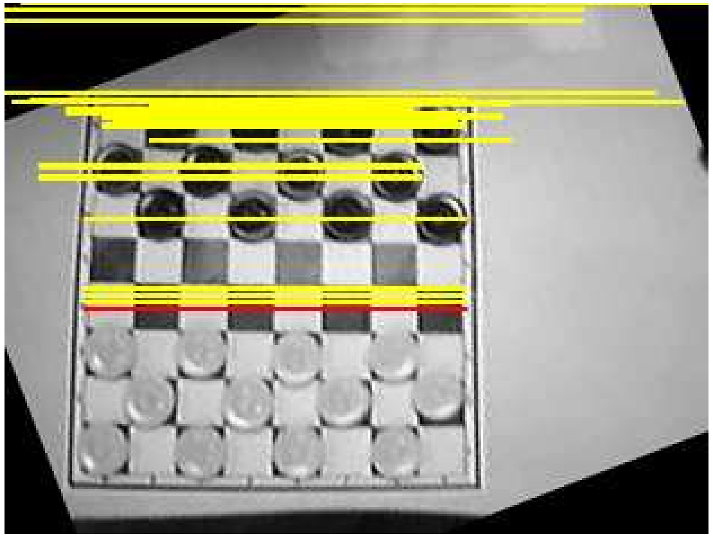
Before

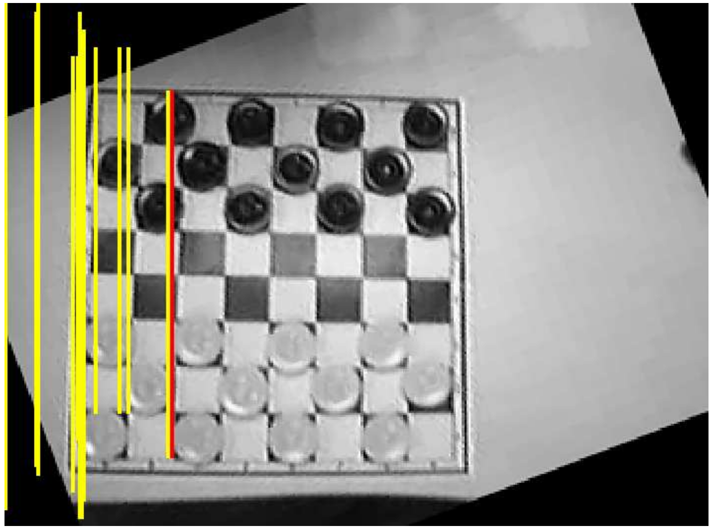


After

Step 2 – Board Location

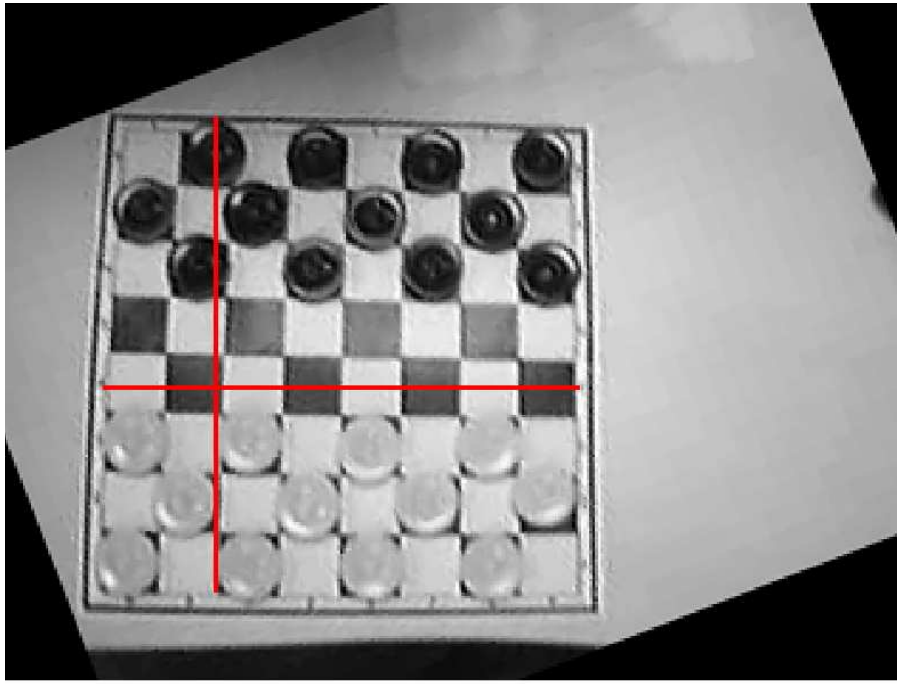
- Pattern matching: match a line of alternating black and white: ■ ■ ■ ■



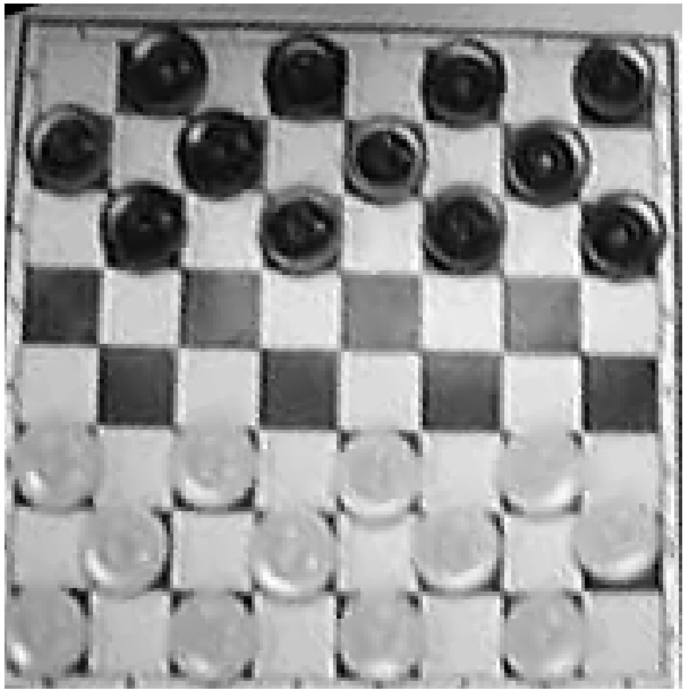


Step 2 – Board Location

- Result of image scanning:



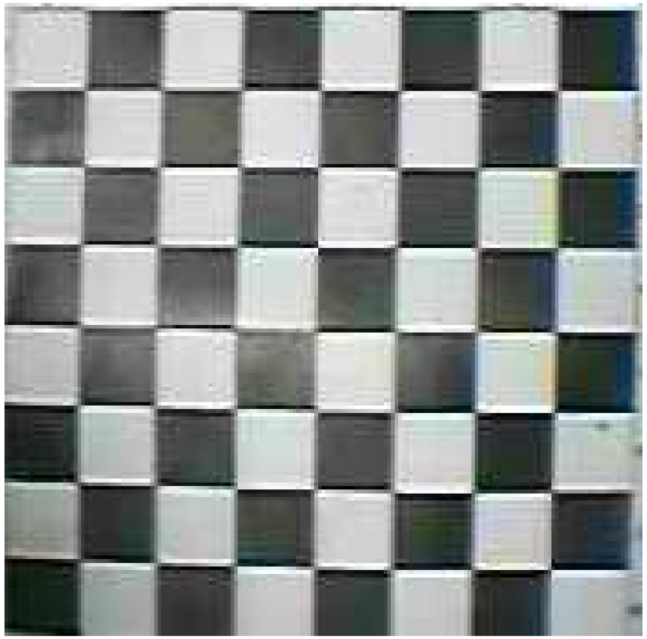
Best match



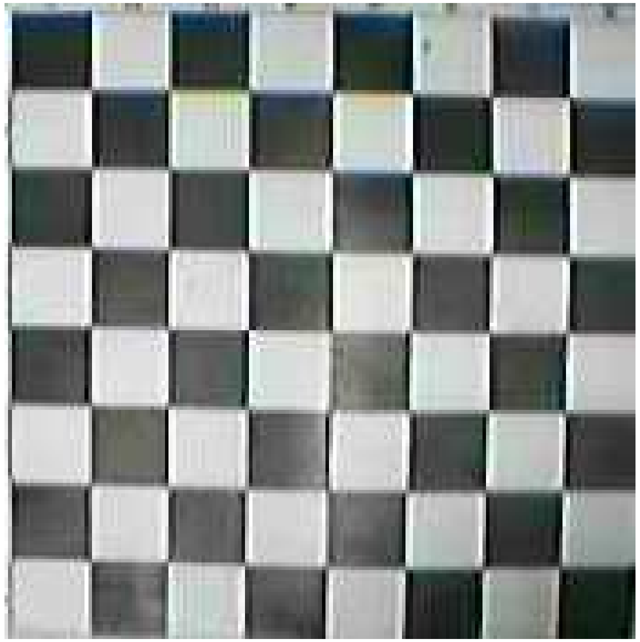
Cropped image

Step 3 – Board Parity

- Statistical detection of luminance over the cells:



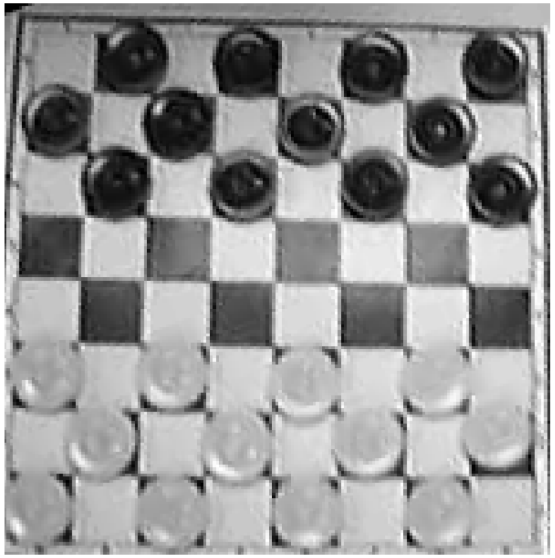
Top-left-white

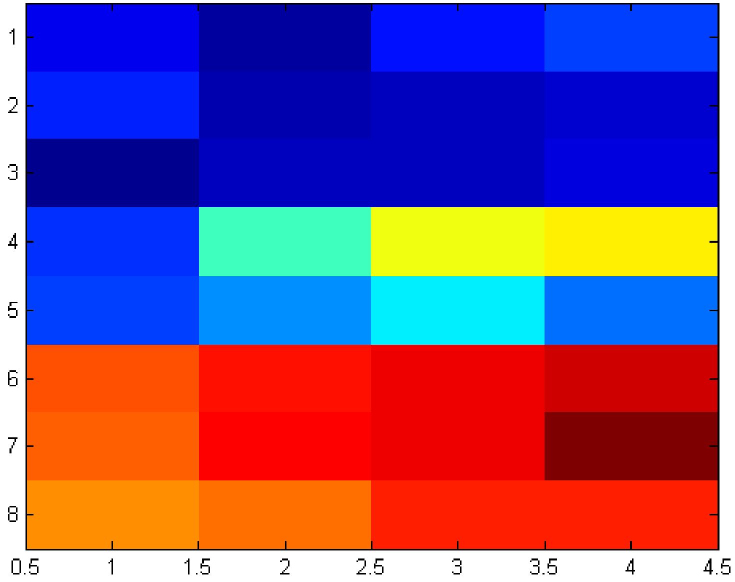


Top-left-black

Step 4 – White Pieces

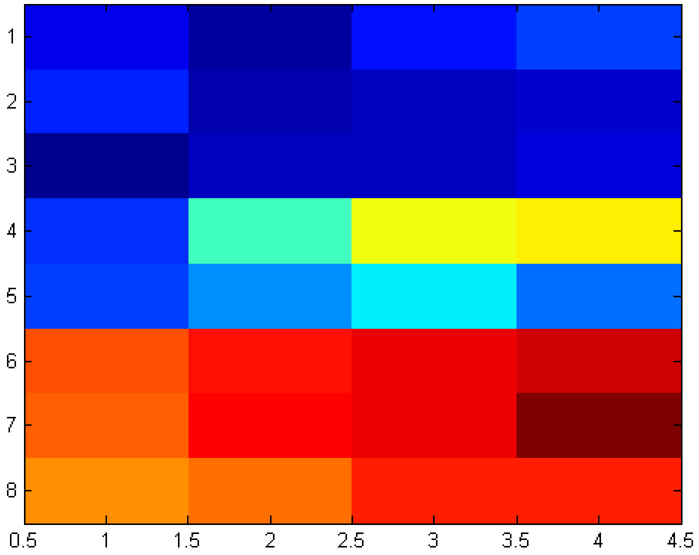
- Playable cells are black.
- A white piece is detectable via luminance.
- Noise & lighting artifacts: Some areas are darker than others.



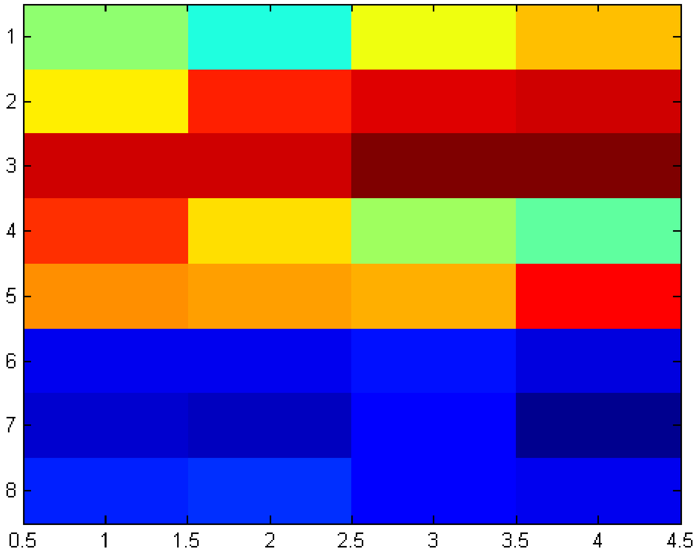


Step 4 – White Pieces

- Fighting the noise: subtracting the luminance of the adjacent cell:
- The value difference now it over 10 times better



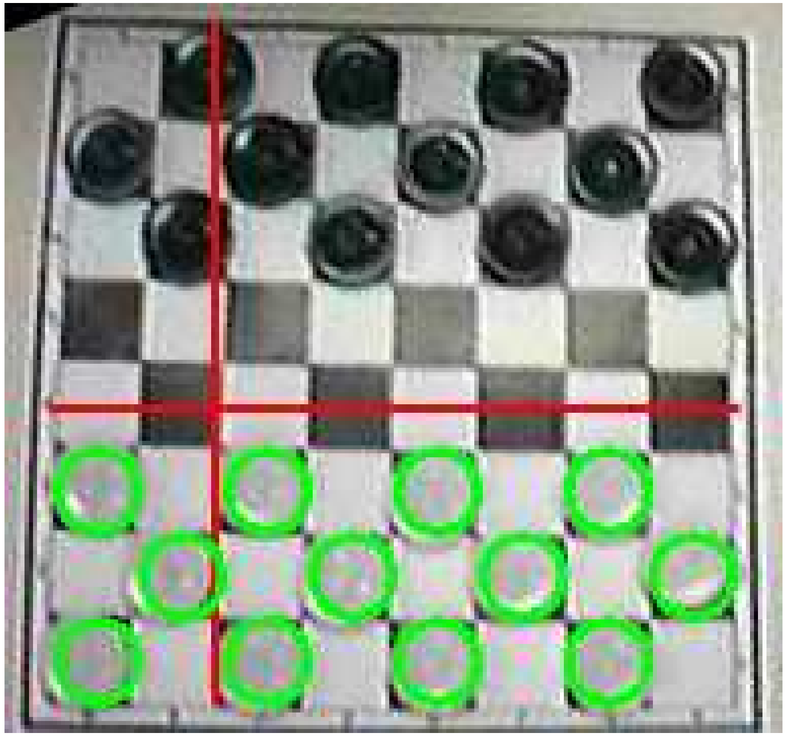
Before

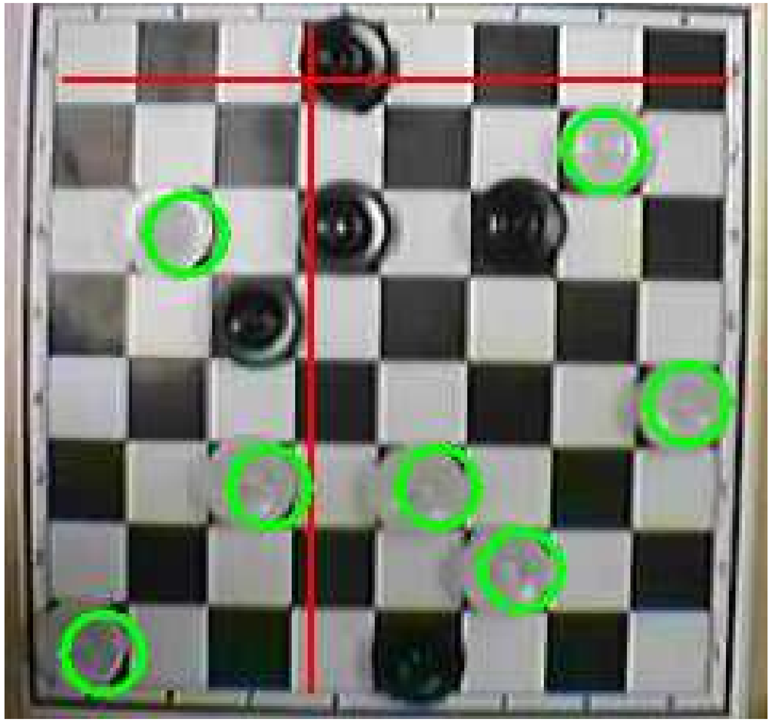


After

Step 4 – White Pieces

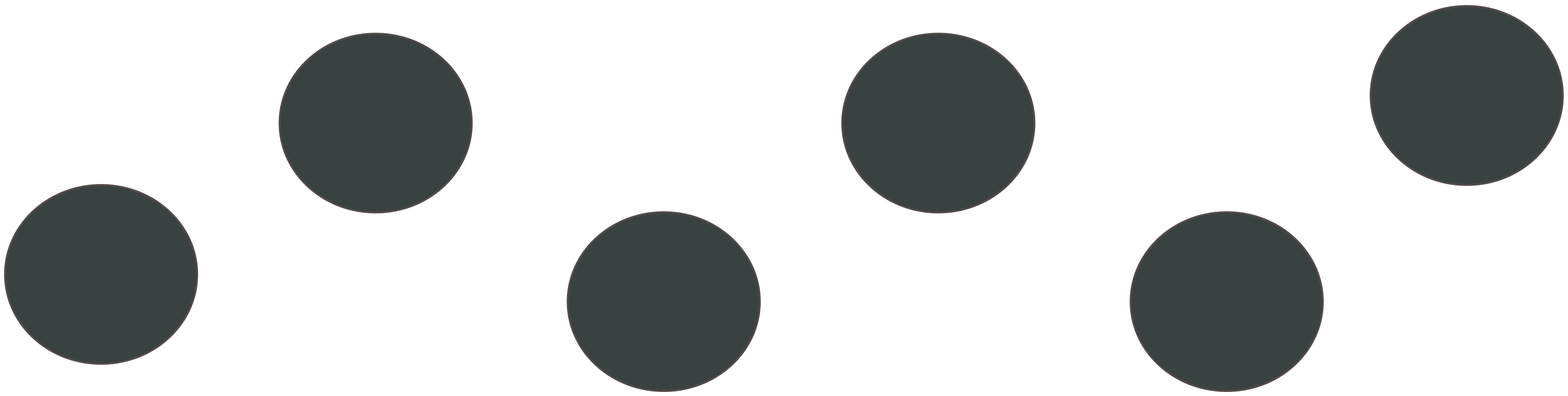
- Results:





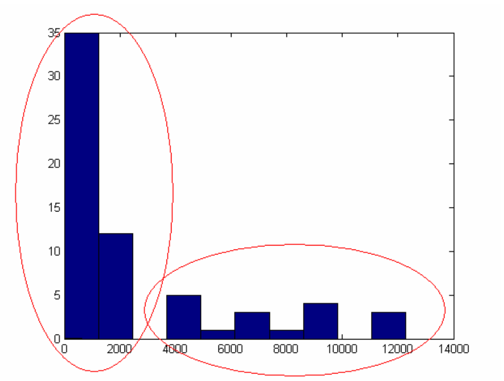
Step 5 – Black Pieces

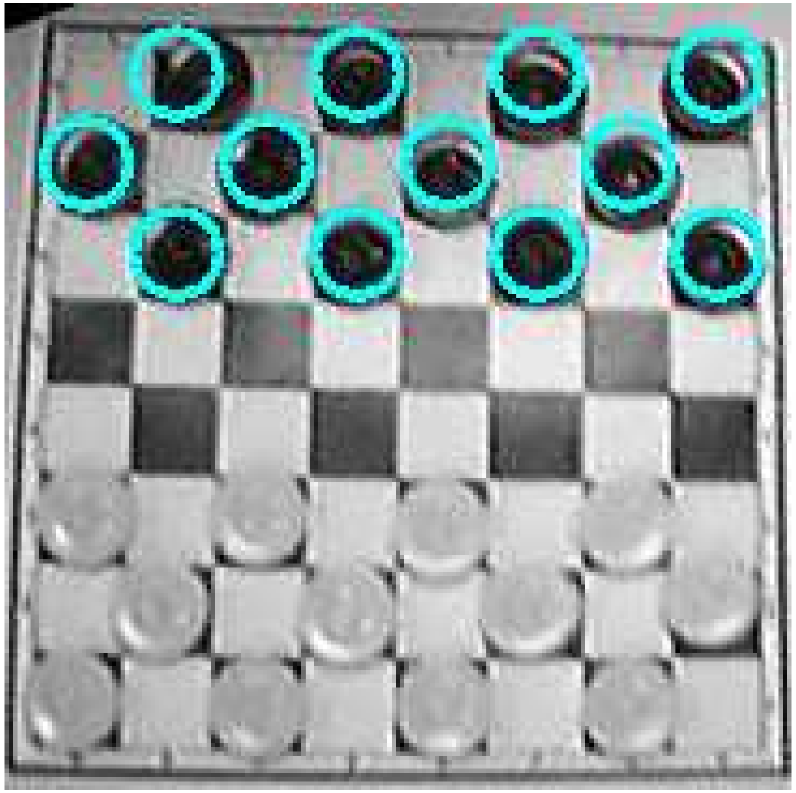
- Hough transform for circles.
- Low threshold - Lots of circles.
- Voting over cells instead of pixels.
- How many circles do we need to know that the cell contains a piece?

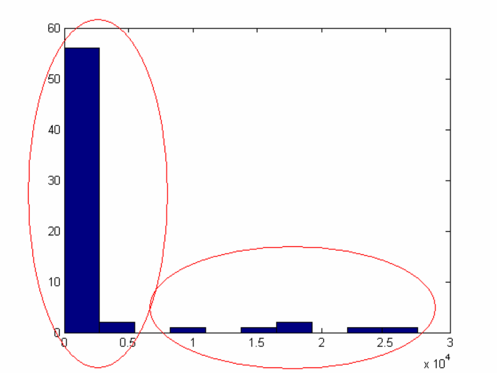


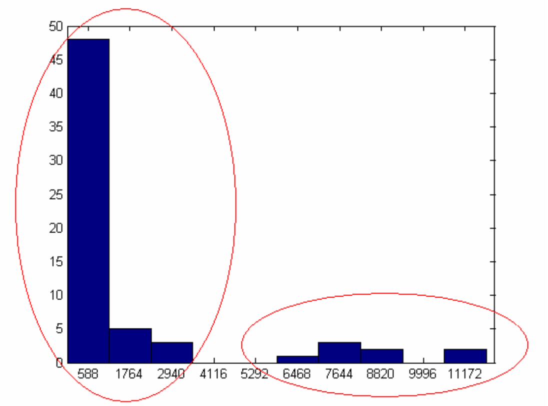
Step 5 – Black Pieces

- Clustering! The boundary is the threshold.



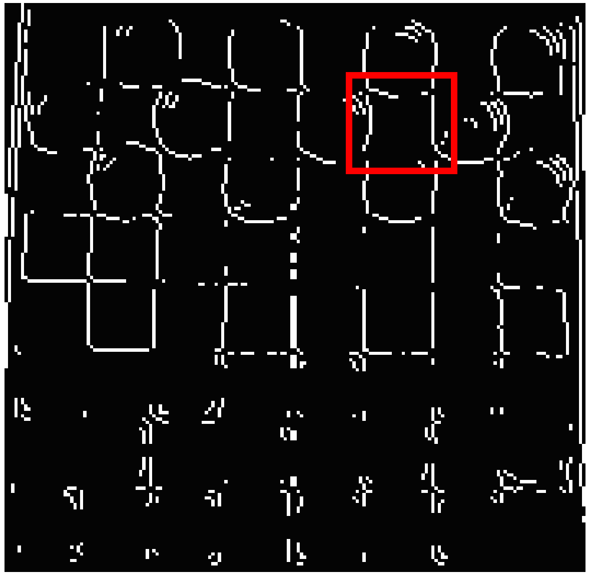


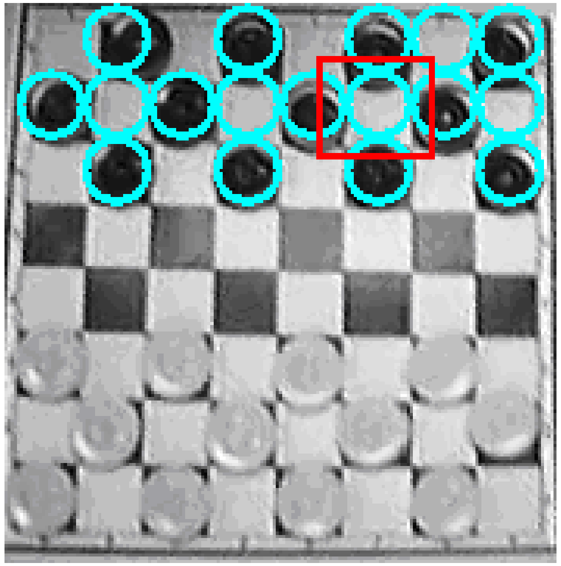




Step 5 – Black Pieces

- Problem: Noise.
- Solution: We did not use any domain knowledge – purge incorrect locations.
  - Pieces can only be found on playable cells.

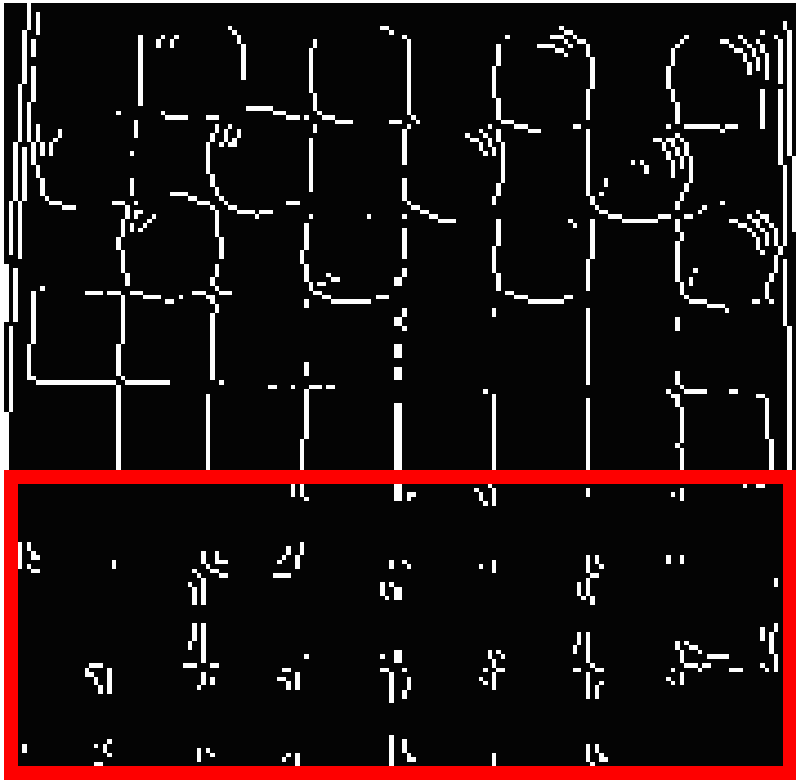




Step 5 – Black Pieces

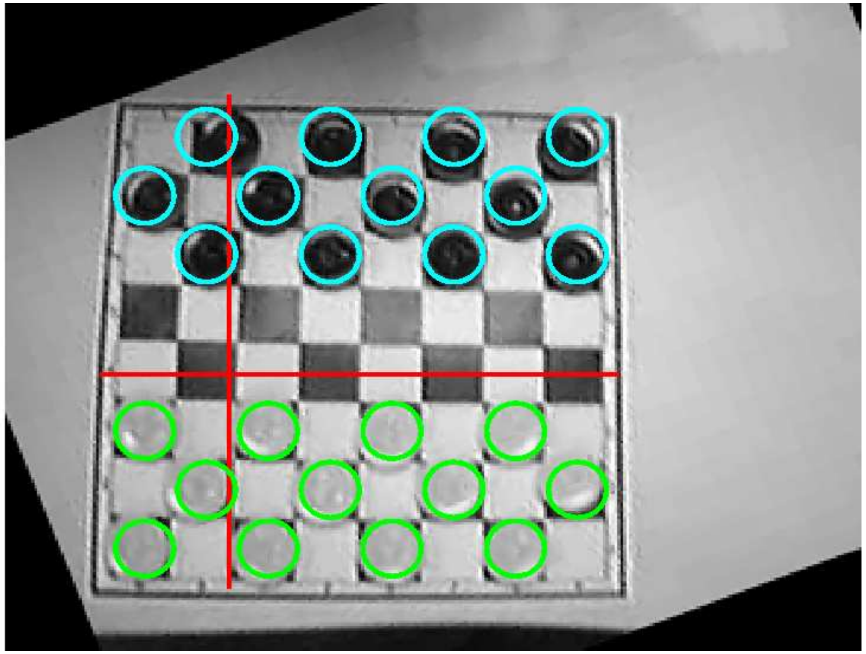
- Why didn’t the hough found any white pieces?

Because Prewitt et al. Did not find them either!



Step 6 – Results (the good)

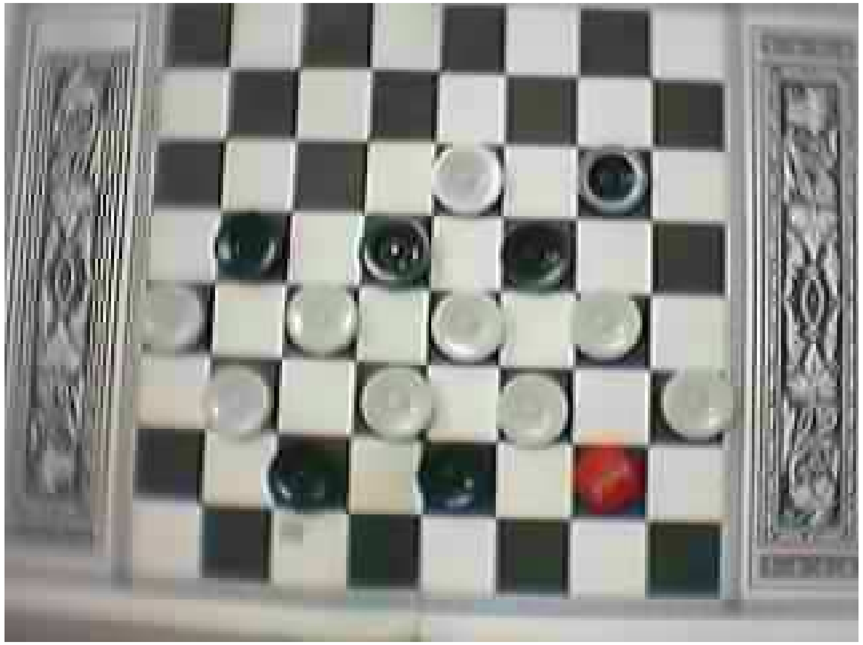
Before



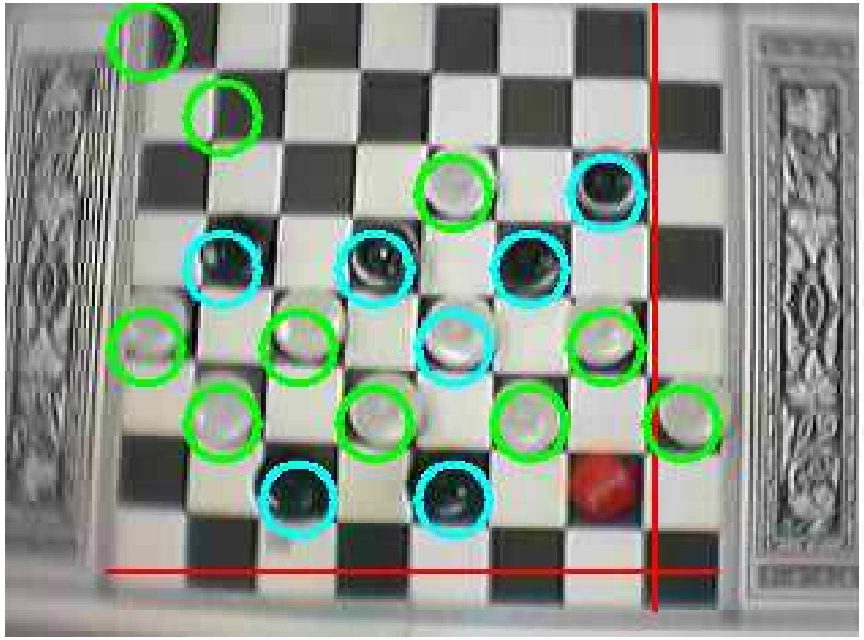
After

Step 6 – Results (the bad)

- Lower perspective – Trapezoidal boards.



Before

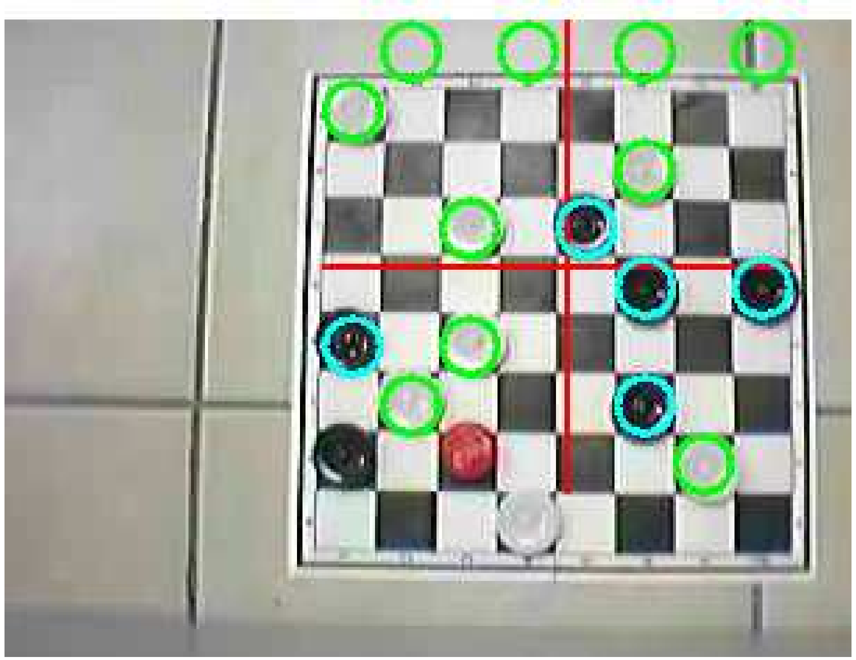


After

Step 6 – Results (the ugly)

- Some backgrounds may disturb.

Before



After

Conclusions

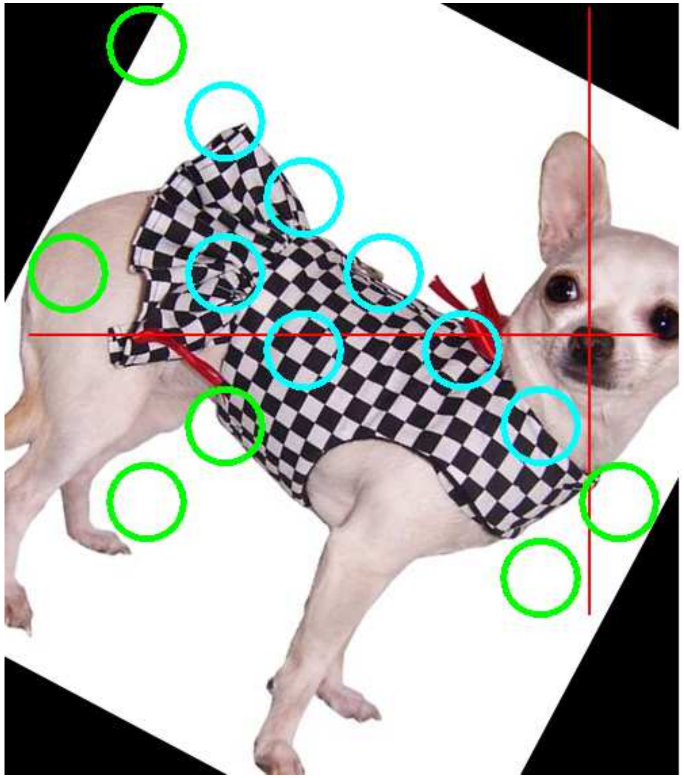
- Checkerboard detection is hard.
  - Restrictive environment constrains.
  - Time consuming (**30+ sec/image**) – not applicable in real time.
- Need a professional equipment.
  - A cheap webcam result:

Conclusions

- The system performance can be improved
  - □ Finding the board using a complete board lock (matching more than one column)
  - □ Using shading.
  - □ Image processing to reduce noise.
  - □ Colors instead of grayscale.
  - □ Two cameras.

Conclusions

- The computer can find what it wants in any picture:



Thanks for your attention