CHESS VISION

Amanpreet Walia
Xue Ying Shi
Youn Sun (Joy) Choi

PRESENTATION AGENDA

- Board Projection
- Board Detection
- Hand Detection
- Piece Detection
- Move Determination
- Chess Board mapping

PROJECT OBJECTIVES

Observe

- Observe a chess game played in the real world

Detect

- Detect the board and each move played in real time

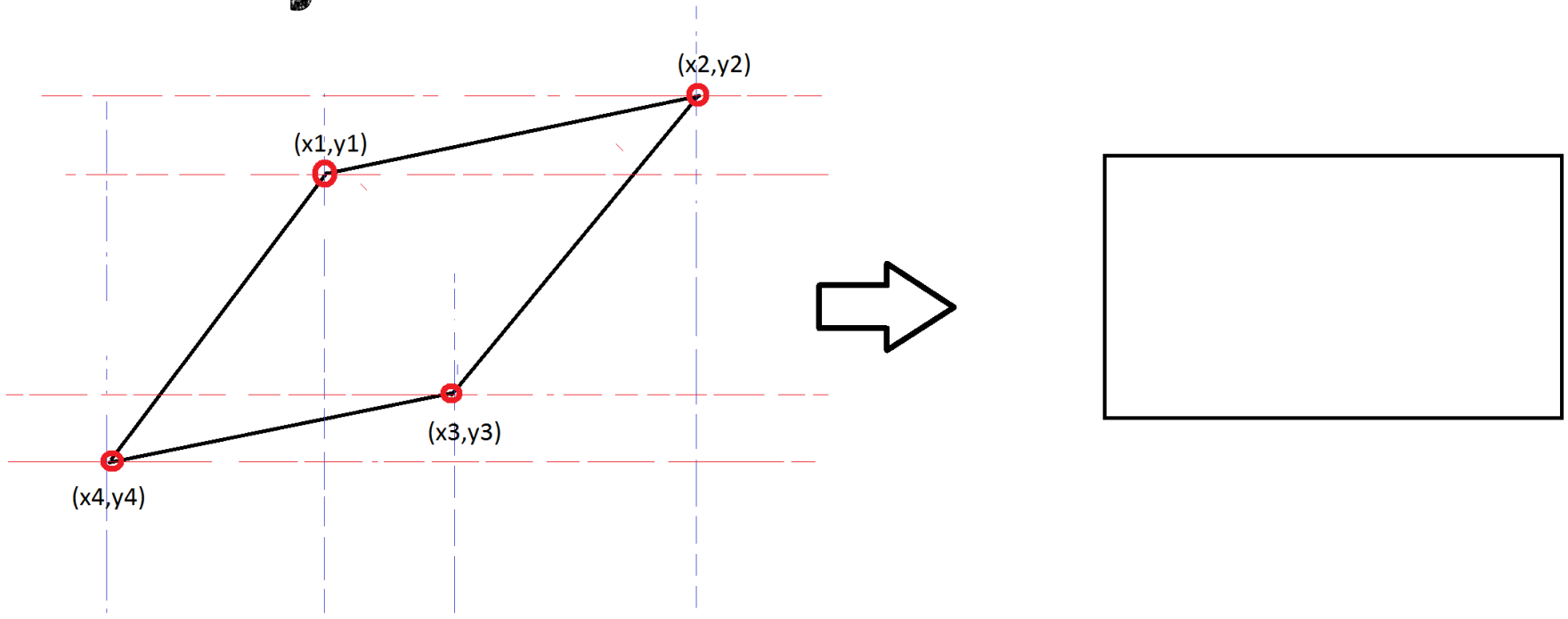
Analyze

- Depict movements from board in the world on a virtual board in the program

BOARD PROJECTION

- Click the 4 corners of the chess board on a static initial image
- The corner points are then processed to get transformation matrix which is applied on board.

BOARD PROJECTION

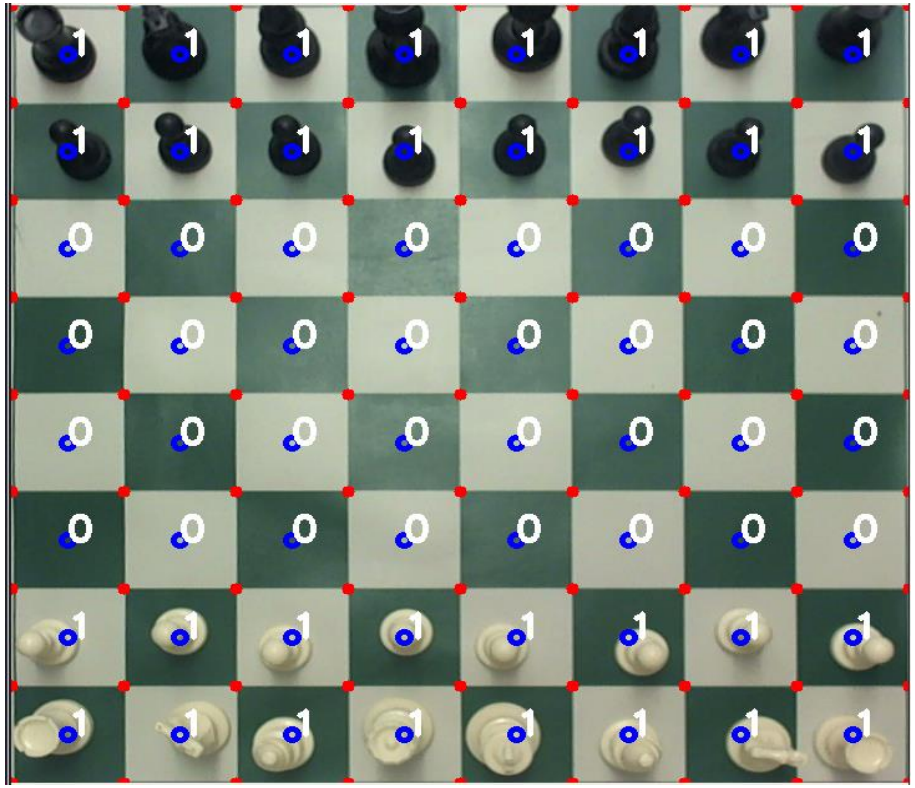


$$\text{width} = \max\left(\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}, \sqrt{(x_3 - x_4)^2 + (y_3 - y_4)^2}\right)$$

$$\text{height} = \max\left(\sqrt{(x_1 - x_3)^2 + (y_1 - y_3)^2}, \sqrt{(x_2 - x_4)^2 + (y_2 - y_4)^2}\right)$$

BOARD DETECTION

- Once we have the area within the four corners defined and projected onto an orthogonal board, it is assumed that the board will remain *stationary* – everything outside of the board area is ignored.
- The width and height of the resulting projection is divided by 9 to get the dimension of each grid square. Each square is then represented as an object of class ChessSquare in our program and contains the following information:
  - Centroid coordinate (blue circle)
  - Four corner coordinates (red dots)
  - Its position in the grid in Cartesian coordinates
  - Information regarding its image features

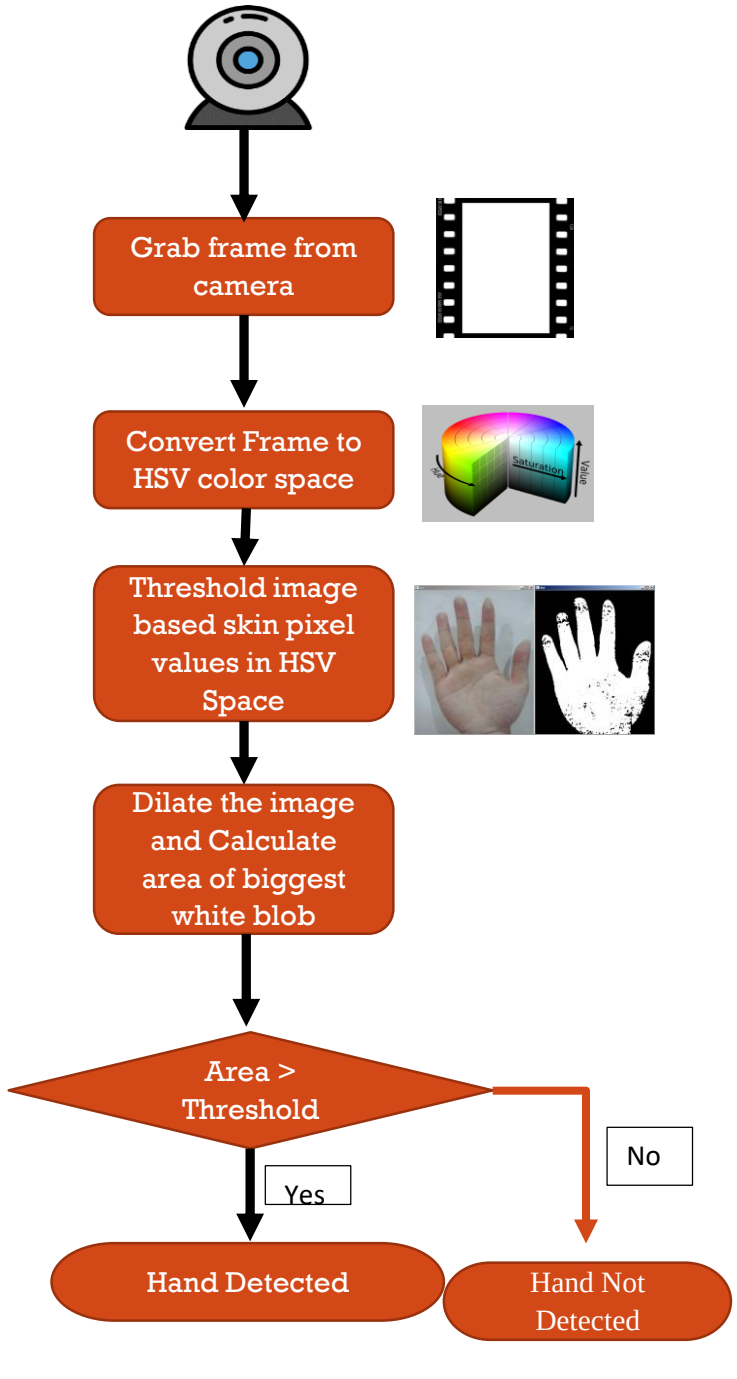


HAND DETECTION

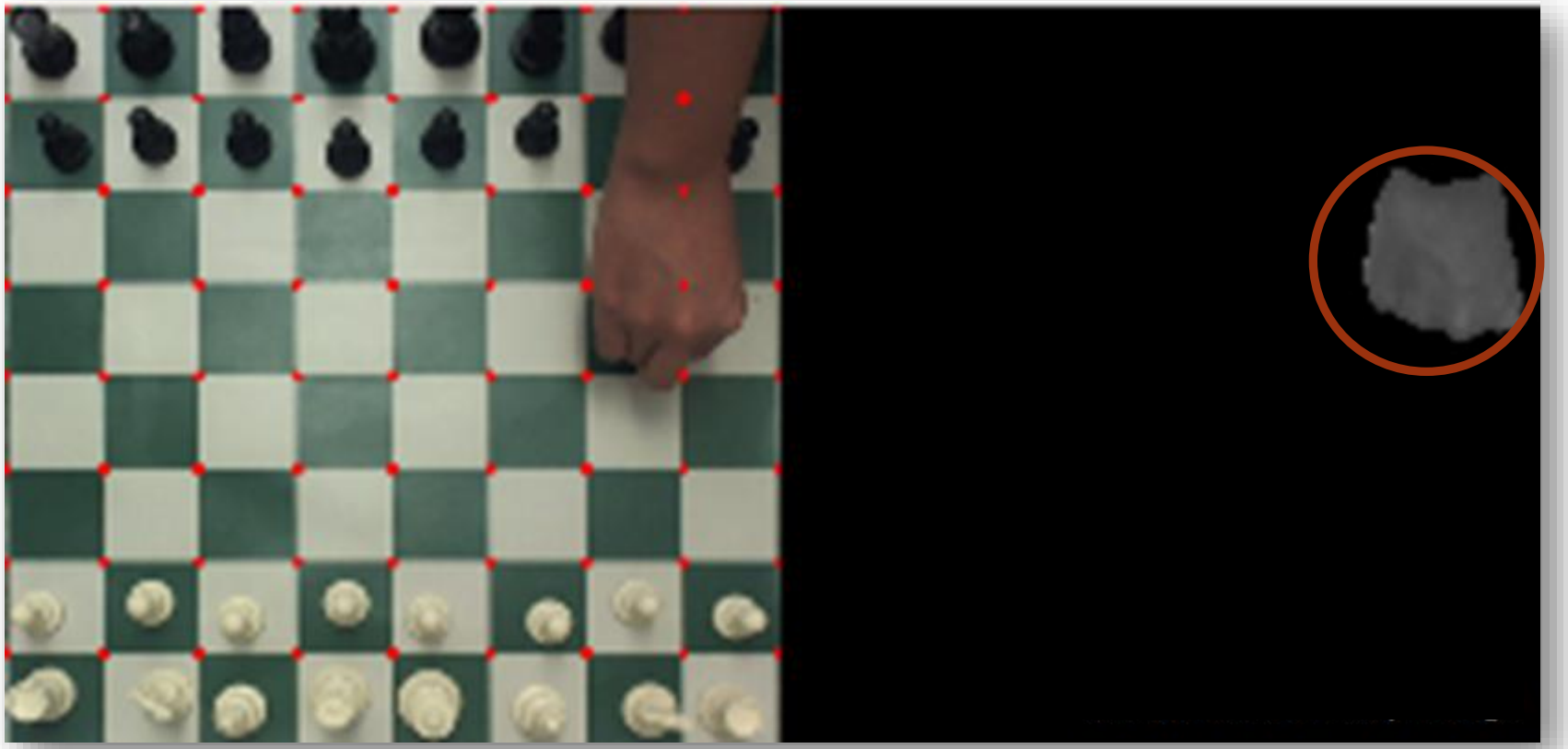
The Challenge: Determine if a hand is in the image – do not try to process chess piece movement if a player is in the process of moving a piece.

Define a range of skin tones ((H,S,V)=>[0, 48, 80]to[20, 255, 255]) to be detected on an HSV scale.

1. Convert the relevant image points to HSV values and determine if within range. Create a binary image based on which points are inside the range and which points are not.
2. Dilate the image.
3. Find the largest contour and area of 1s.
4. If the area exceeds a set threshold, a hand is detected to be in the image



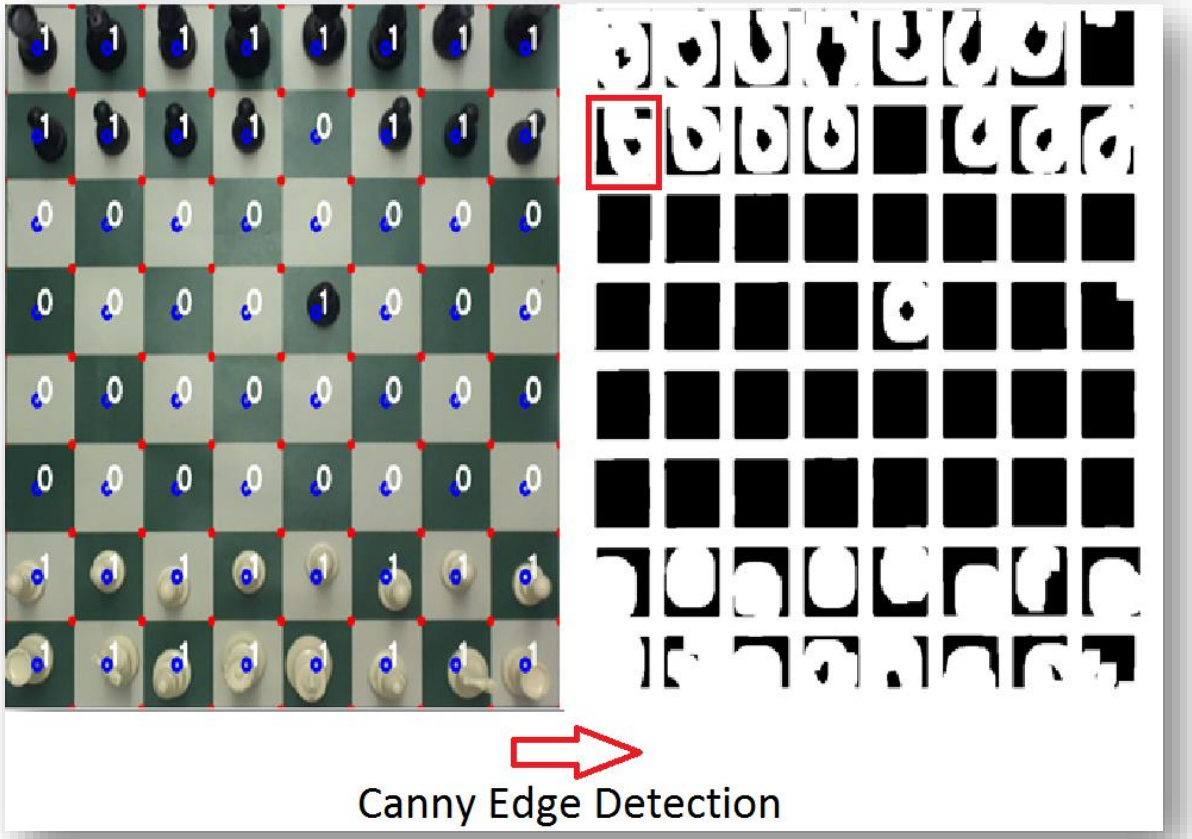
HAND DETECTION



A hand is detected in this image, so no further processing is done

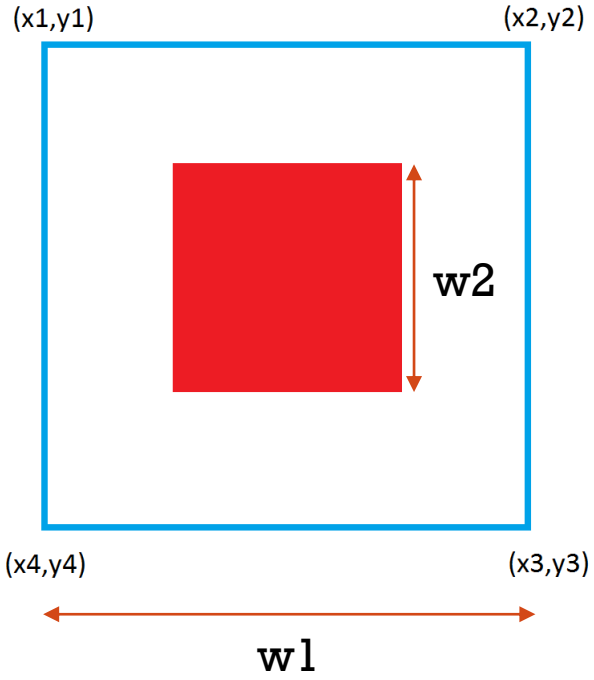
PIECE DETECTION(I)

- **The Next Step**: Focus on the content of each square to determine whether or not it contains a piece, and if it does, whether it is white or black.



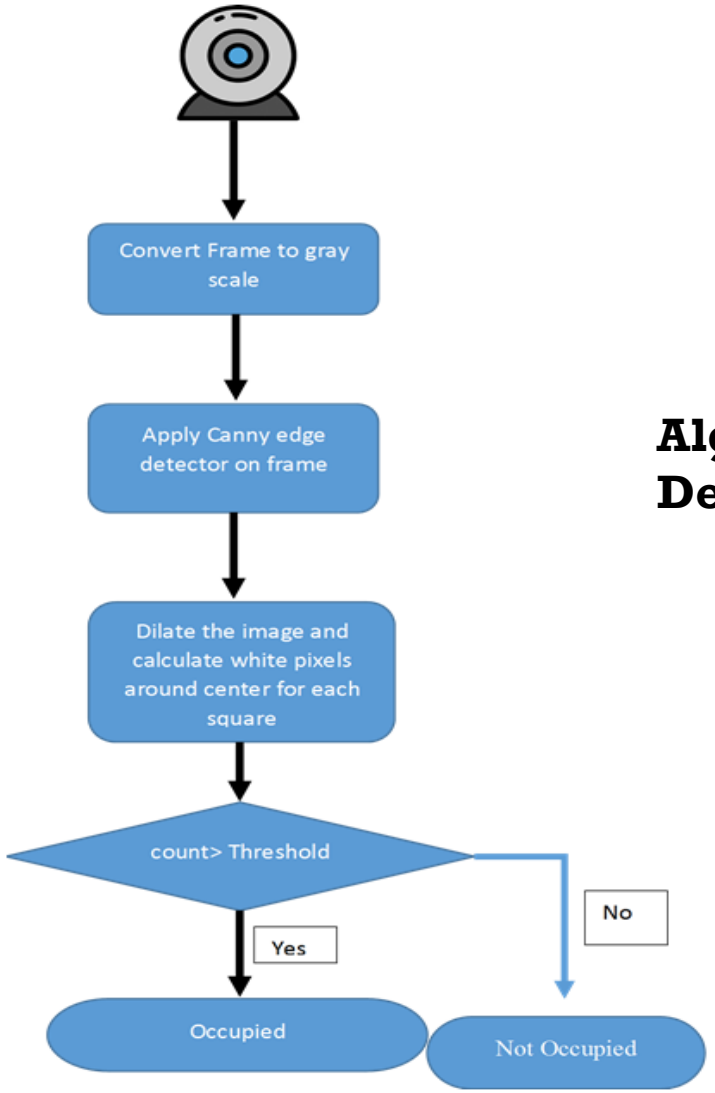
1. Canny Edge Detection is applied to a greyscale image of the current frame.
2. Apply dilation.
3. Count white pixels within a set area (green box) around the centroid (blue circle).
4. If the count passes set threshold, there is a piece.

PIECE DETECTION(I)



Individual Chess square

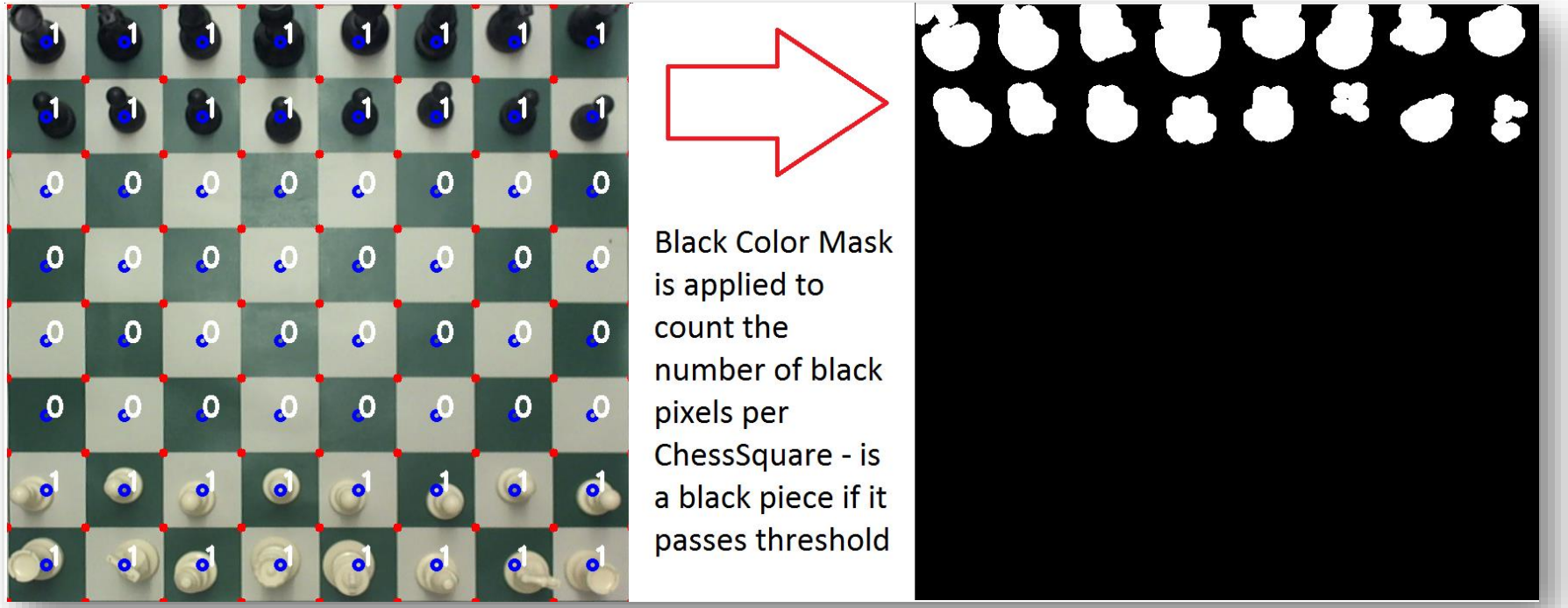
w2 is calculated by using % threshold of w1



Algorithm for Piece Detection

PIECE DETECTION(II)

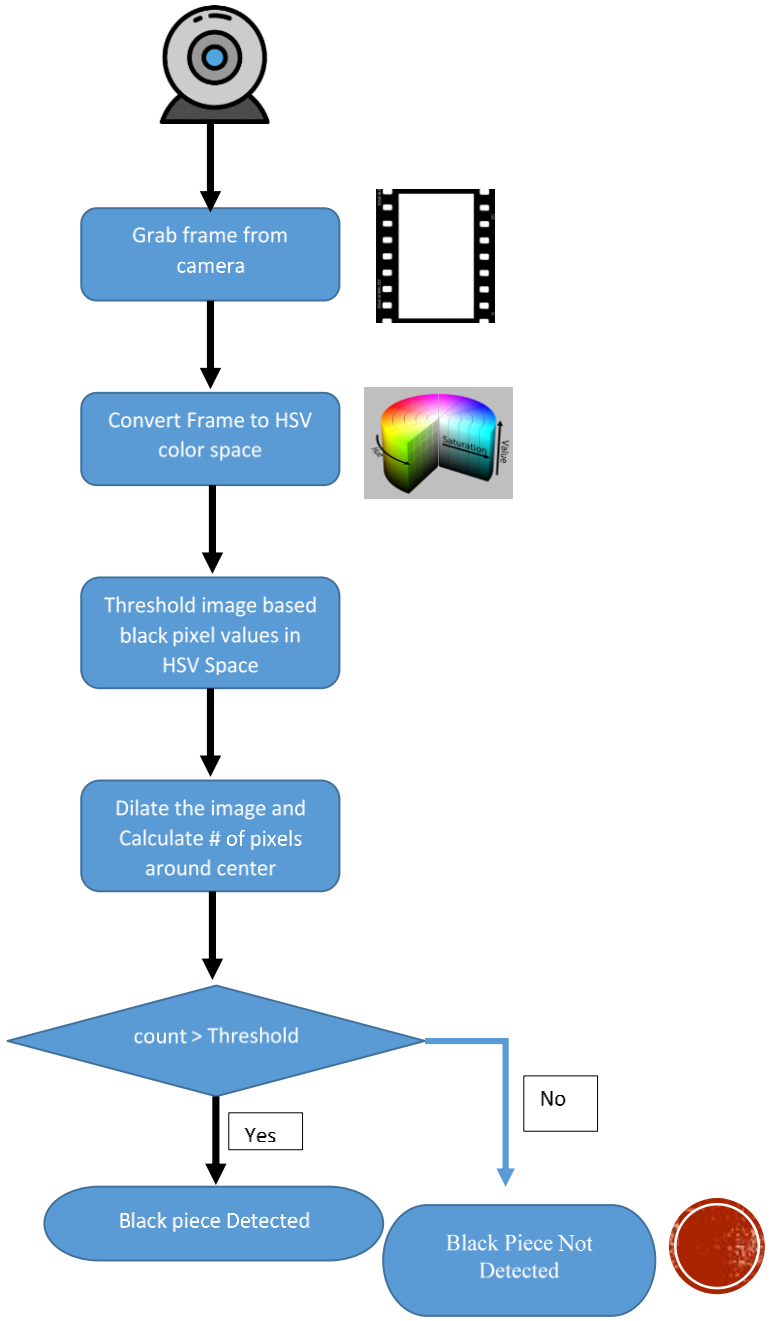
- **Canny Edge Detector FAILS for black pieces very often due to lack of reflection features which helps in edge detection.**
- **We need something else to see where are all the white pieces .**
- **Color Detection based on black color worked with fairly good accuracy.**



PIECE DETECTION(II)

Define a range of black color{(H,S,V)=[0, 0, 0]to [180, 255, 30]} to be detected on an HSV scale.

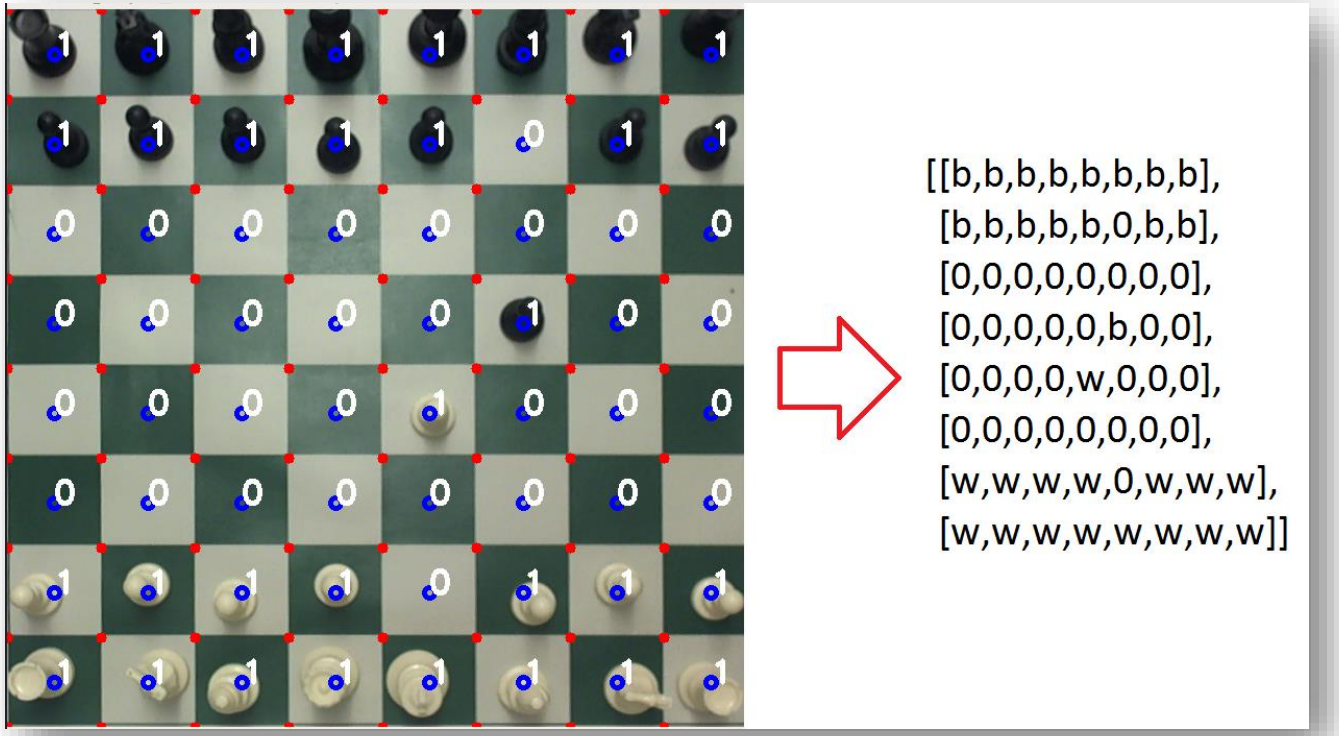
1. Convert the relevant image points to HSV values and determine if within range. Create a binary image based on which points are inside the range and which points are not.
2. Dilate the image.
3. Calculate the number of pixels around center using same technique as before.
4. If the count exceeds a set threshold, a black is detected to be in the image.



COMBINED PIECE DETECTION

MOVE DETERMINATION

- Information about what piece is in each square is passed to the move determination module in the following format: an 8x8 array containing ‘0’ if there is no piece, ‘w’ if it contains a white piece, and ‘b’ if it contains a black piece
- Each 8x8 array is stored and compared to previous frame arrays, and if there is any change in the array between the states, a move was made



CHESS BOARD MAPPING

- Once it is determined that there was chess piece movement, the following logic is applied to determine what move was made:
  - If a square in the previous state contained a piece, but current state does not: the piece in the square from the previous state was moved
  - The destination is either: a square in the current state that contains a piece, but did not contain a piece in the previous state; or a square that contains a piece of a different color in the current state than the previous state.
- With the start and destination squares in hand, the board state is updated and a virtual representation of the board is rendered to the console for each movement.