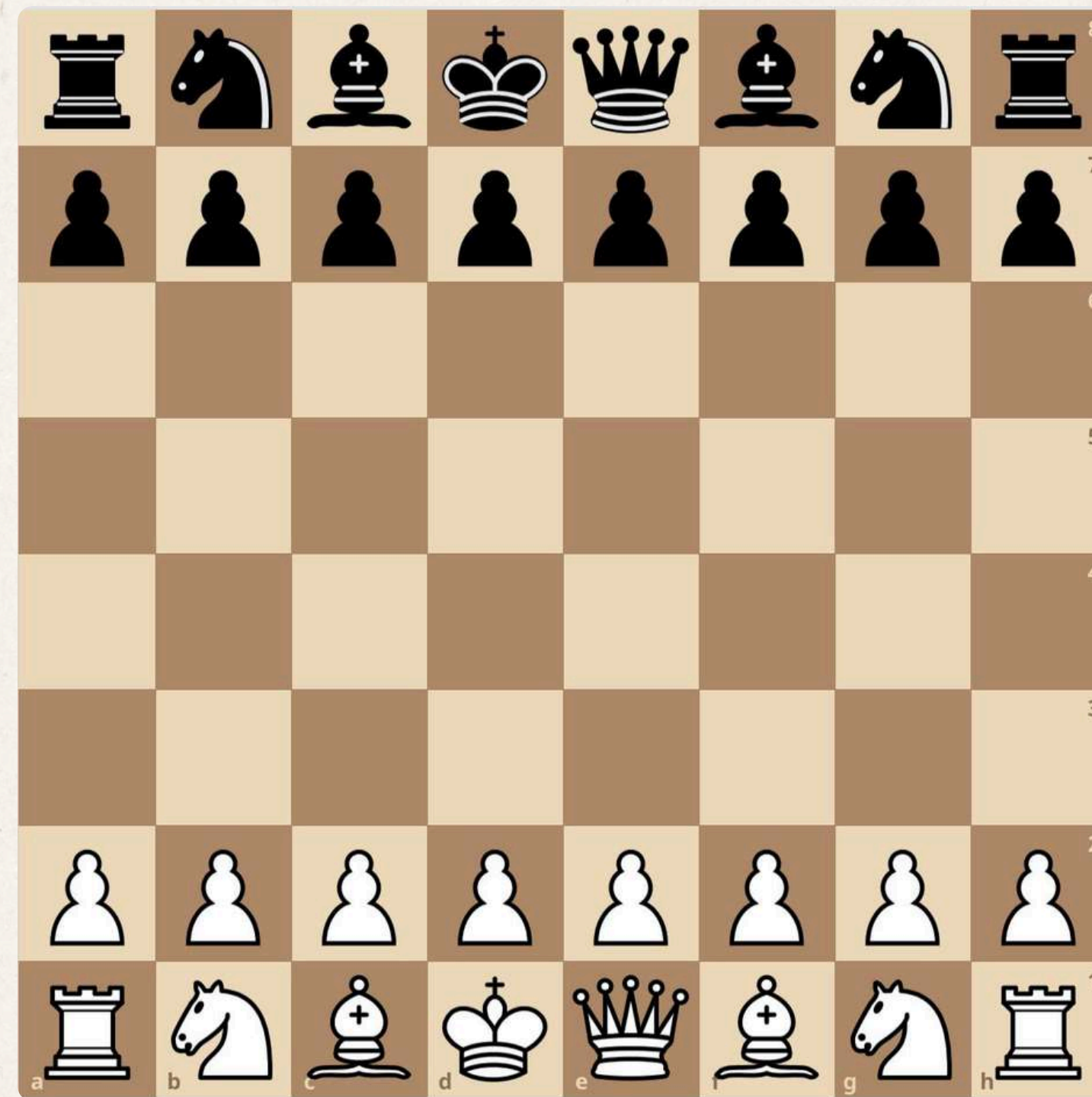


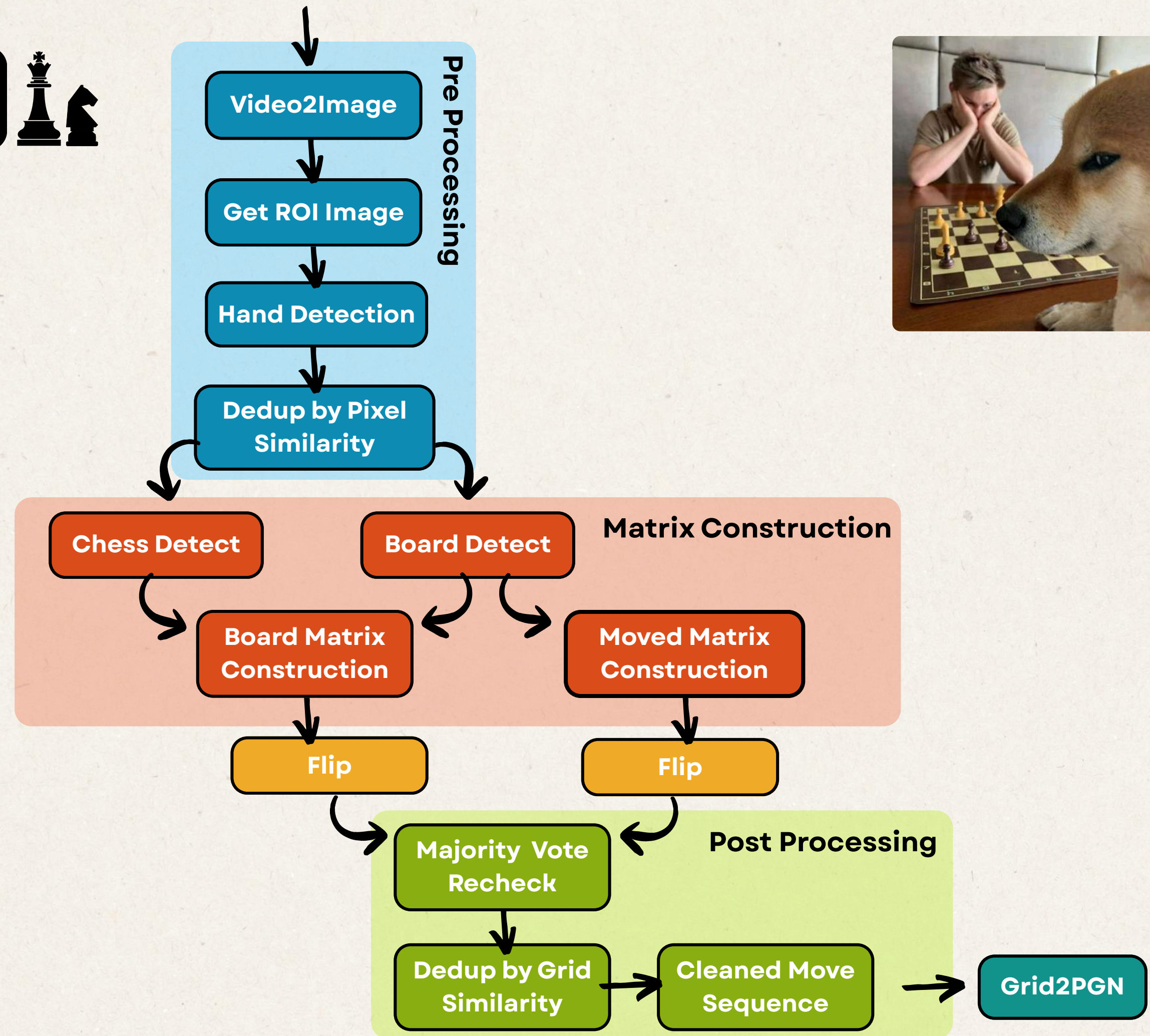


CHESS DETECTION COMPETITION

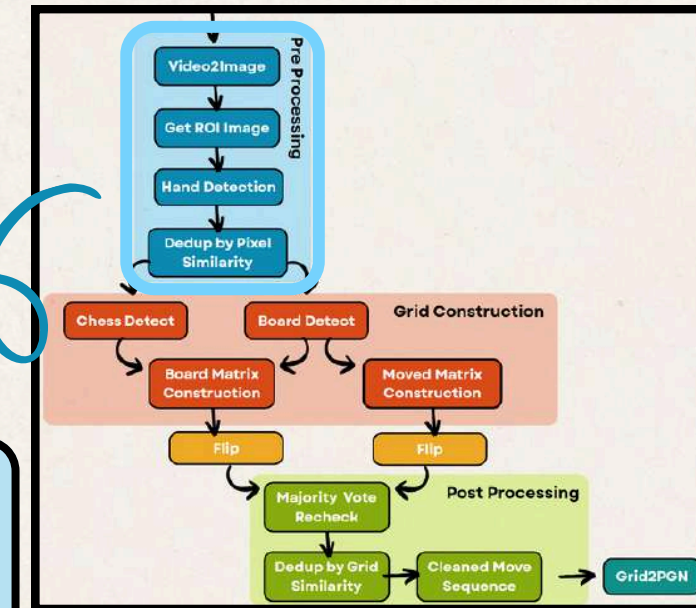
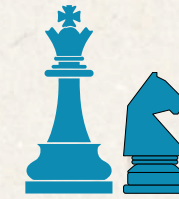
TEAM: JUJU อย่าลืมทำงาน



Pipeline : Overall



Pipeline : Pre - Processing



Video



Image



Video2Image



board_image

Get ROI Image



chess_image

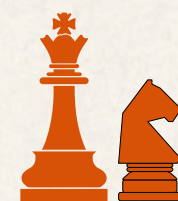
Hand Detection



Dedup by Pixel Similarity



Pipeline : Matrix Construction



chess_image



Chess Detect



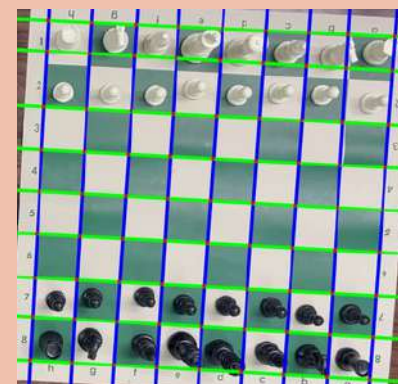
Board Matrix
Construction

R	.	B	.	Q	.	.	R
P	P	K	.	.	P	.	P
.	N	.	.
.	.	.	P	.	.	P	.
.	.	.	p	P	.	.	n
.	.	.	.	p	.	.	.
.	p	.	.	b	p	p	p
N	n	b	k	q	.	.	r

board_image



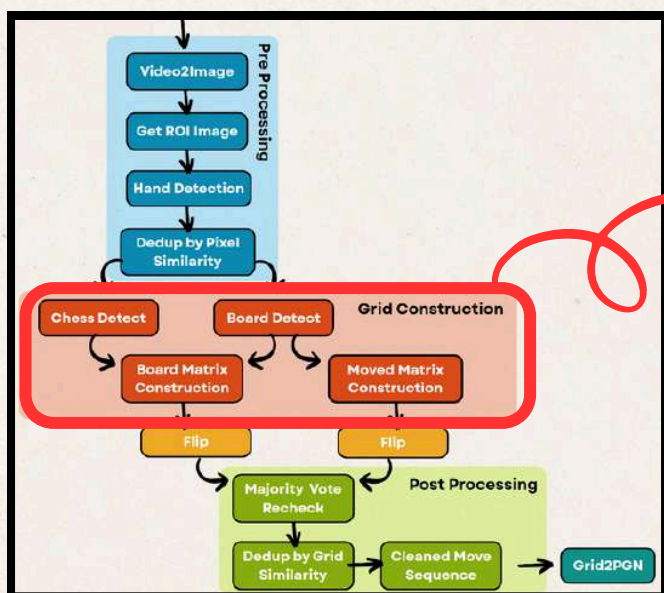
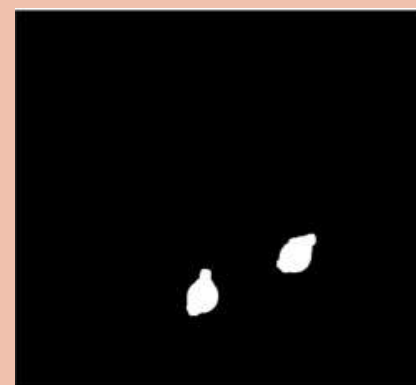
Board Detect



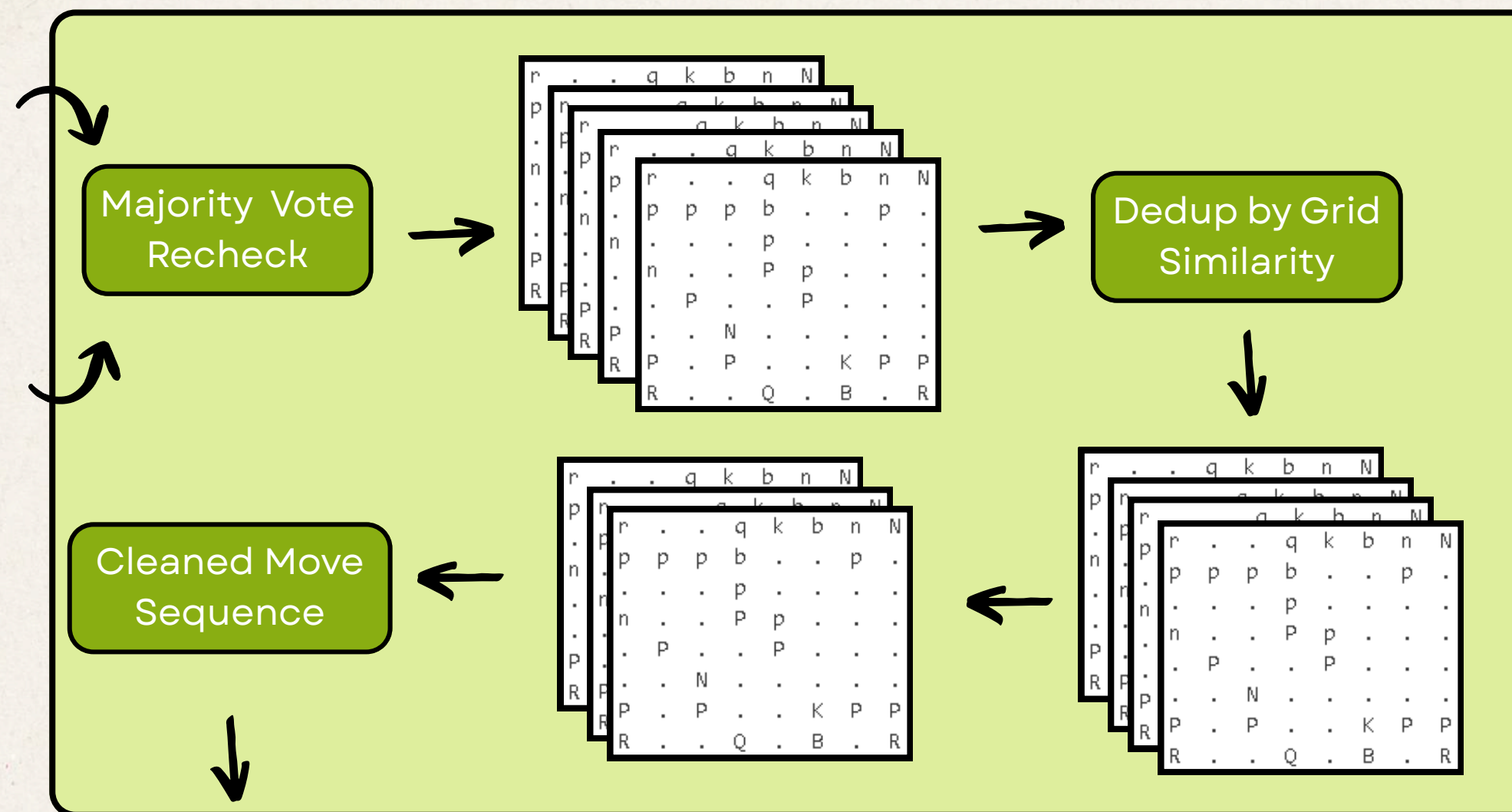
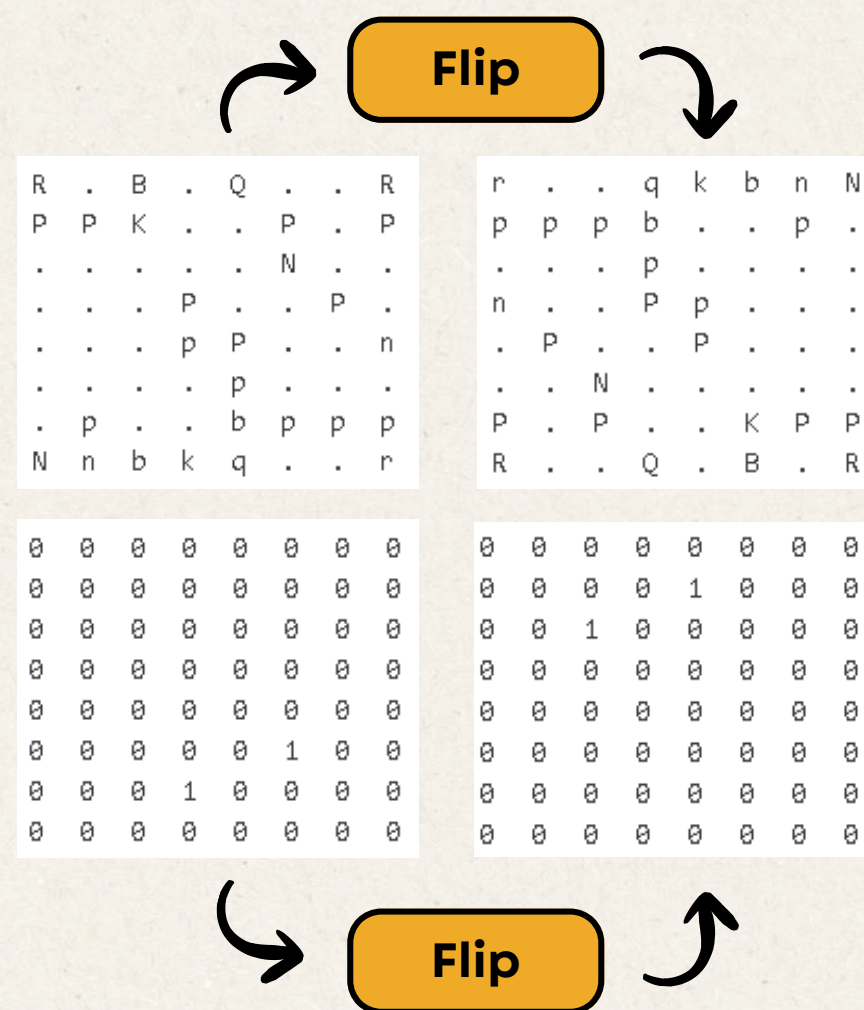
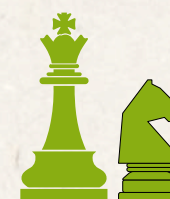
Moved Matrix
Construction

0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	1	0	0
0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	0

Dedup by Pixel
Similarity

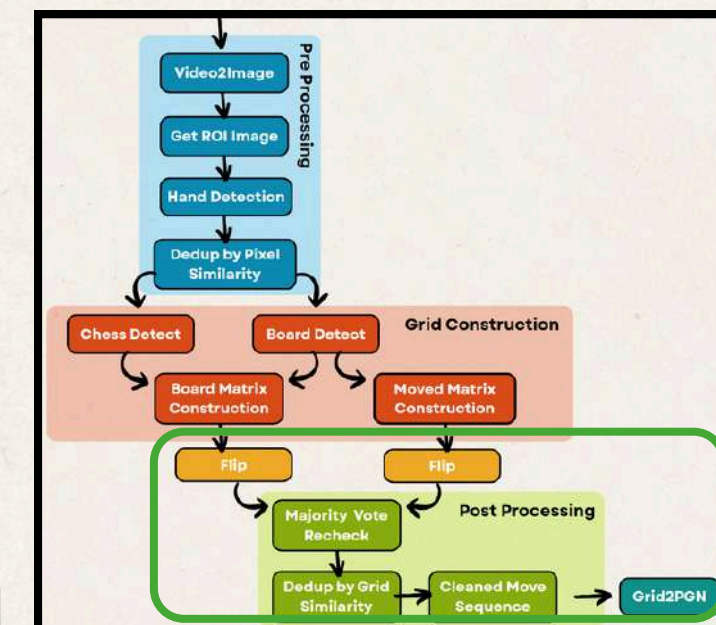


Pipeline : Post - Processing



Grid2PGN

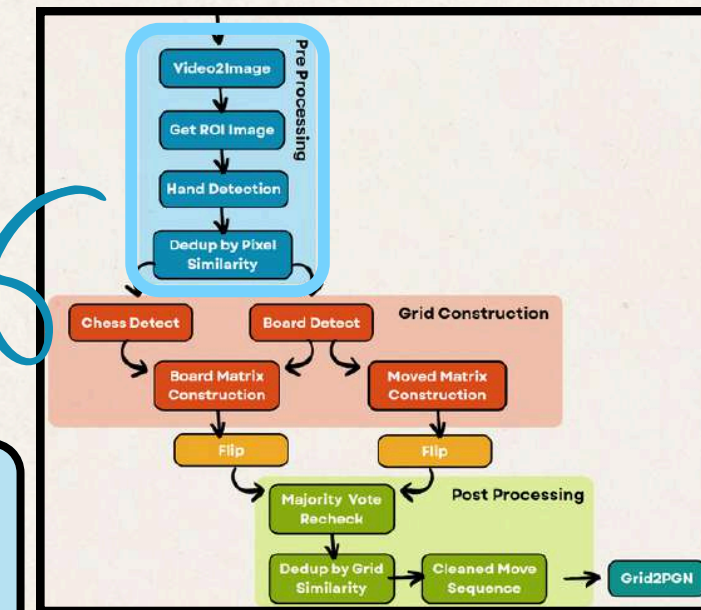
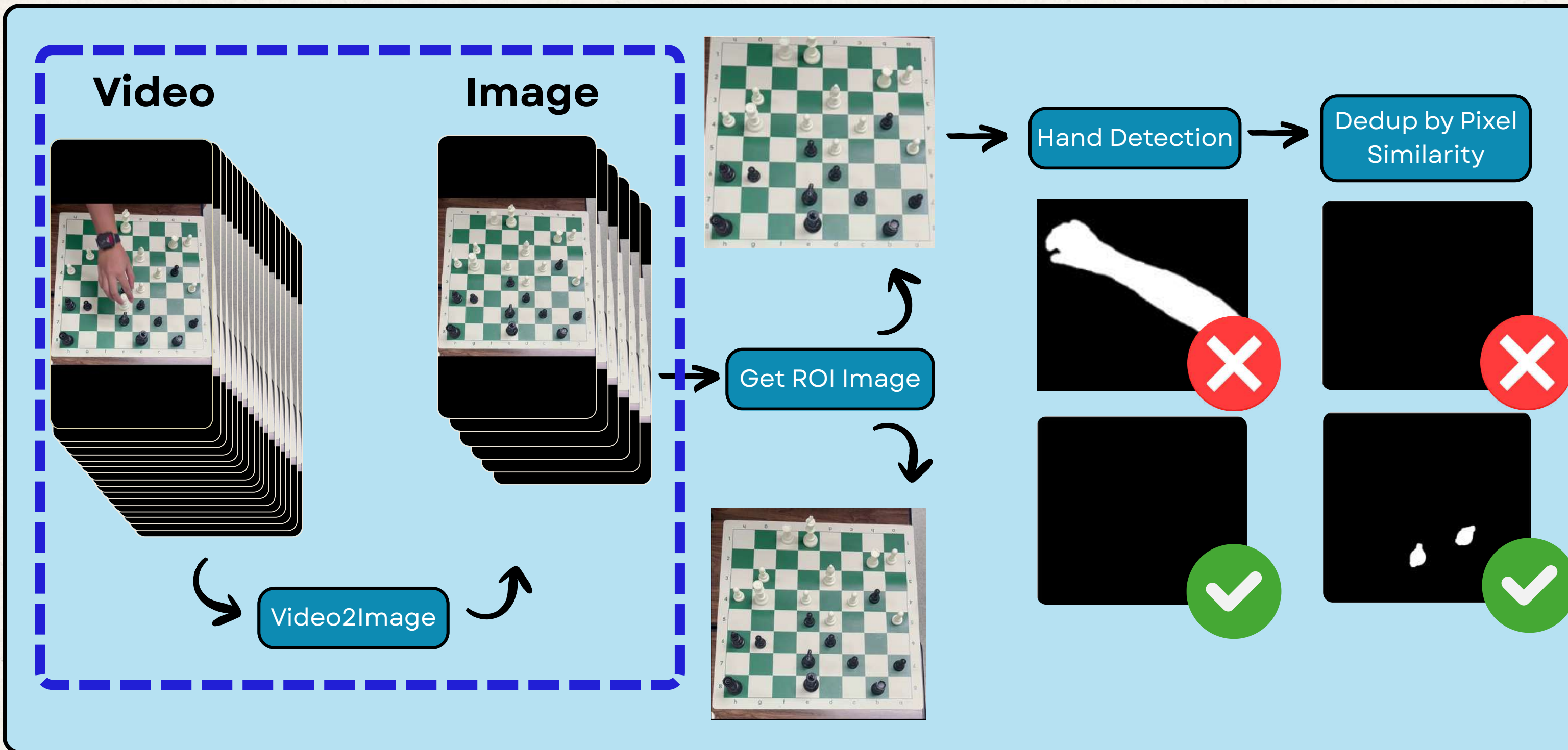
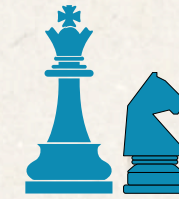
1... Qh4+ 2. g3



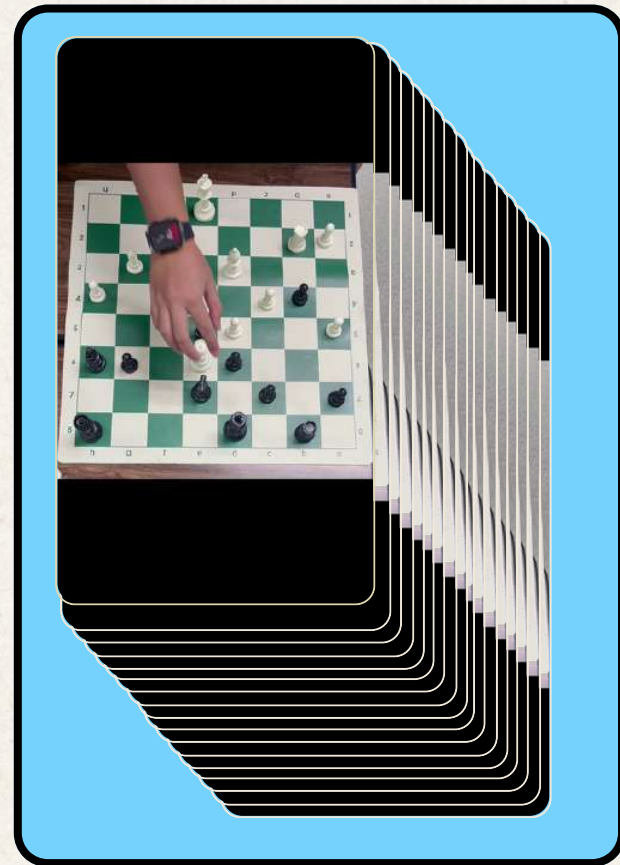
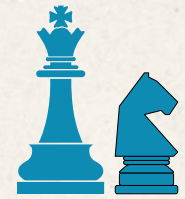


Pre - Processing

Pipeline : Pre - Processing



Video2Image



Video

Frame 15 clean	Frame 30 clean	Frame 45 block	Frame 60 block	Frame 75 block	Frame 90 clean
-------------------	-------------------	-------------------	-------------------	-------------------	-------------------

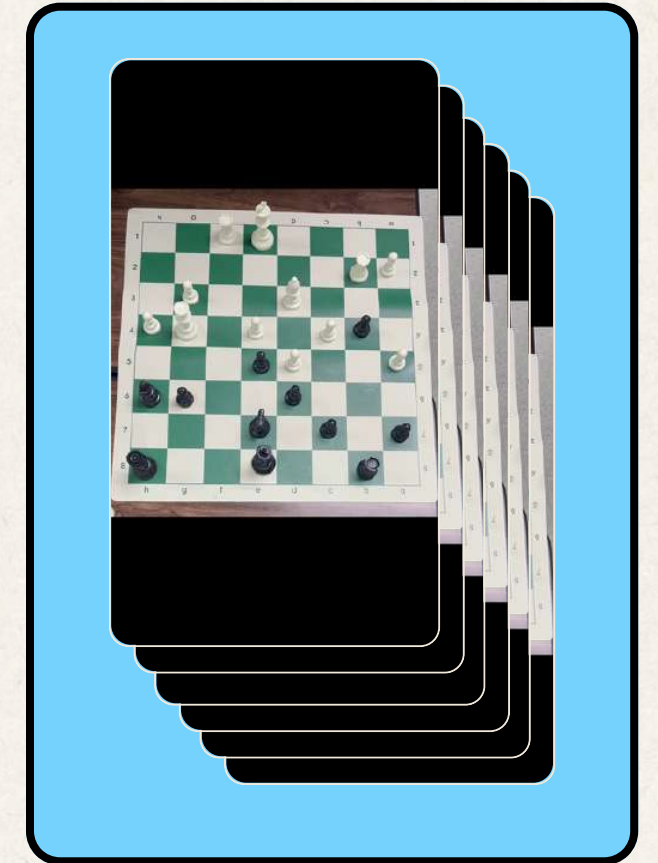
save

save

By YOLOv8 nano

Found Hand

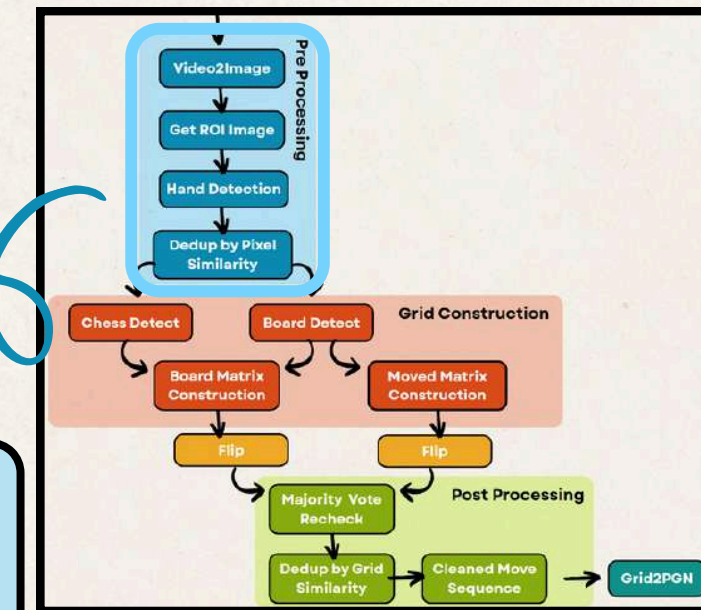
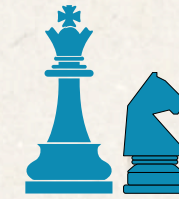
Not Found Hand



Image



Pipeline : Pre - Processing



Video

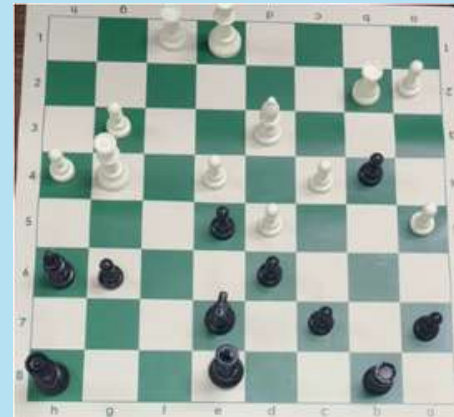


Video2Image

Image



Get ROI Image



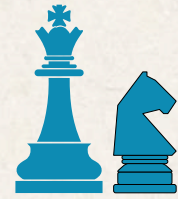
Hand Detection



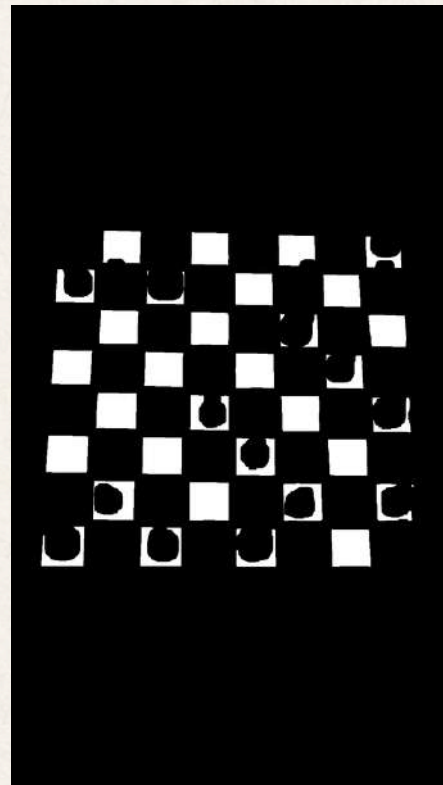
Dedup by Pixel Similarity



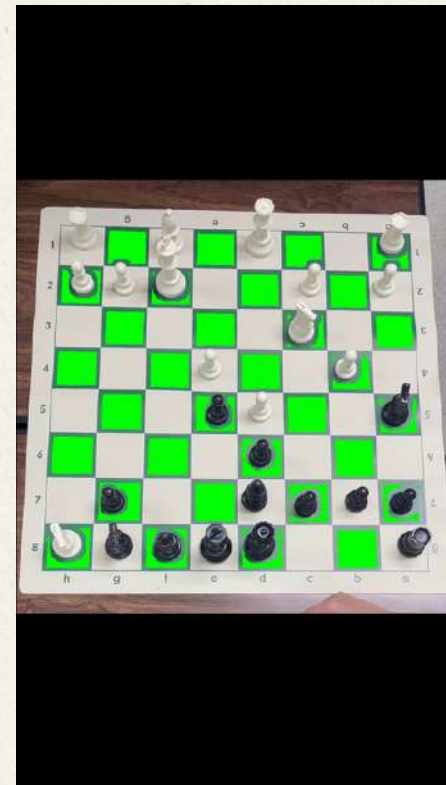
ROI Image



Original Frame



Green Mask
Opening



Mask Overlay

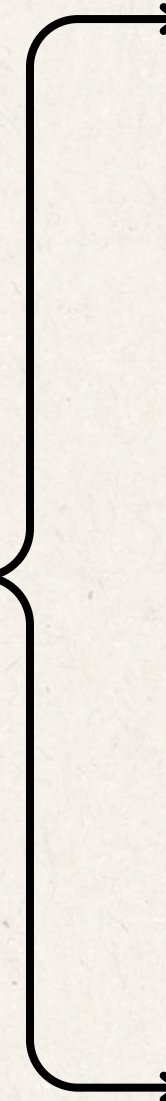


Frame with Box



Cropped Image

+ Grid padding



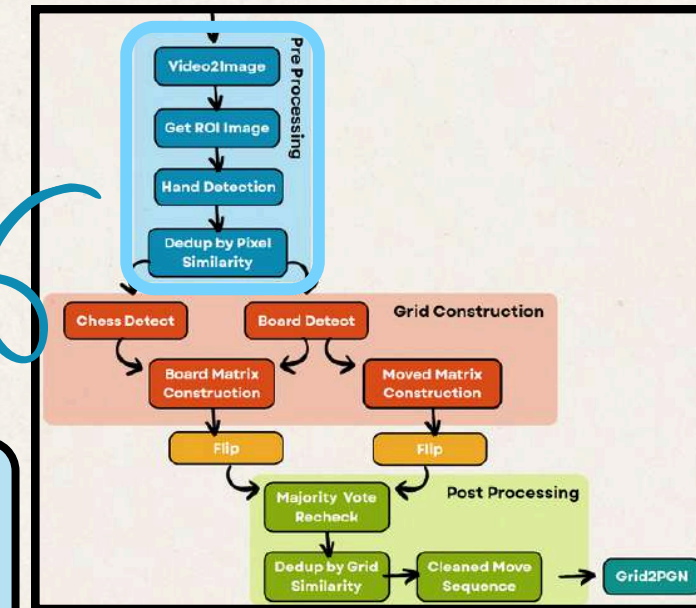
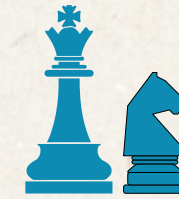
Board Image

+ Image padding



Chess Image

Pipeline : Pre - Processing



Video

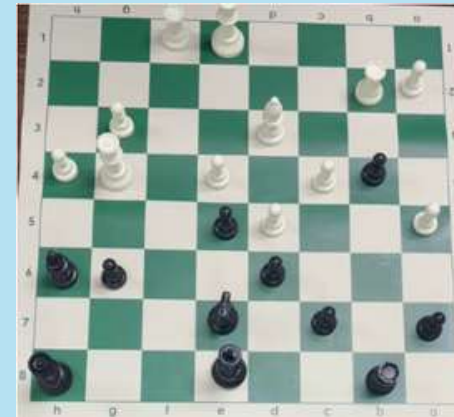


Image

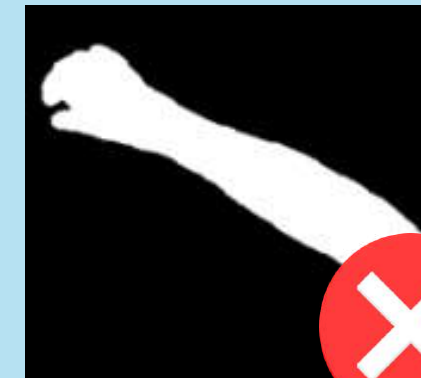


Video2Image

Get ROI Image



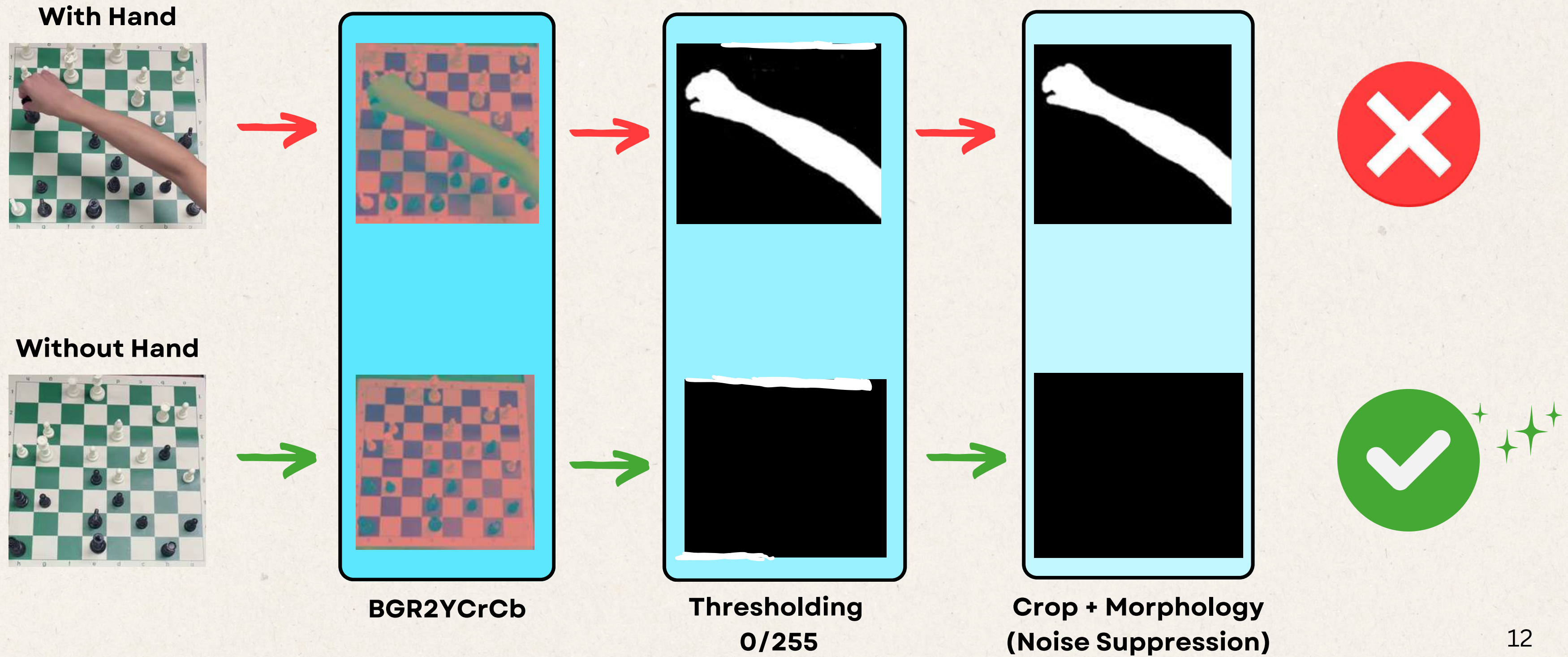
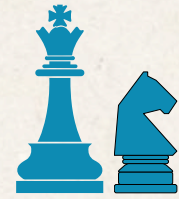
Hand Detection



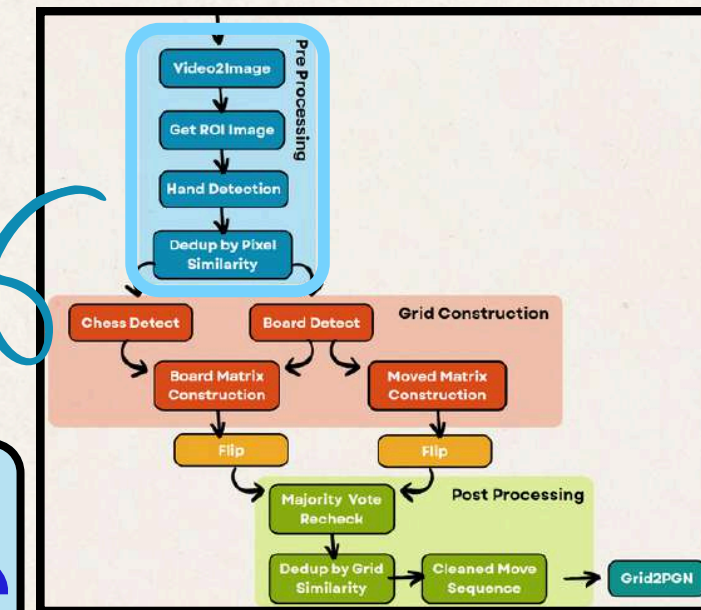
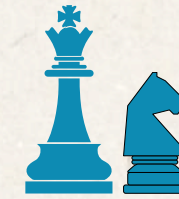
Dedup by Pixel Similarity



Hand Detection



Pipeline : Pre - Processing



Video

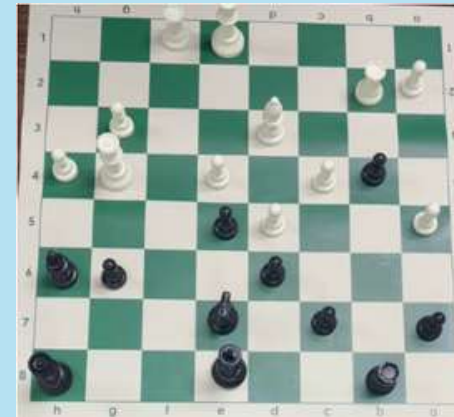


Video2Image

Image



Get ROI Image



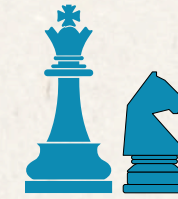
Hand Detection



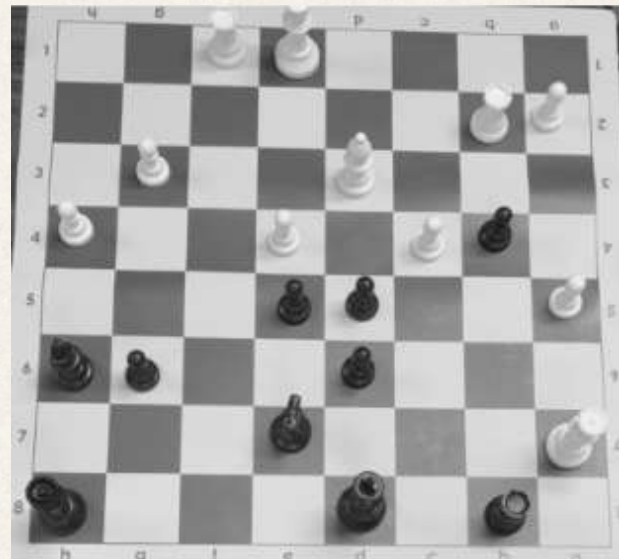
Dedup by Pixel Similarity



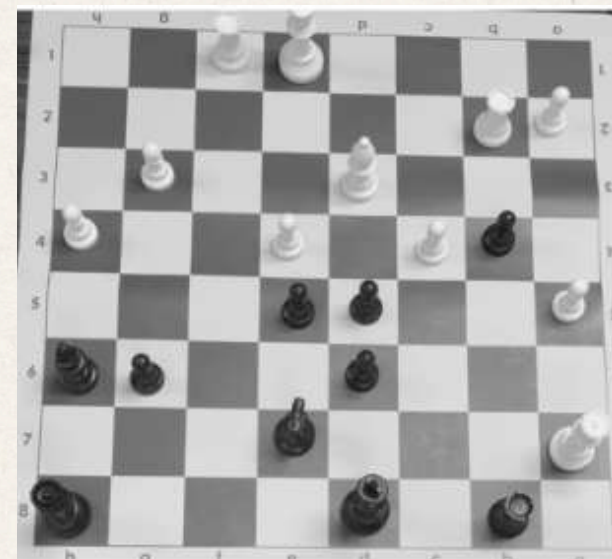
Dedup by pixels similarity



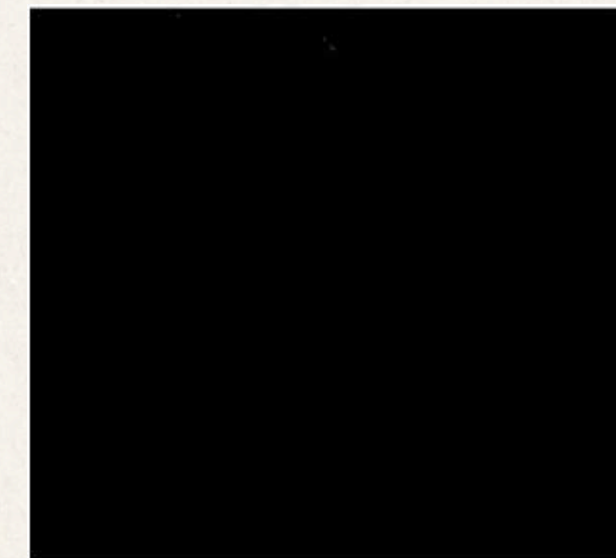
Same Frame



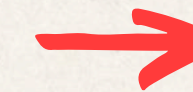
Current Frame
(Gray Scale)



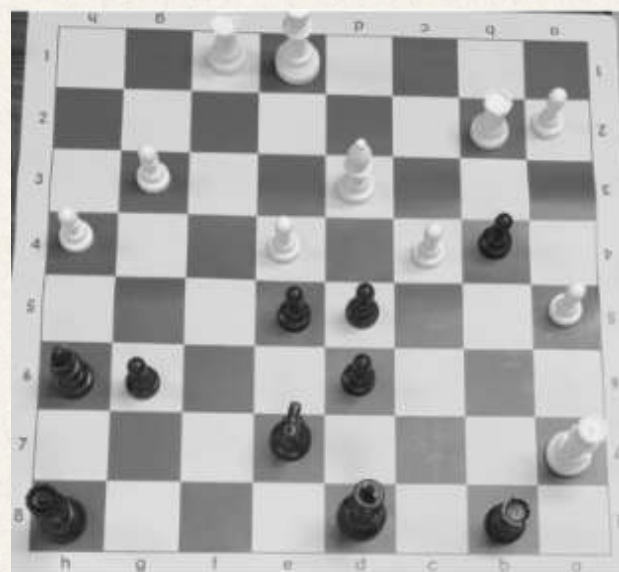
Current Frame
(Gaussian Blur)



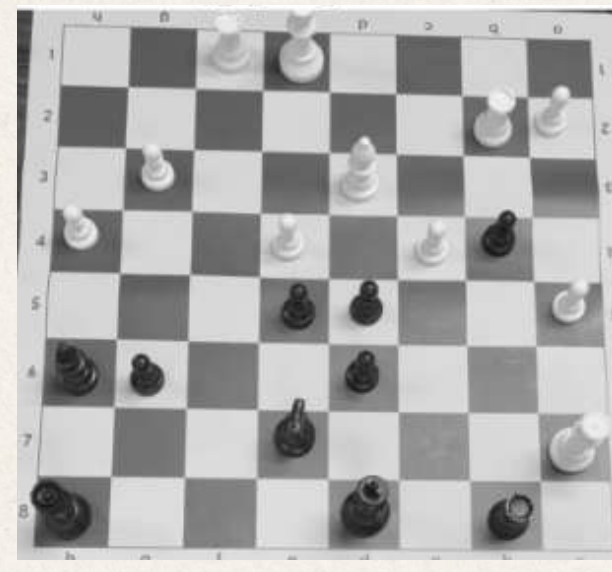
Difference Image
after Thresholding
(Pixel Change Detection)



Crop+ Morphological
(Noise Suppression)



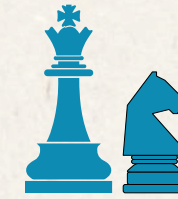
Previous Frame
(Gray Scale)



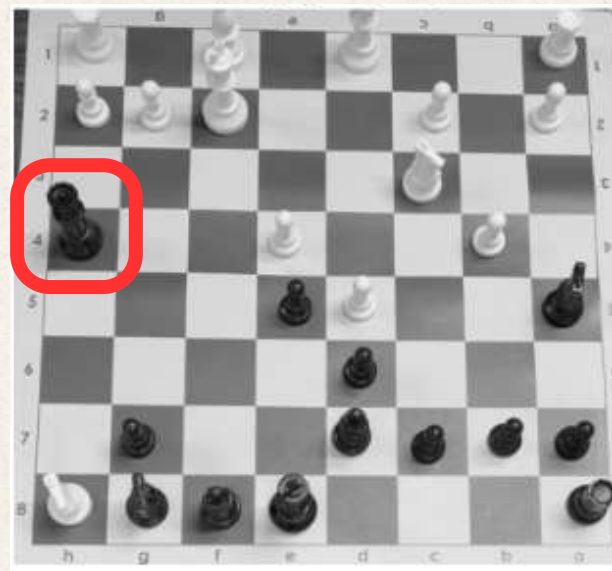
Previous Frame
(Gaussian Blur)



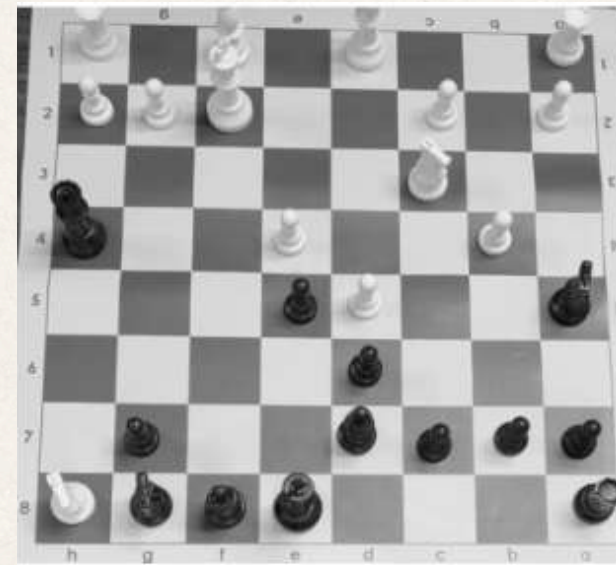
Dedup by pixels similarity



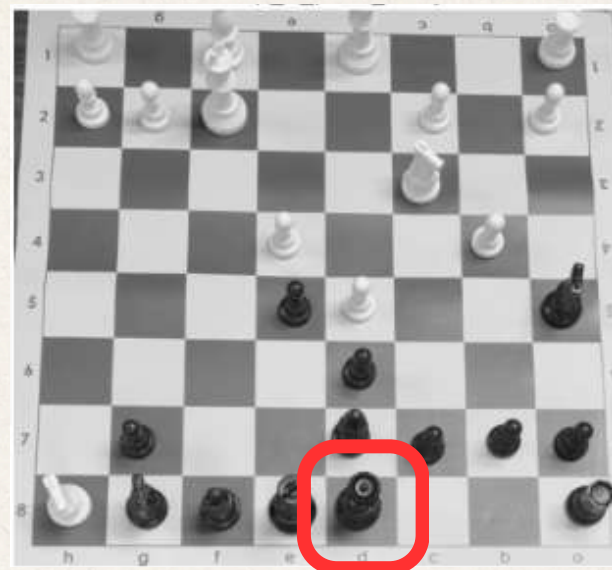
Different Frame



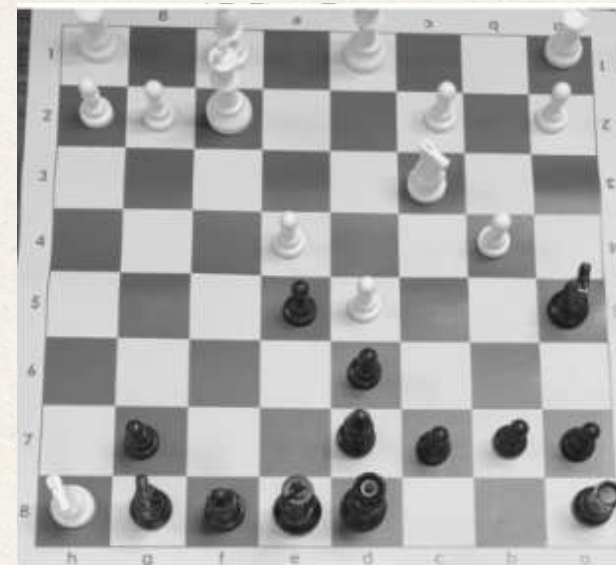
Current Frame
(Gray Scale)



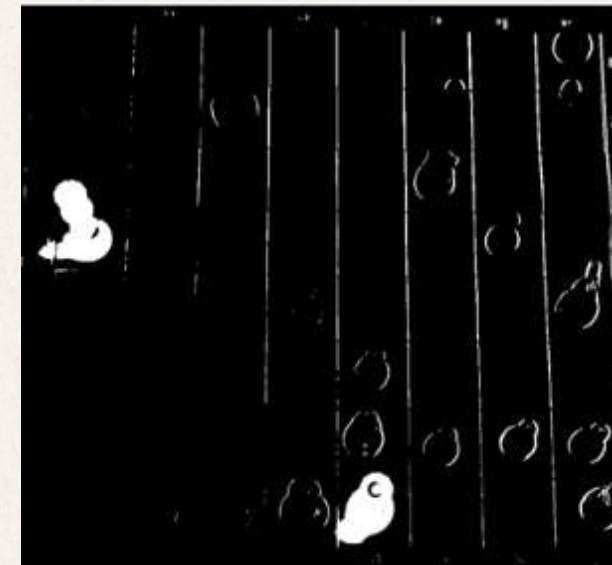
Current Frame
(Gaussian Blur)



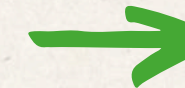
Previous Frame
(Gray Scale)



Previous Frame
(Gaussian Blur)



Difference Image
after Thresholding
(Pixel Change Detection)



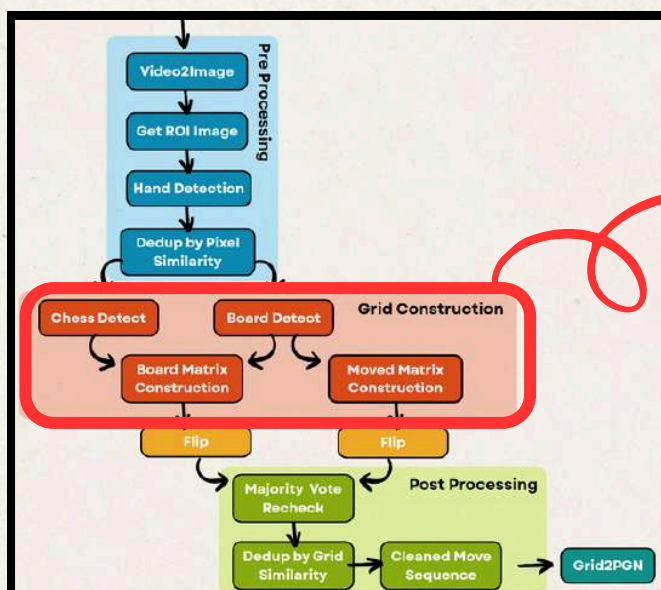
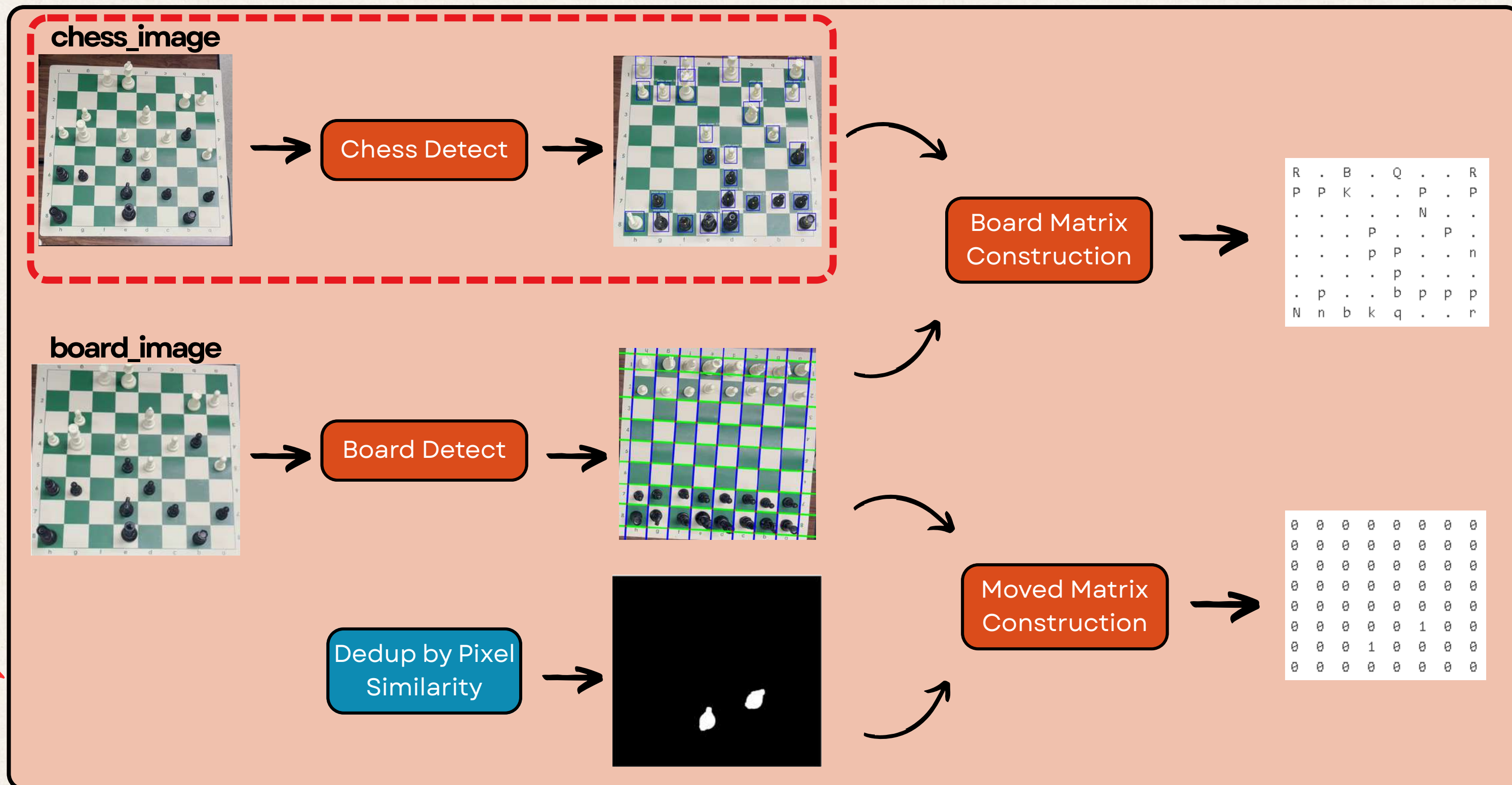
Crop+ Morphological
(Noise Suppression)





Matrix Construction

Pipeline : Matrix Construction



Pieces Detection



b : "black-bishop"

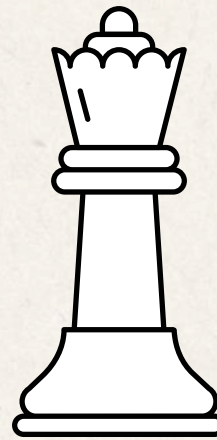
k : "black-king"

n : "black-knight"

p : "black-pawn"

q : "black-queen"

r : "black-rook"



B : "white-bishop"

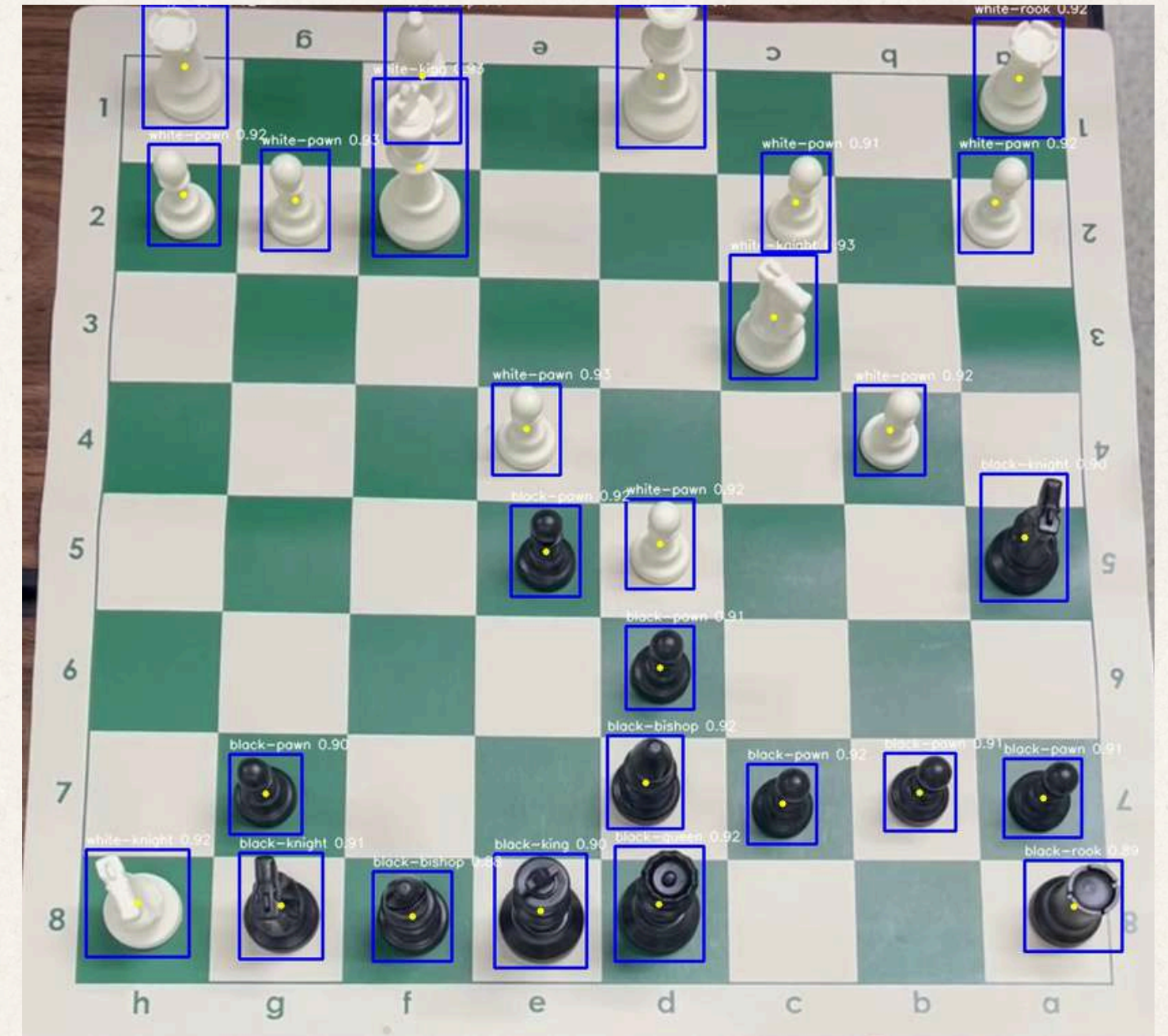
K : "white-king"

N : "white-knight"

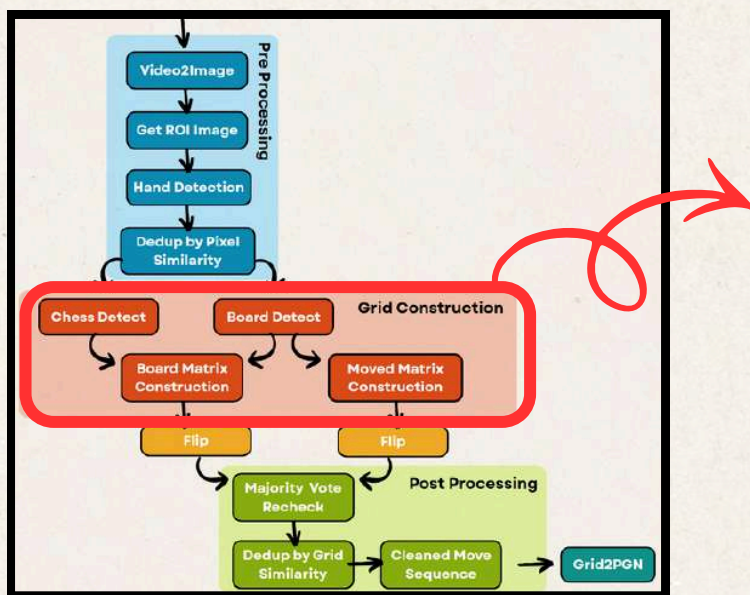
P : "white-pawn"

Q : "white-queen"

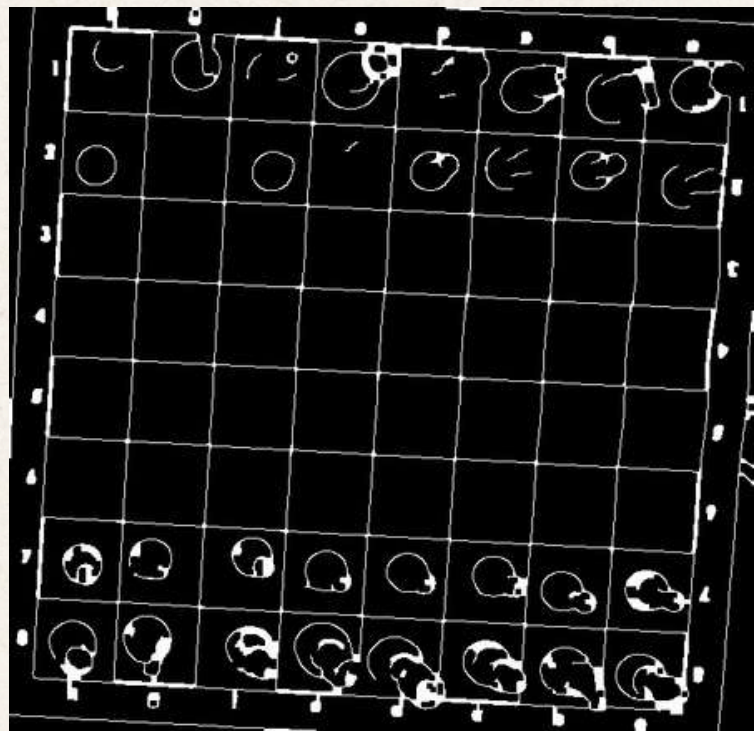
R : "white-rook"



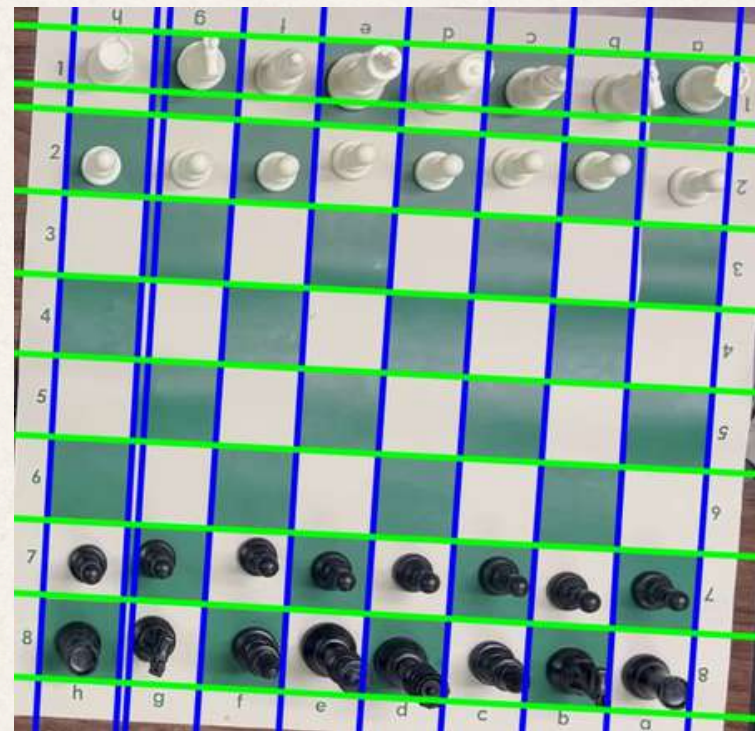
Special Yolo8m



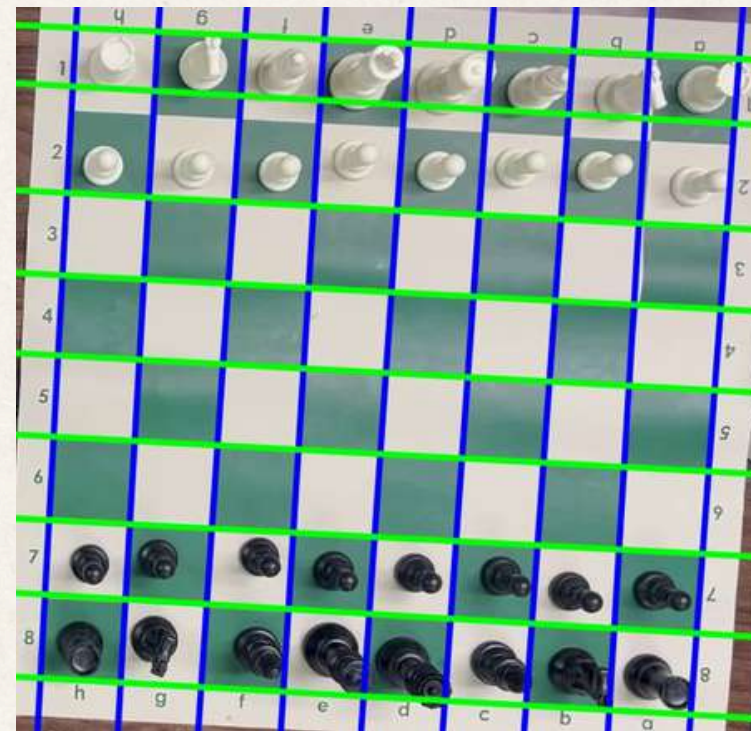
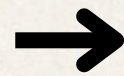
Board Detection



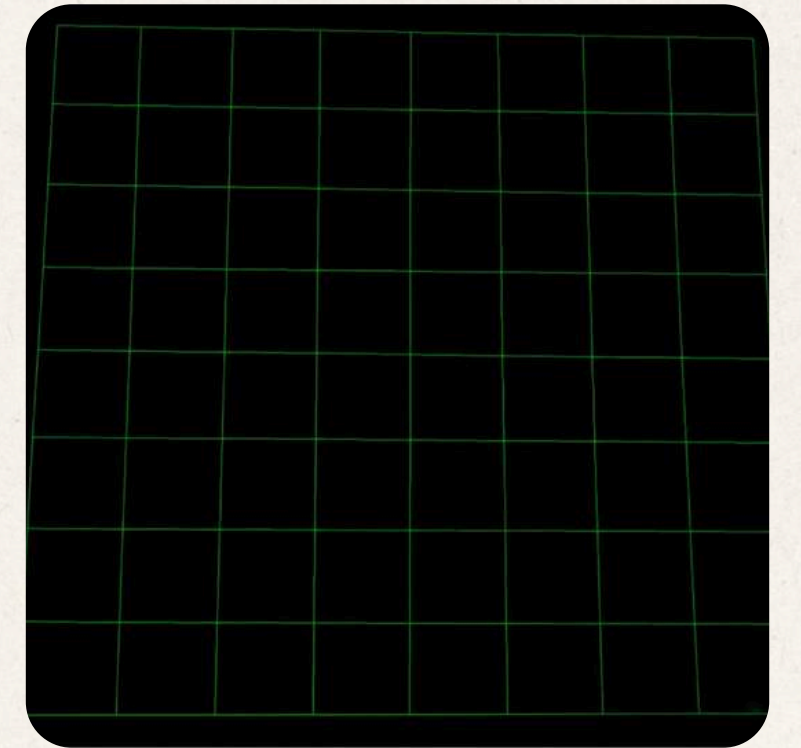
Canny Edge
Closing



LSD

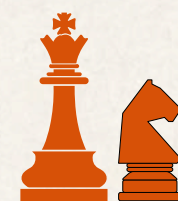


Hungarian



Board Grid

Pipeline : Matrix Construction



chess_image



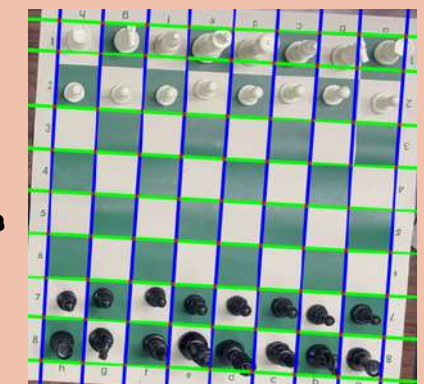
Chess Detect



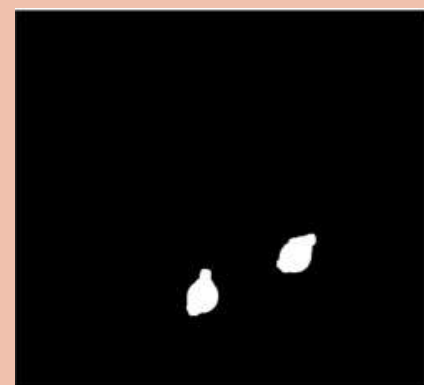
board_image



Board Detect



Dedup by Pixel Similarity

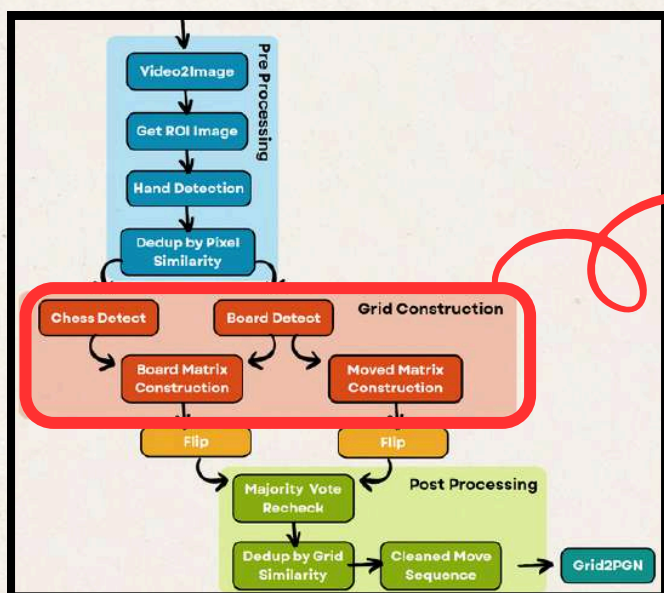


Board Matrix Construction

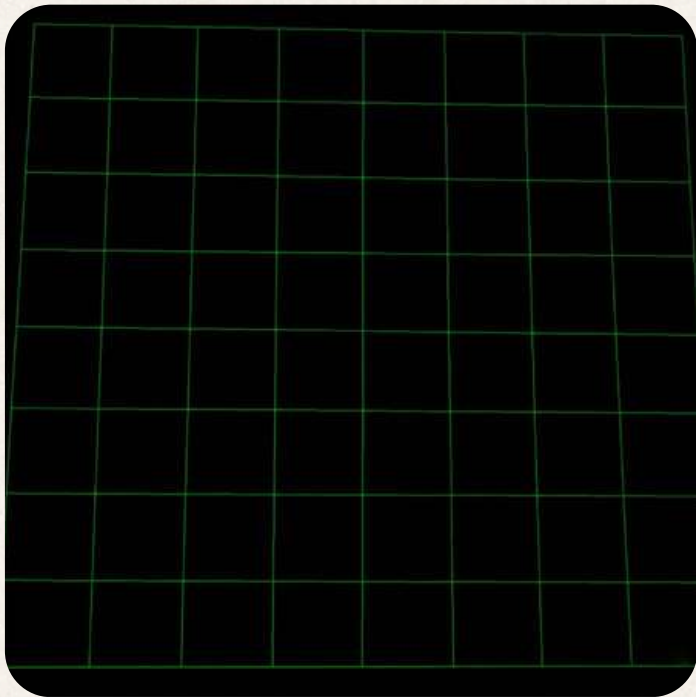
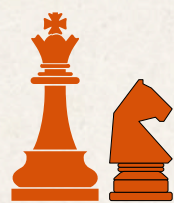
R	.	B	.	Q	.	.	R
P	P	K	.	.	P	.	P
.	N	.	.
.	.	.	P	.	.	P	.
.	.	.	p	P	.	.	n
.	.	.	.	p	.	.	.
.	p	.	.	b	p	p	p
N	n	b	k	q	.	.	r

Moved Matrix Construction

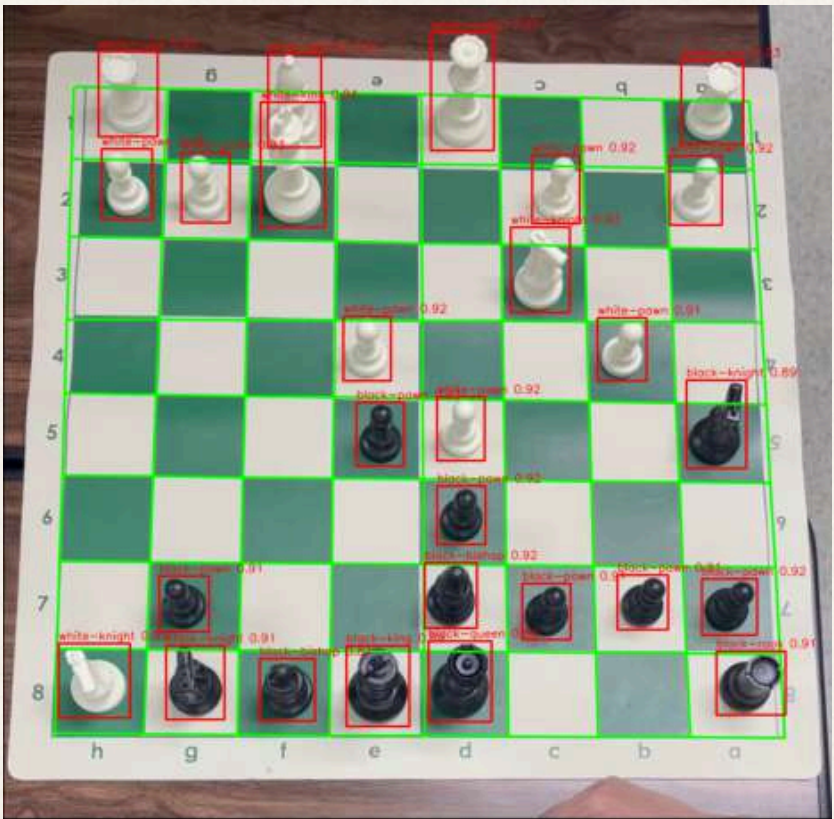
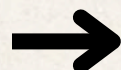
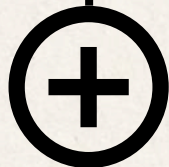
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	1	0	0
0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	0



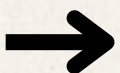
Board Matrix



Board Detection



ChessBoard Detection

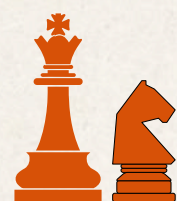


R	.	B	.	Q	.	.	R
P	P	K	.	.	P	.	P
.	N	.	.
.	.	.	P	.	.	P	.
.	.	.	p	P	.	.	n
.	.	.	.	p	.	.	.
.	p	.	.	b	p	p	p
N	n	b	k	q	.	.	r



Chess Detection

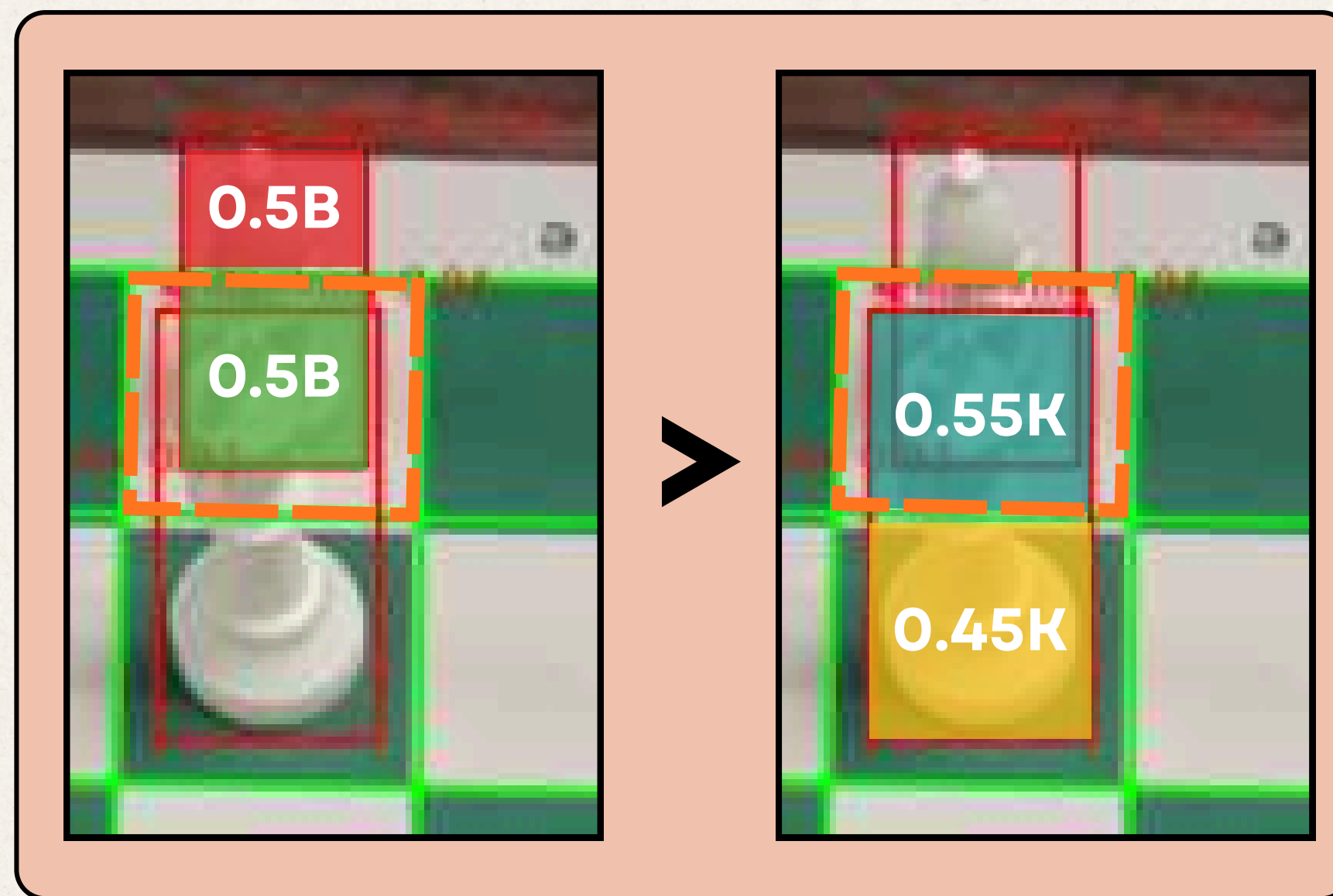
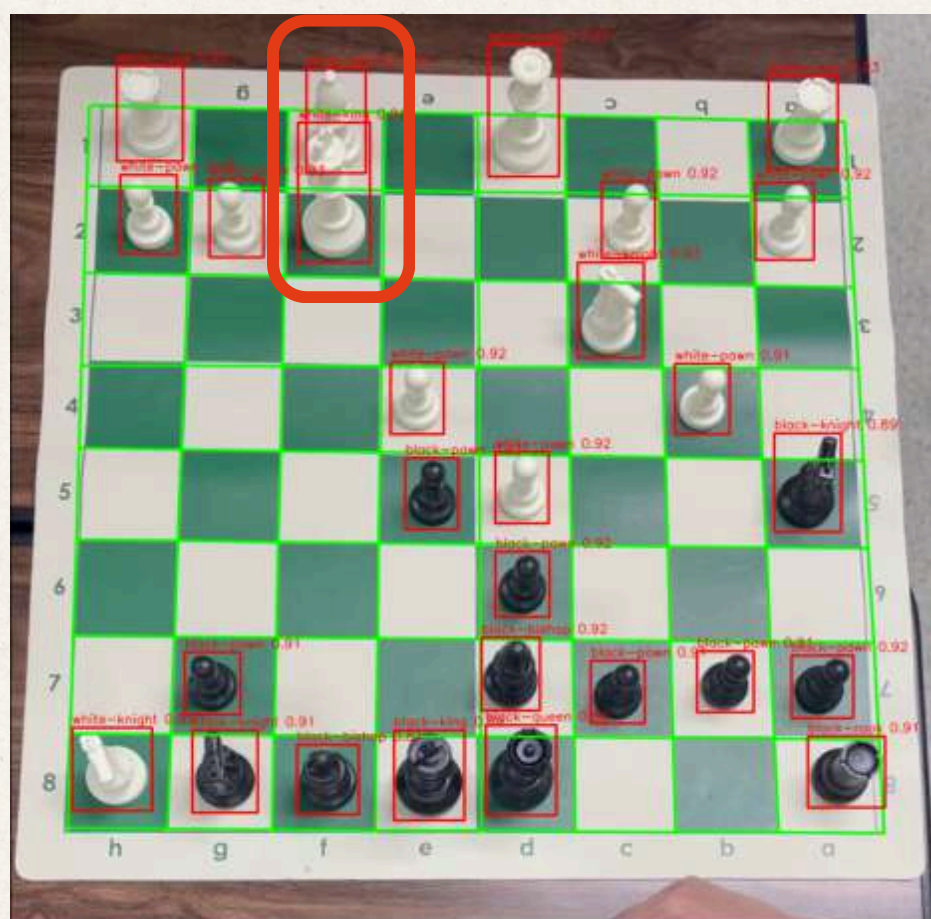
Board Matrix



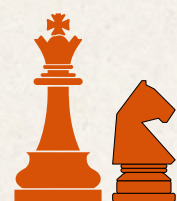
Score = Piece area in Cell / Piece area in Board



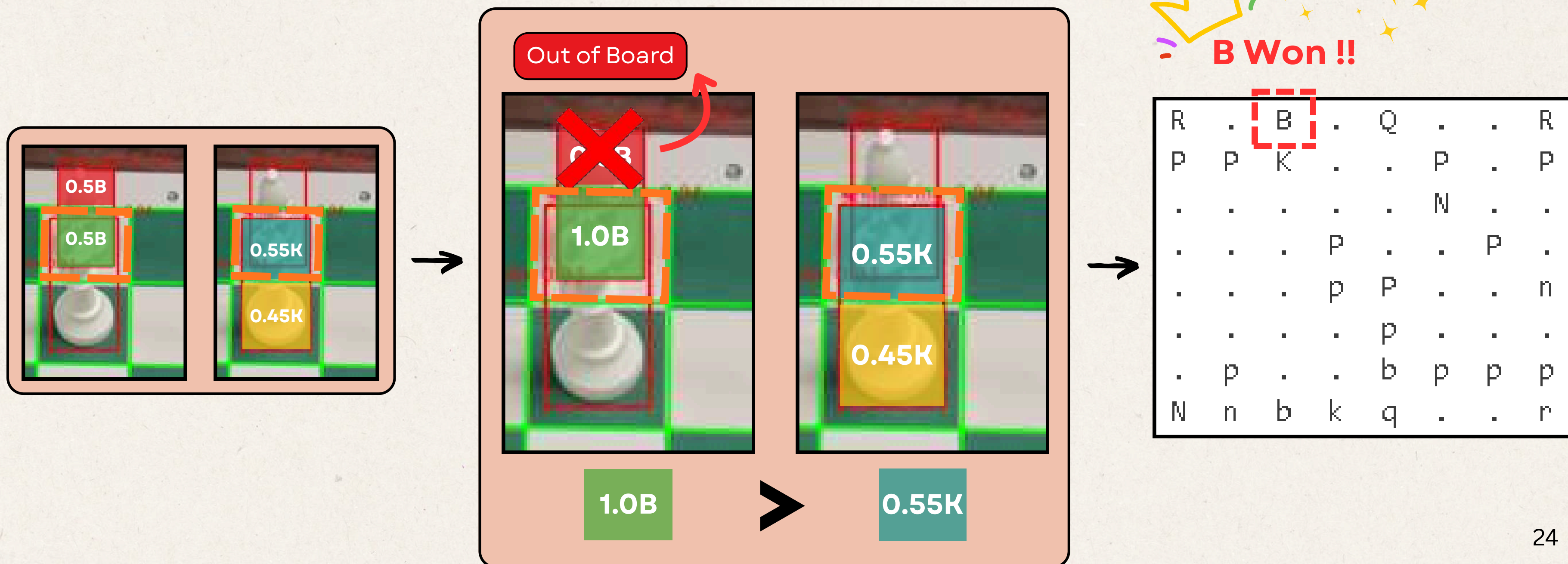
- Red Box = Area B out of Cell
- Green Box = Area B in Cell
- Blue Box = Area K in Cell
- Yellow Box = Area K out of Cell
- Orange Outline = Cell boundary



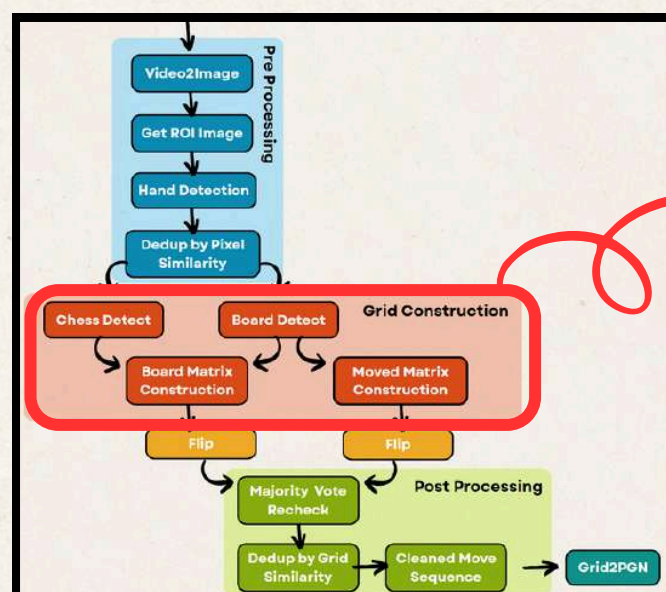
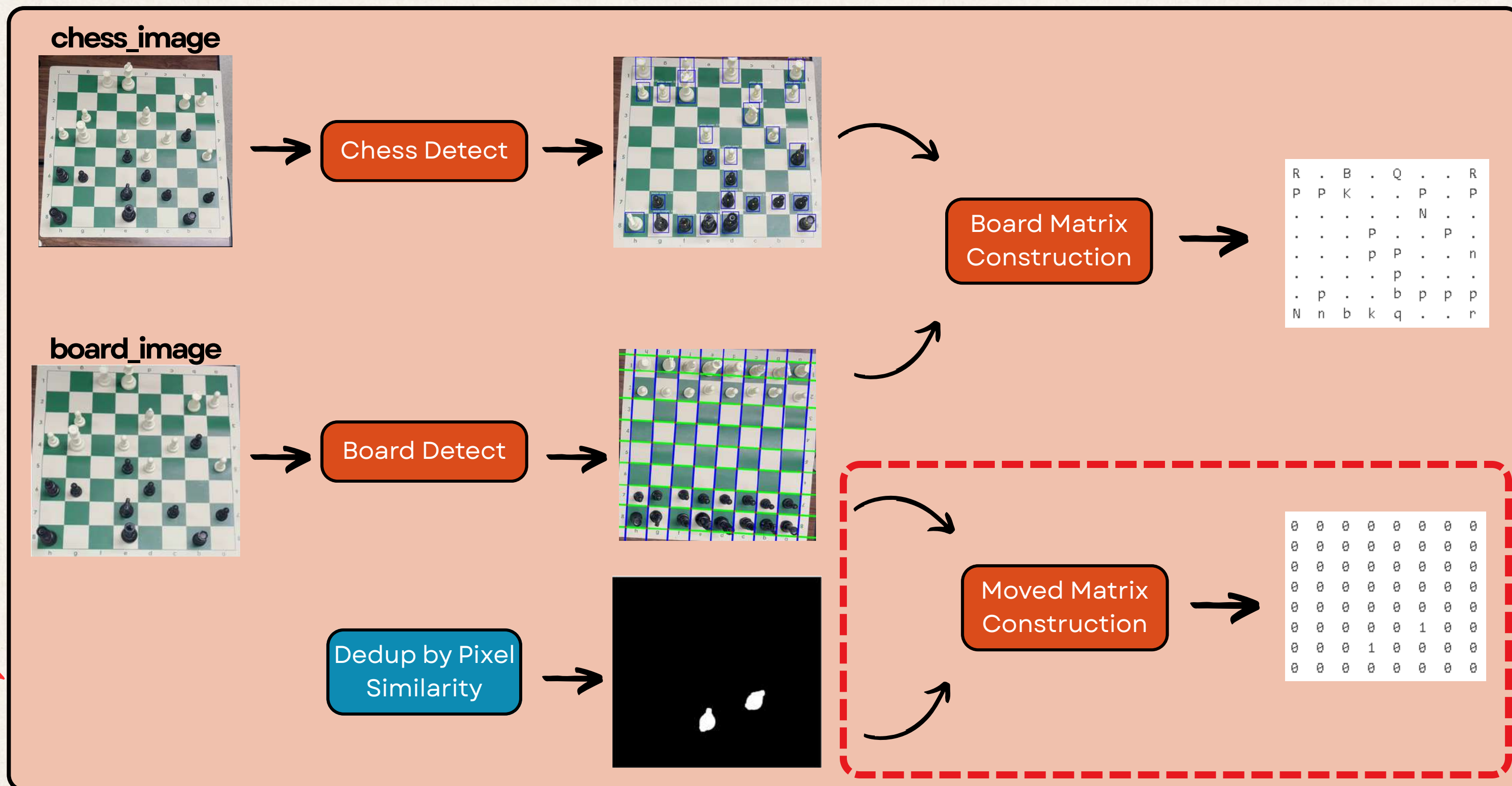
Board Matrix



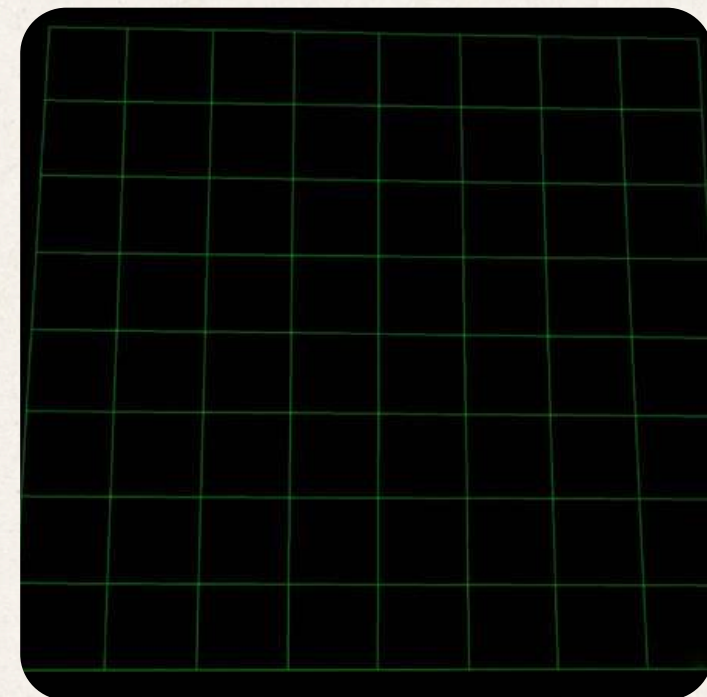
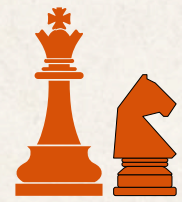
Score = Piece area in Cell / Piece area in Board



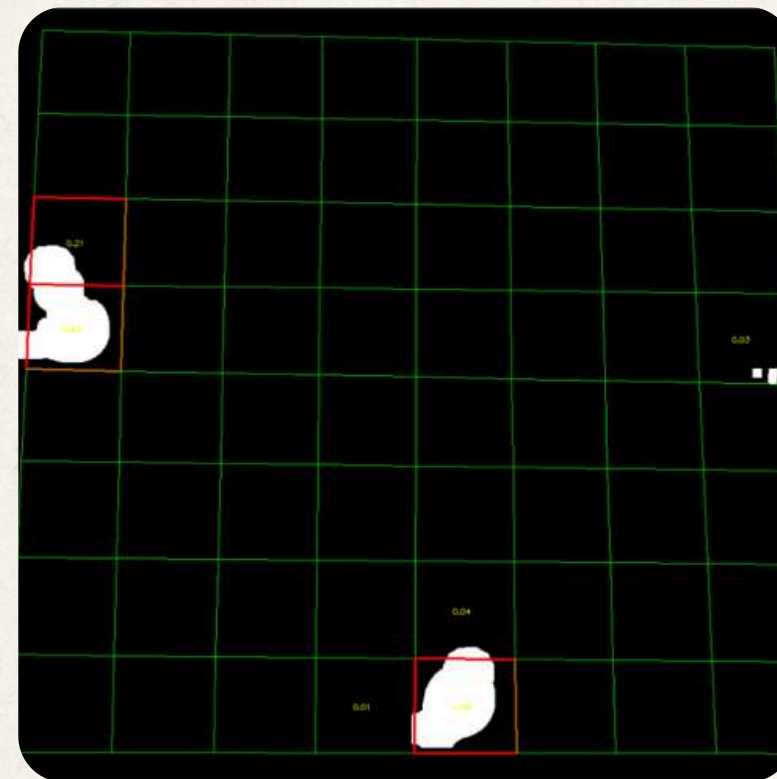
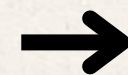
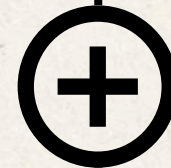
Pipeline : Matrix Construction



Moved Matrix



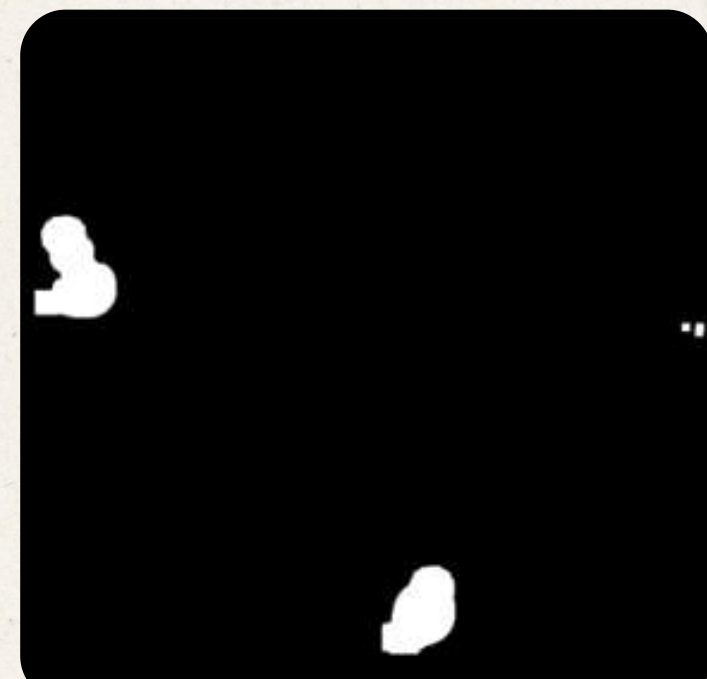
Board Detection



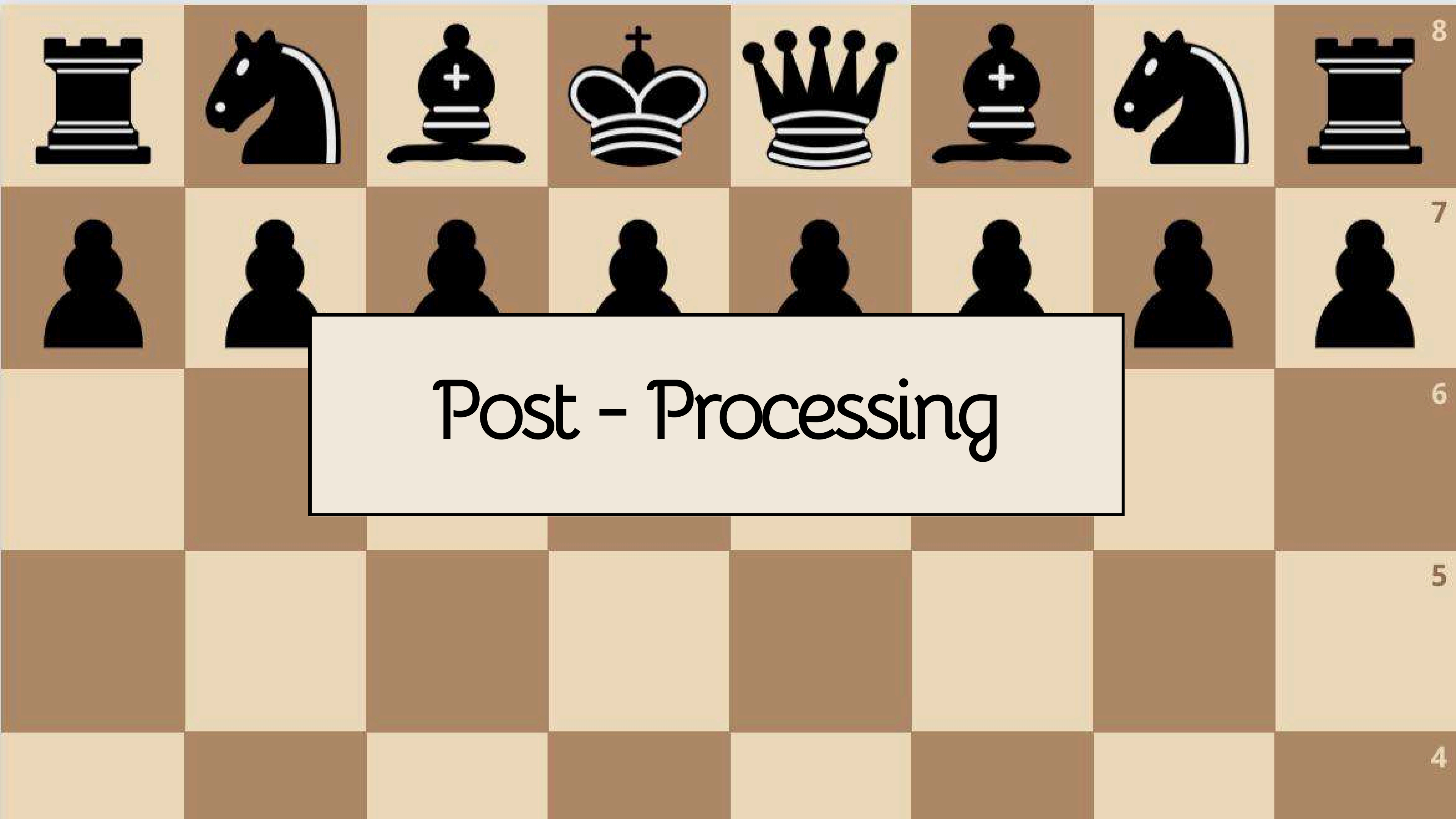
Movable Detection



0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0

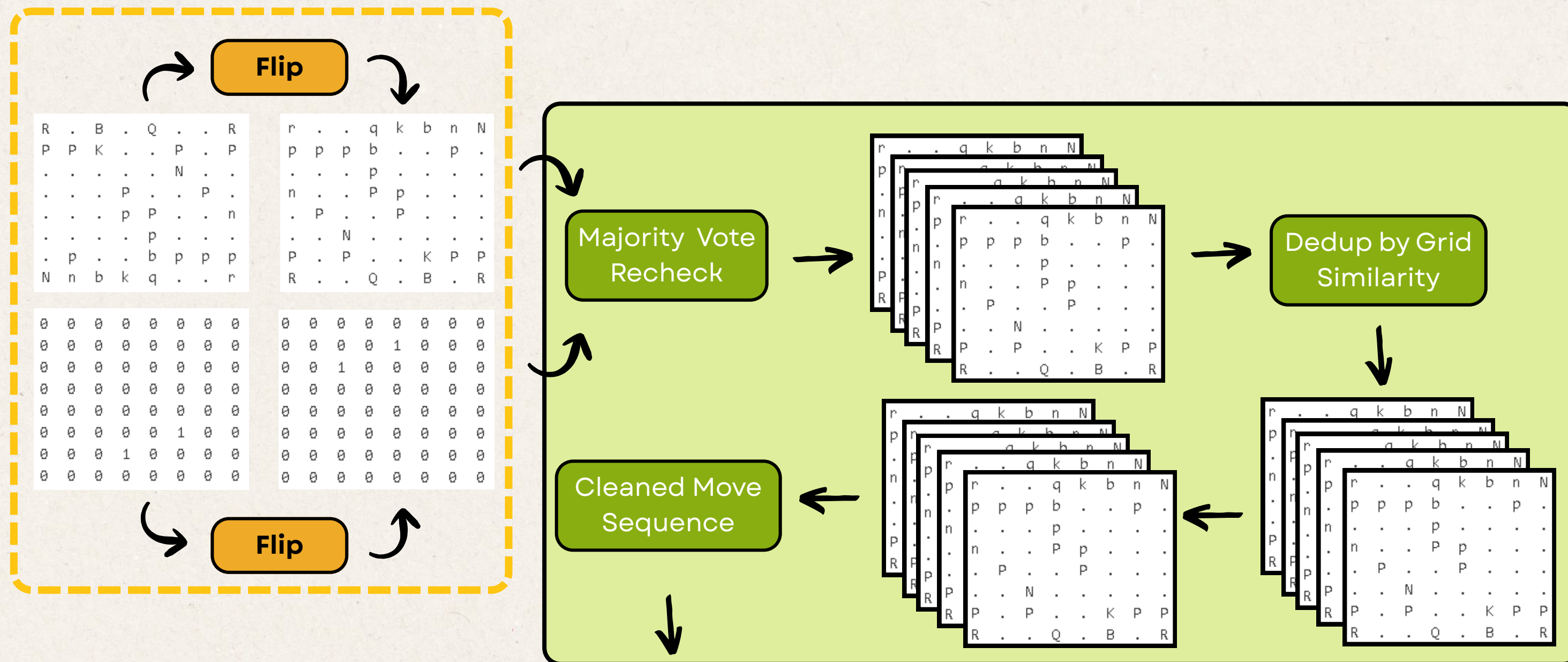
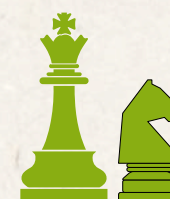


Movable Detection



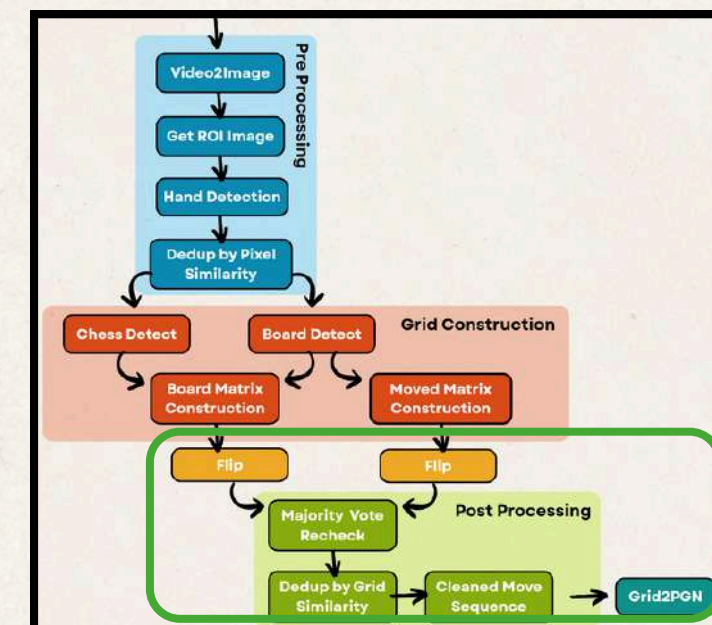
Post - Processing

Pipeline : Post - Processing

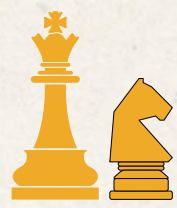


Grid2PGN

1... Qh4+ 2. g3



Flip / Rotation

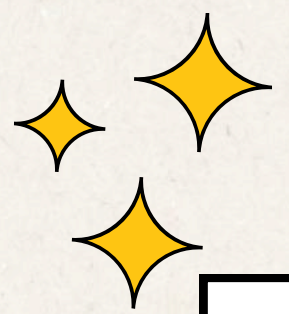


Left : 4

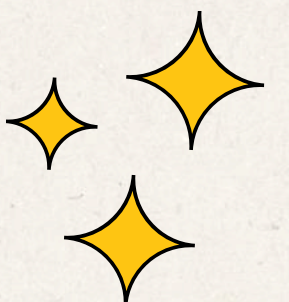
R	.	B	.	Q	.	.	R
P	P	K	.	.	P	.	P
.	N	.	.
.	.	.	P	.	.	P	.
.	.	.	p	P	.	.	n
.	.	.	.	p	.	.	.
.	p	.	.	b	p	p	p
N	n	b	k	q	.	.	r

Right : 3

Bottom : 1

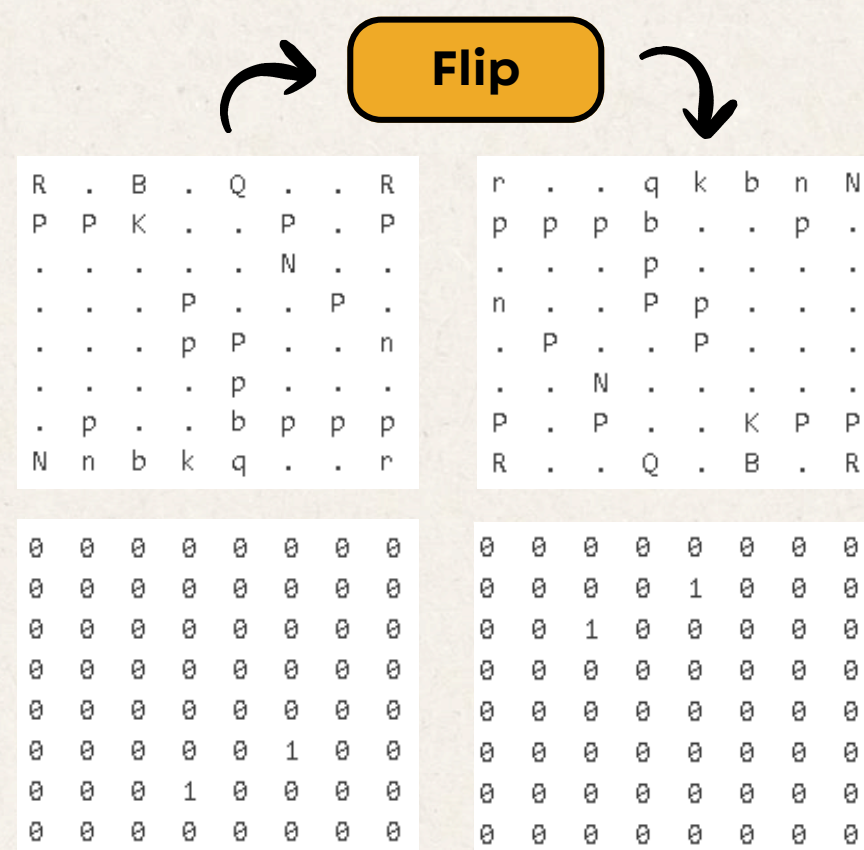
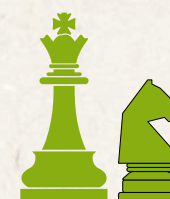


r	.	.	q	k	b	n	N
p	p	p	b	.	.	p	.
.	.	.	p
n	.	.	P	p	.	.	.
.	P	.	.	P	.	.	.
.	.	N
P	.	P	.	.	K	P	P
R	.	.	Q	.	B	.	R



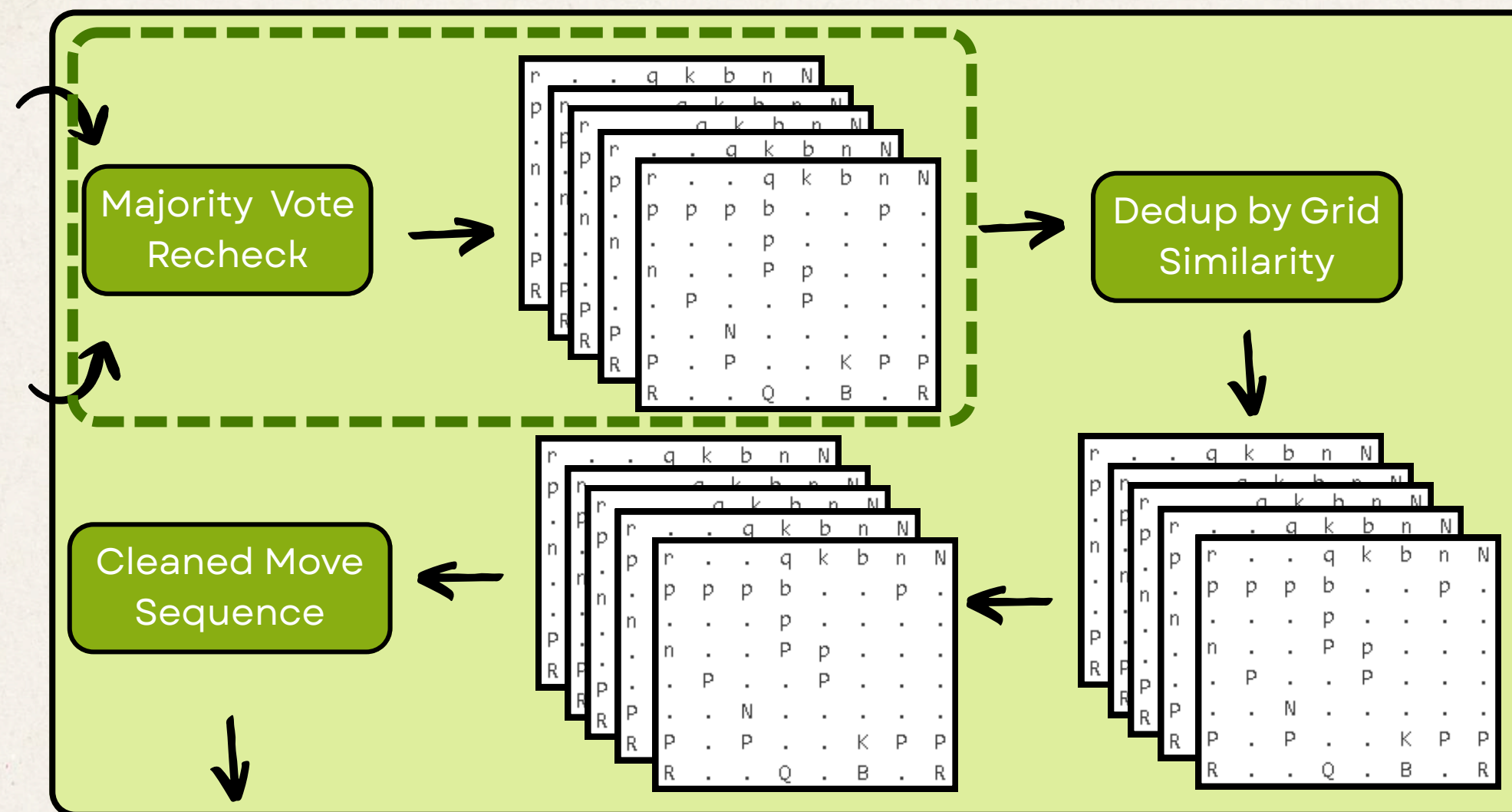
Standard Layout Board Matrix

Pipeline : Post - Processing



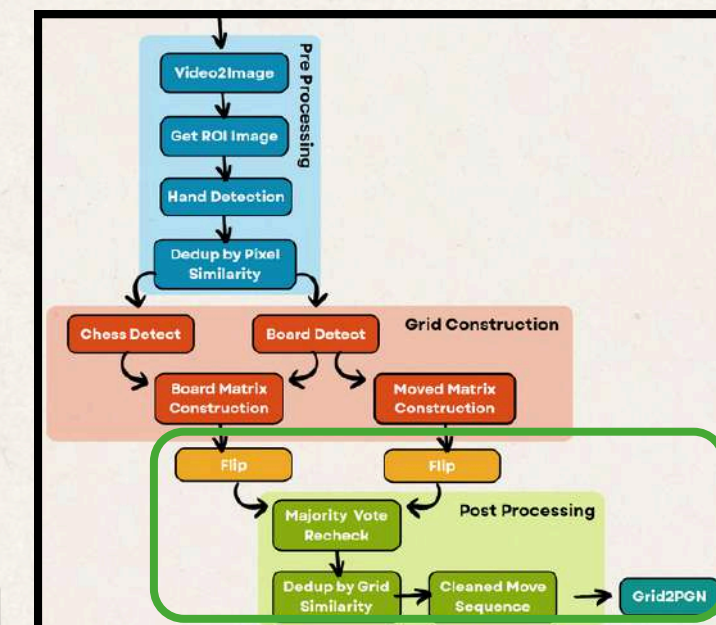
Flip

Flip

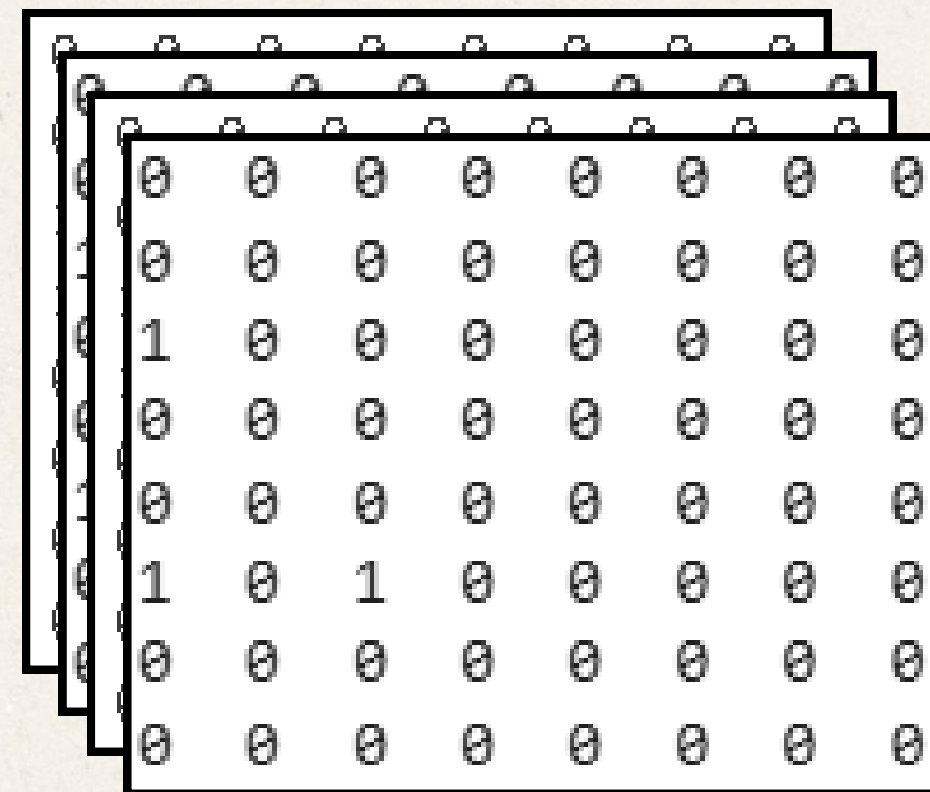
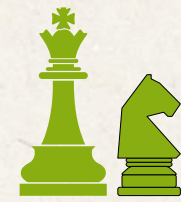


Grid2PGN

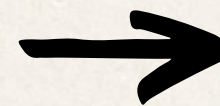
1... Qh4+ 2. g3



Majority Vote Recheck



0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0



When To Move

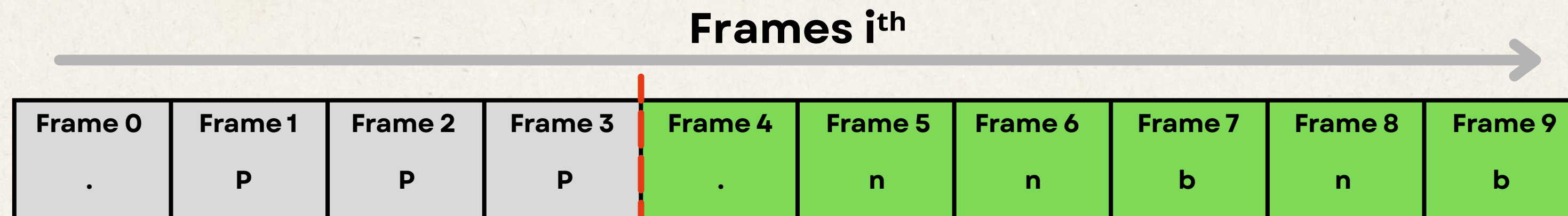
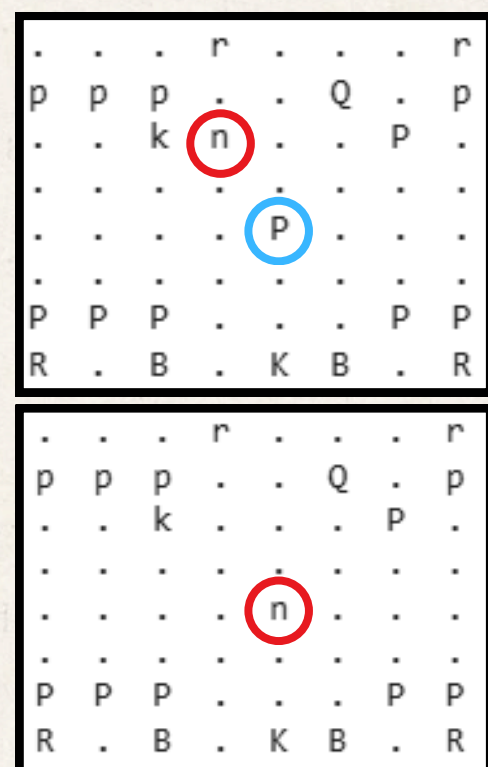
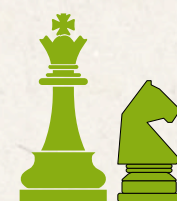
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0



MOVE !



Majority Vote Recheck

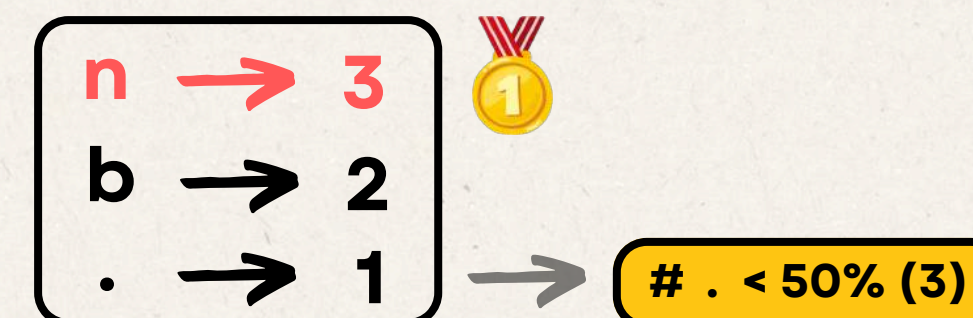


Move

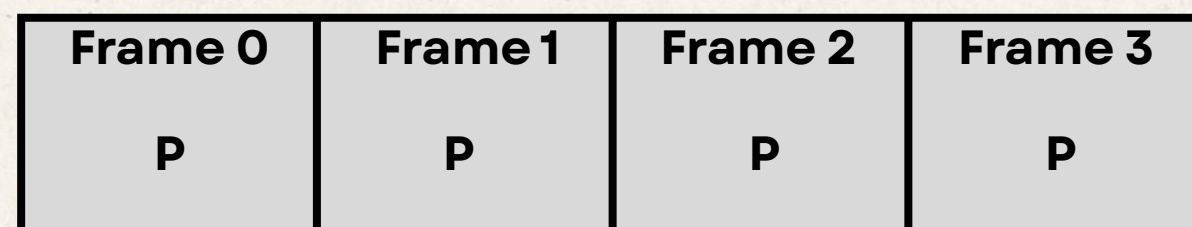
Group 1



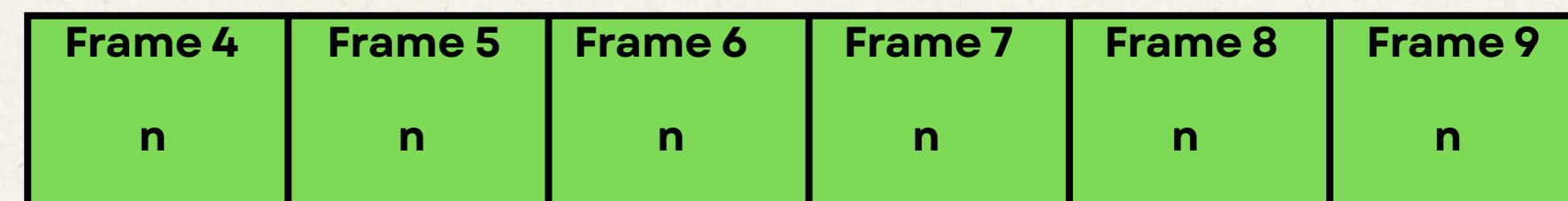
Group 2



Majority vote

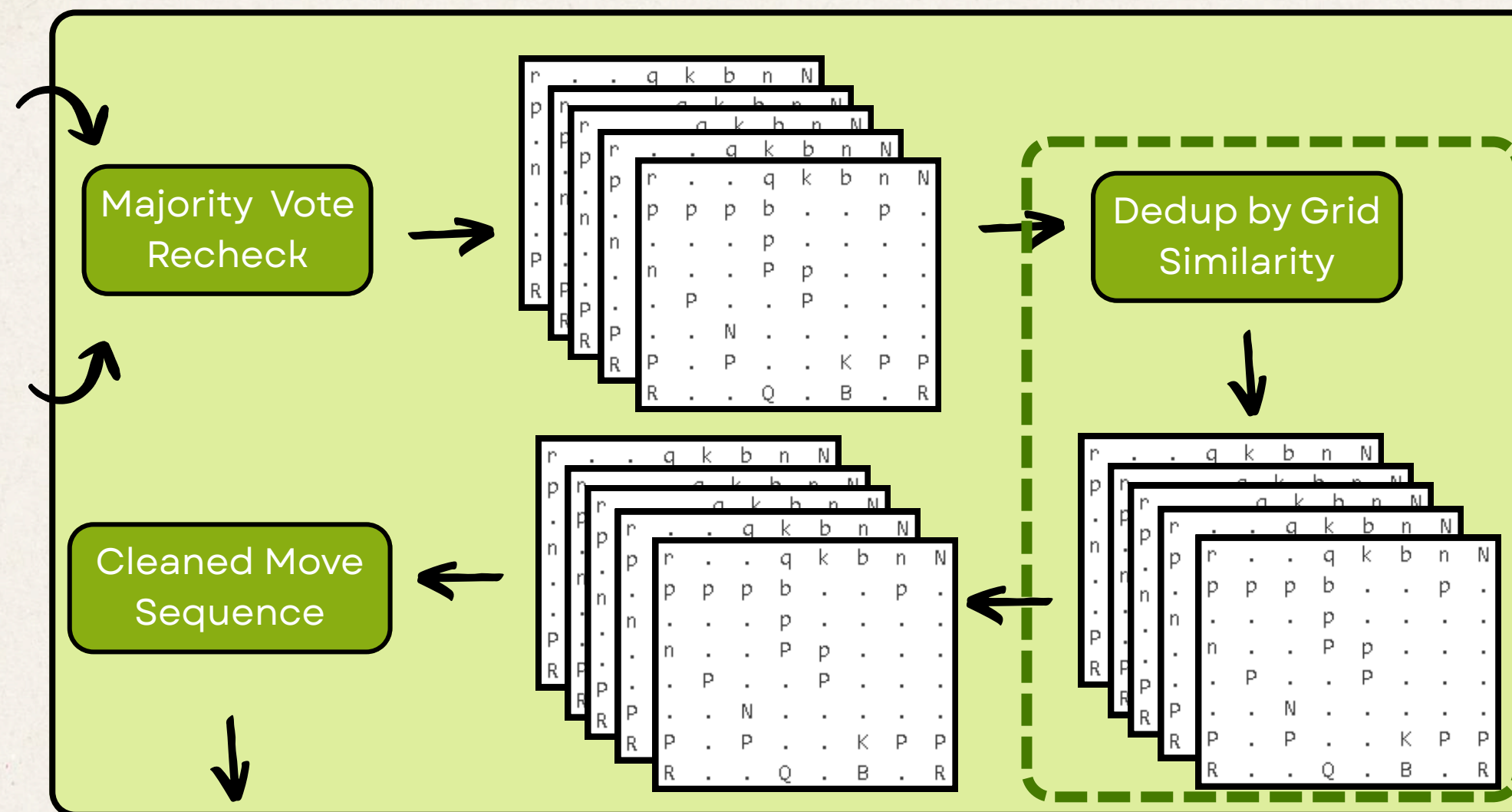
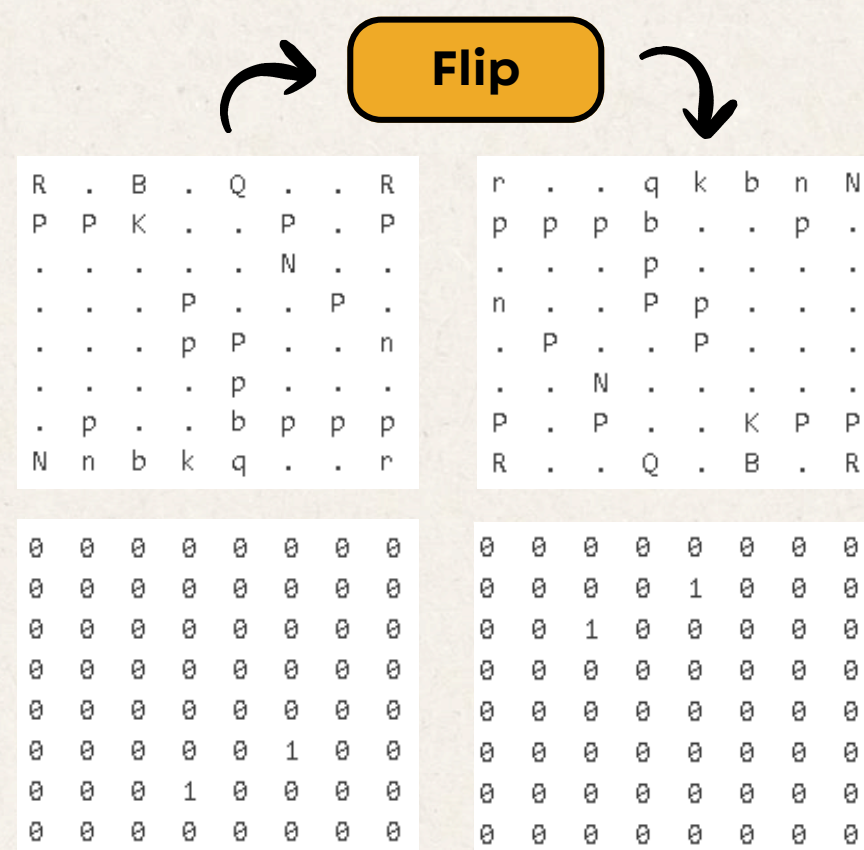
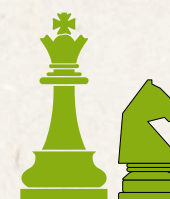


Group 1



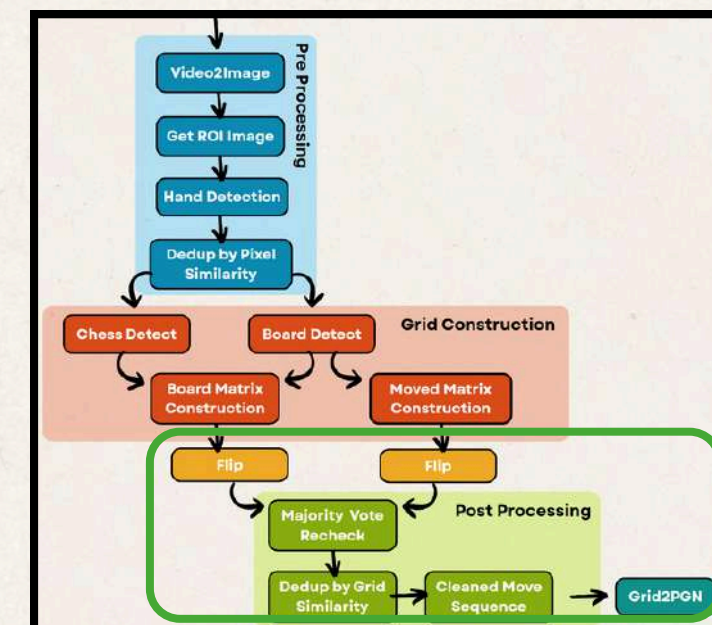
Group 2

Pipeline : Post - Processing



Grid2PGN

1... Qh4+ 2. g3



Deduplication (Grid Similarity)



- Start by keeping the **first board** as the initial unique frame
- For each next frame:

Compare with the last kept board using `array_equal`

- If different → keep it
- If identical → skip it (duplicate)



r	.	.	q	k	b	n	N
p	p	p	b	.	.	p	.
.	.	.	p
n	.	.	P	p	.	.	.
.	P	.	.	P	.	.	.
.	.	N
P	.	P	.	.	K	P	P
R	.	.	Q	.	B	.	R

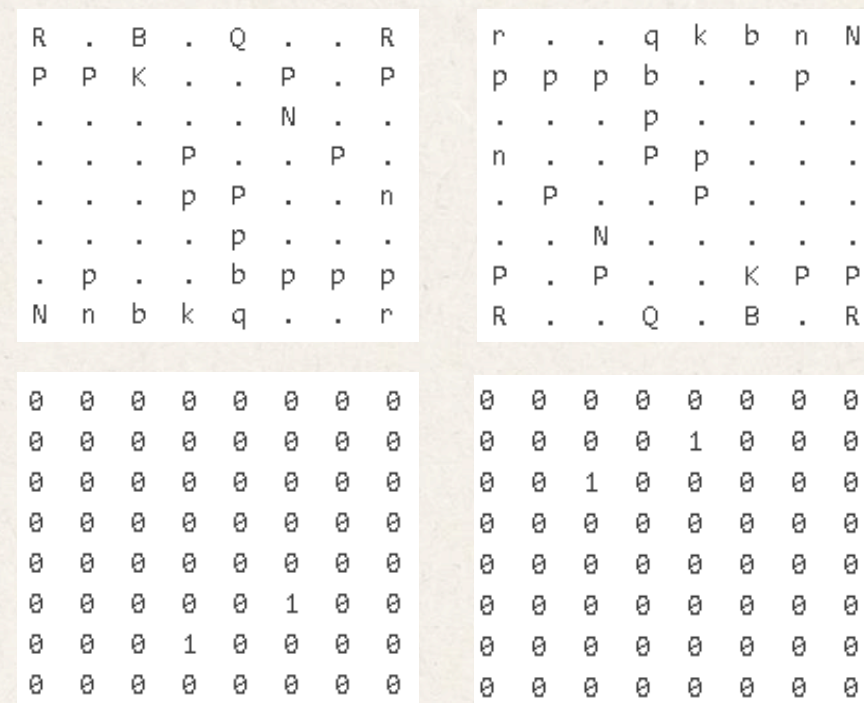
≠

r	.	.	q	k	b	n	N
p	p	p	b	.	.	p	.
.
n
.	P
.	.	N
P	.	P	.	.	K	P	P
R	.	.	Q	.	B	.	R

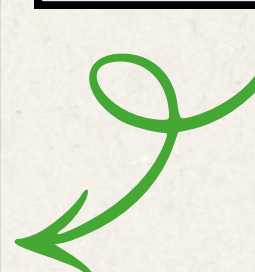
=

r	.	.	q	k	b	n	N
p	p	p	b	.	.	p	.
.	.	.	p
n	.	.	P	p	.	.	.
.	P	.	.	P	.	.	.
.	.	N
P	.	P	.	.	K	P	P
R	.	.	Q	.	B	.	R

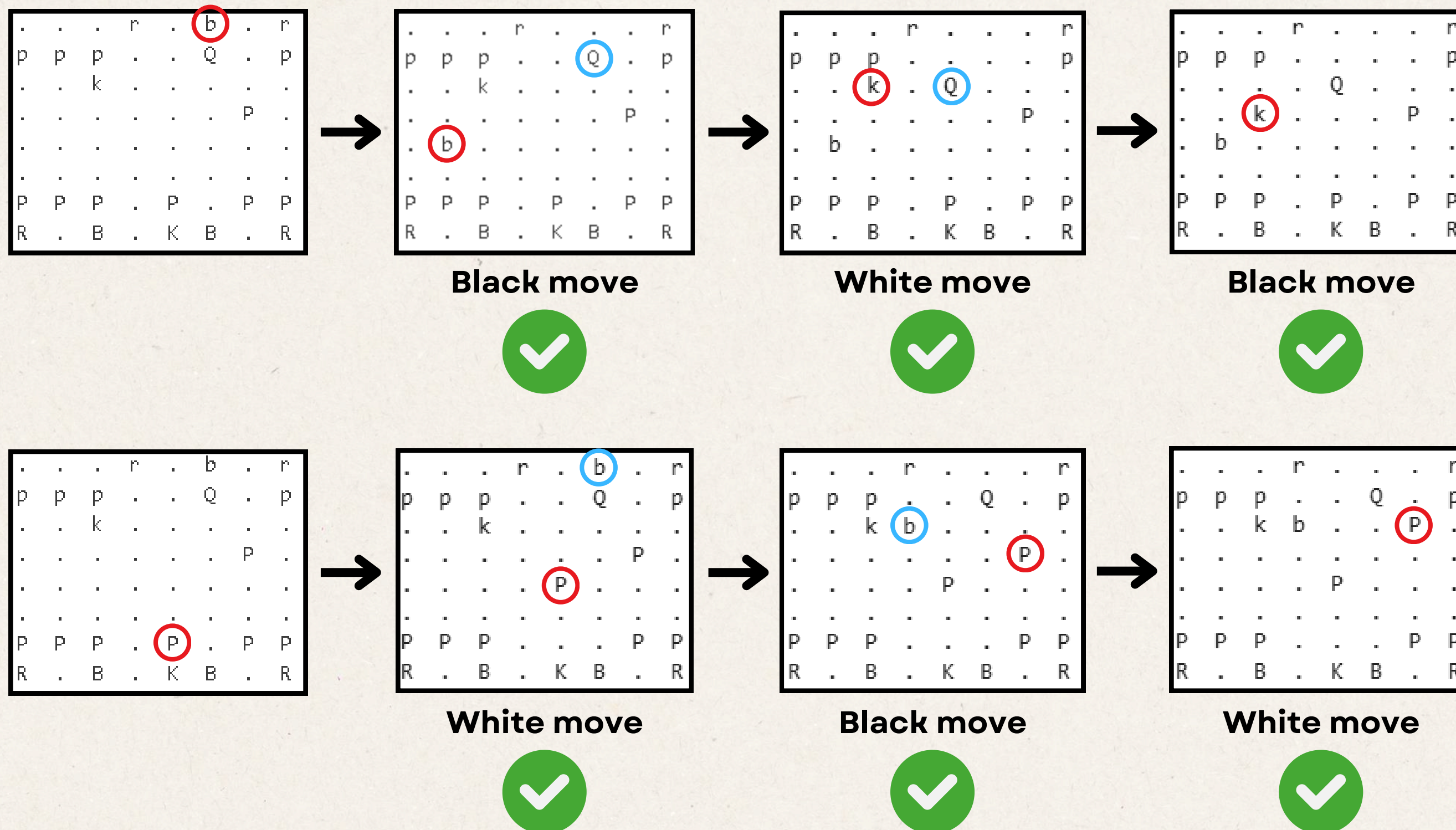
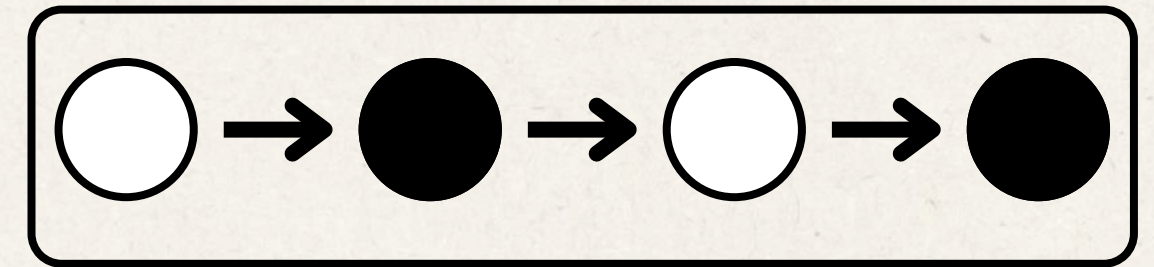
r	.	.	q	k	b	n	N
p	p	p	b	.	.	p	.
.	.	.	p
n	.	.	P	p	.	.	.
.	P	.	.	P	.	.	.
.	.	N
P	.	P	.	.	K	P	P
R	.	.	Q	.	B	.	R



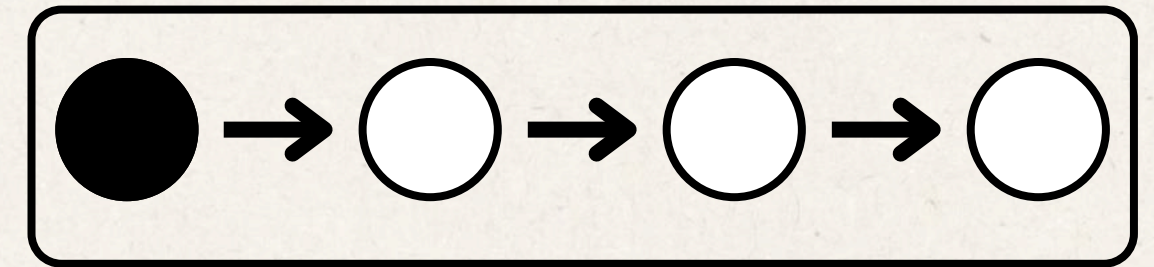
1. . . Qh4+
2. g3



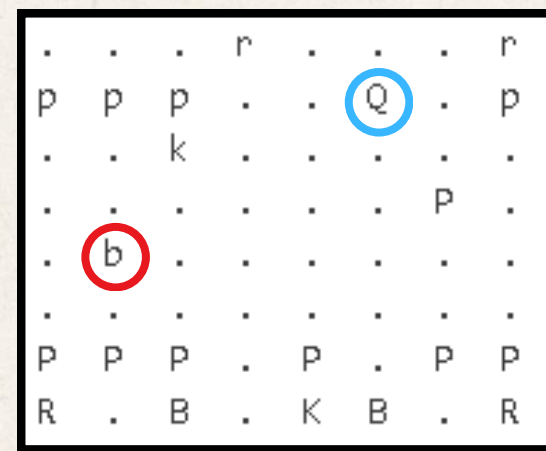
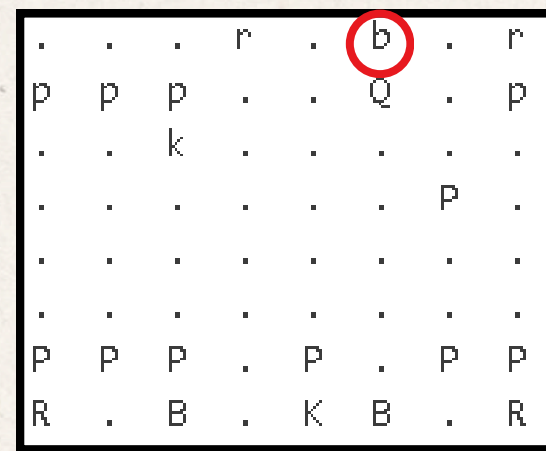
Clean Board Sequence



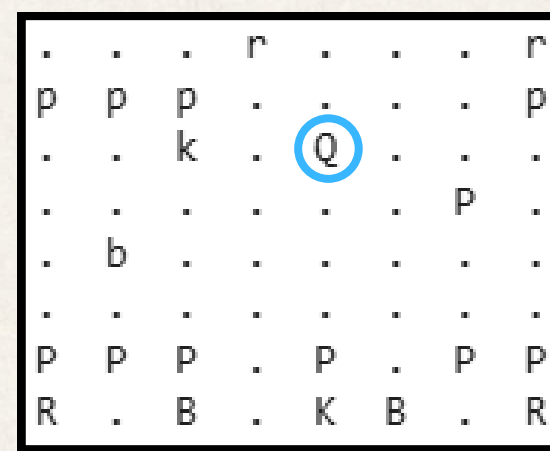
Clean Board Sequence



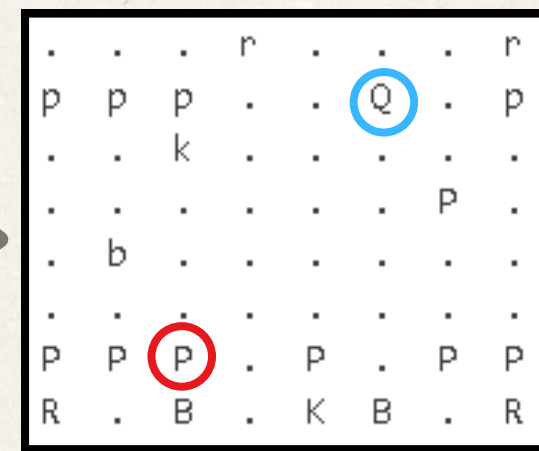
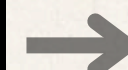
SKIP !



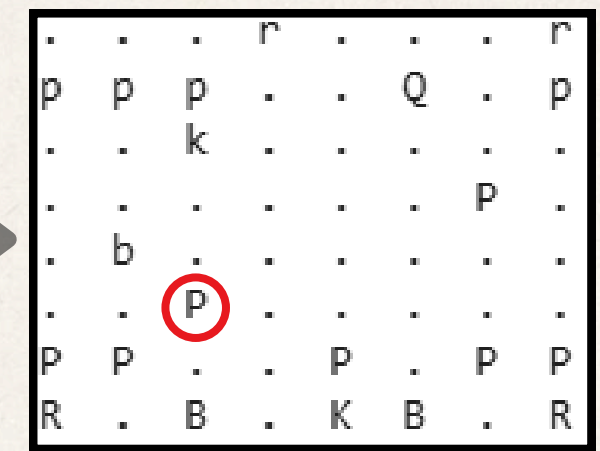
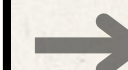
Black move



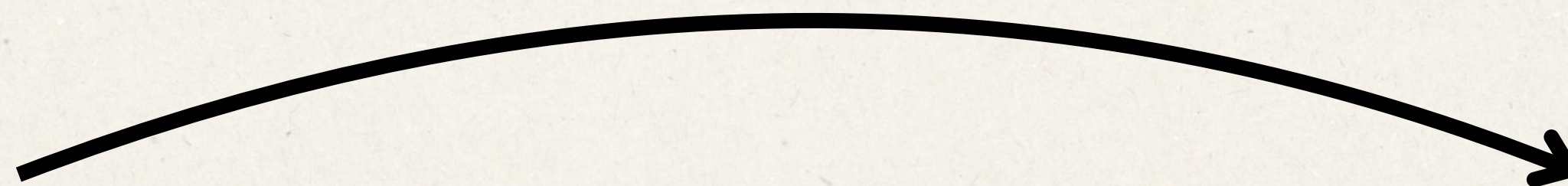
White move



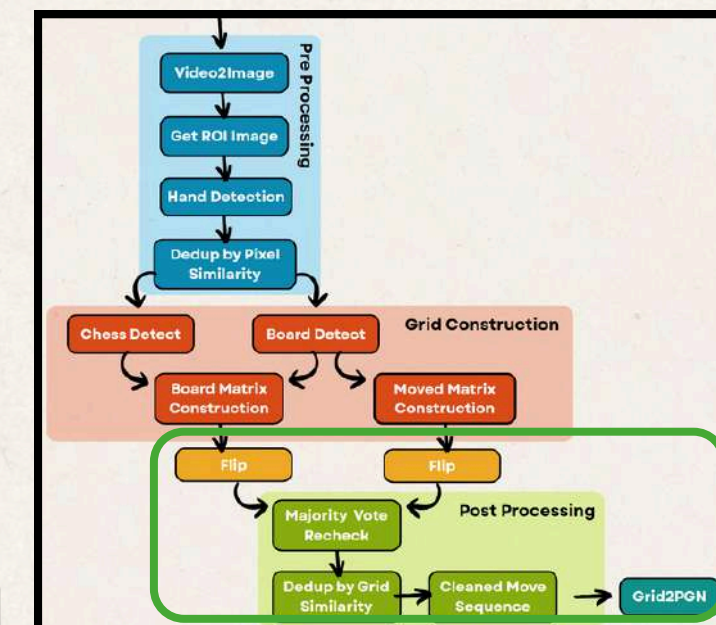
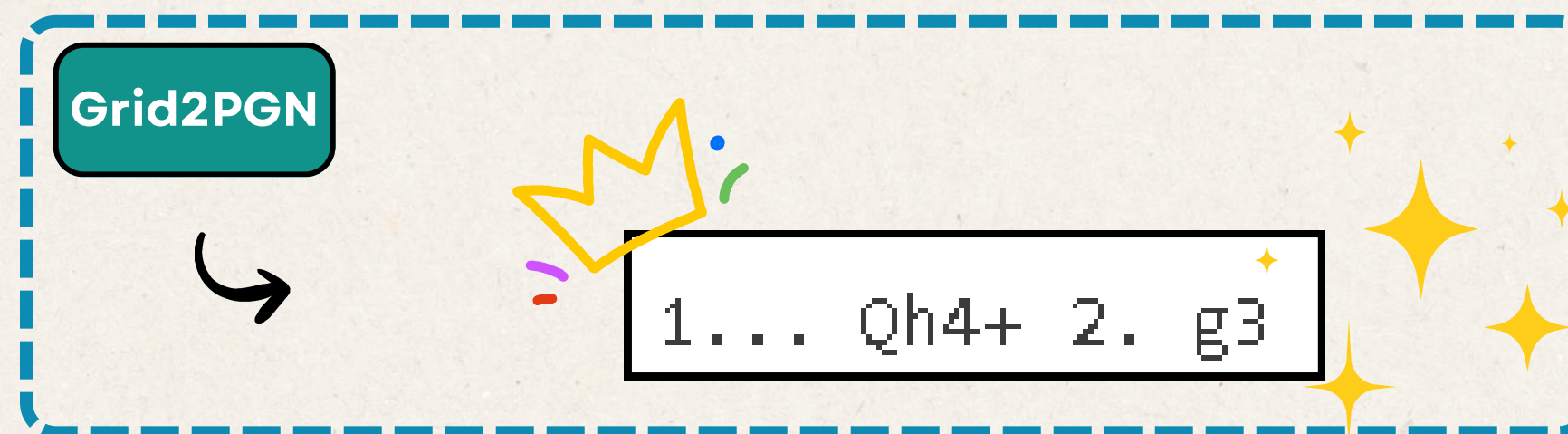
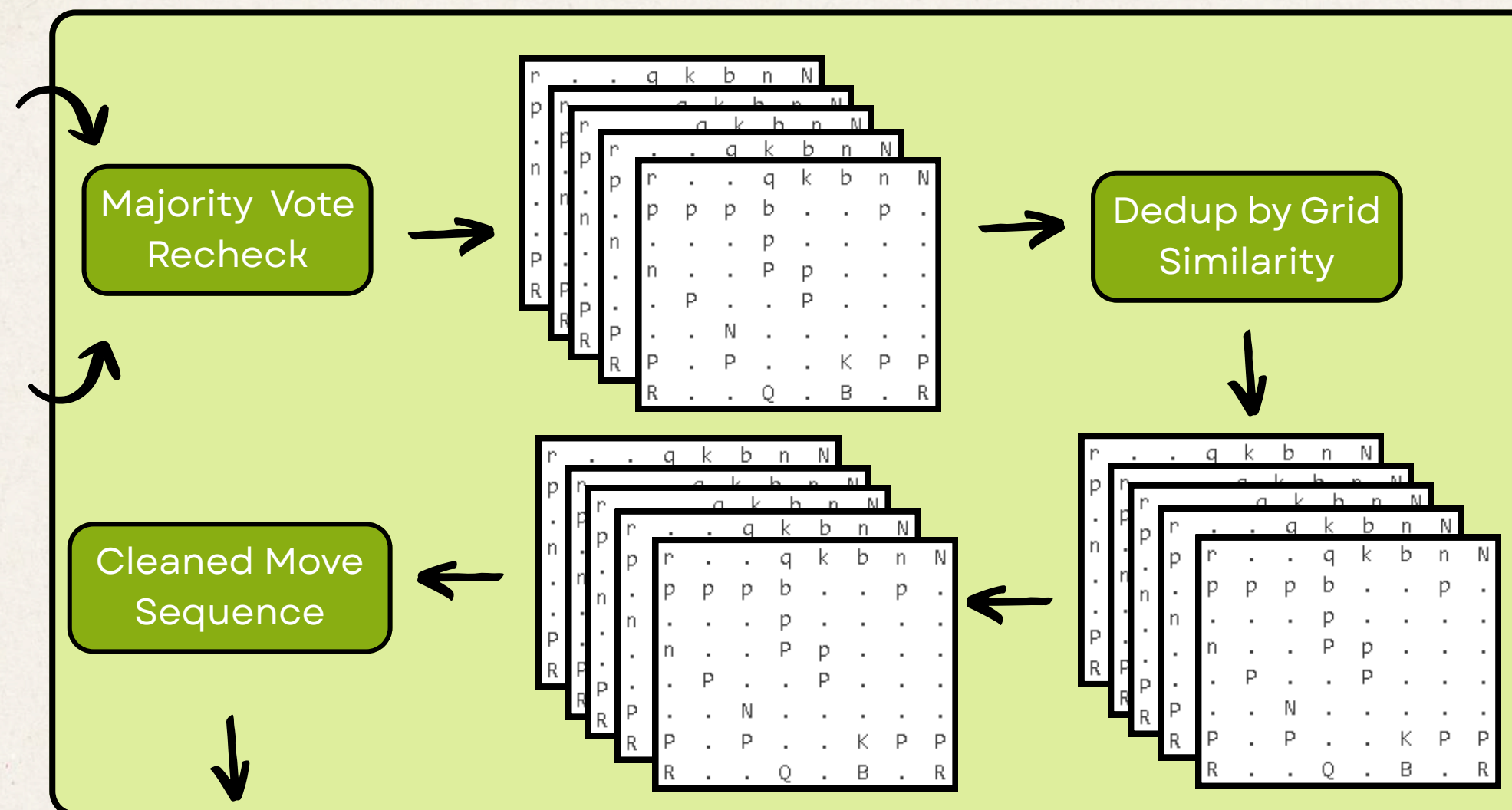
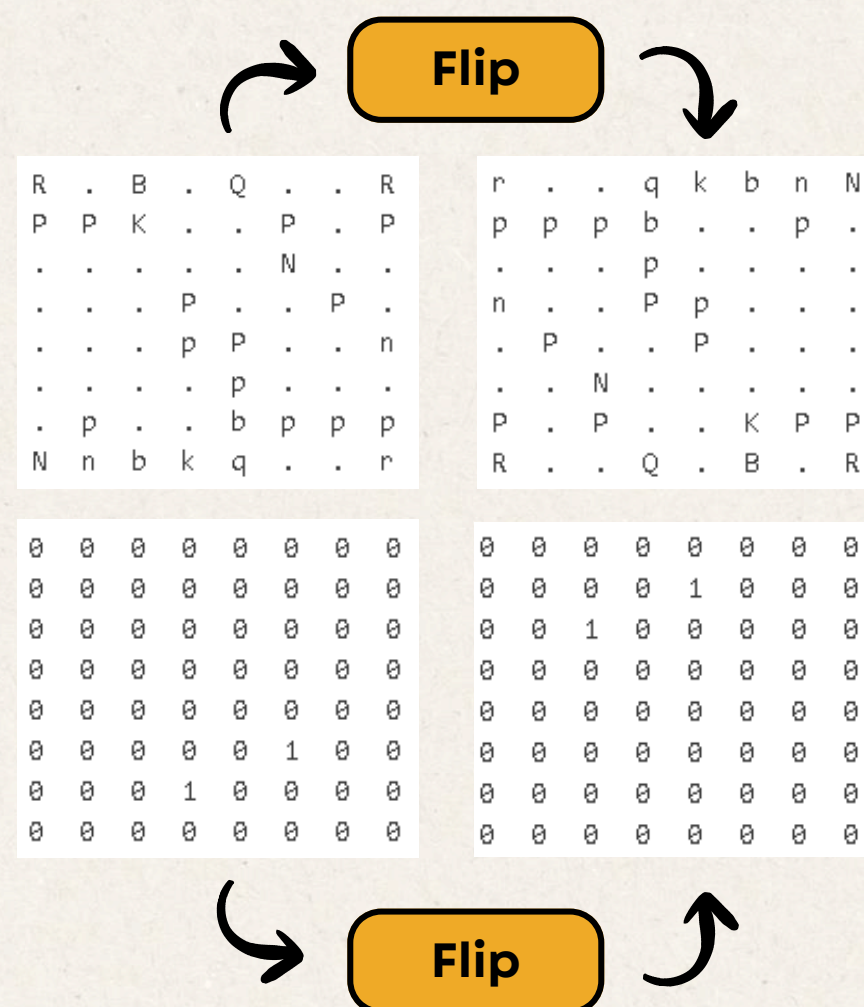
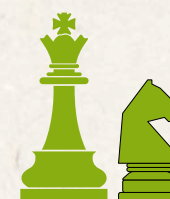
White move



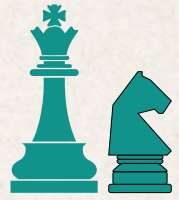
White move (Last)



Pipeline : Post - Processing



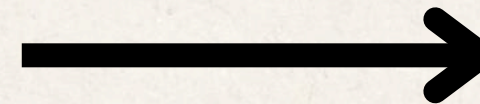
PGN



Finally!!



```
[  
  ['r', '.', '.', 'q', 'k', 'b', 'n', 'N'],  
  ['p', 'p', 'p', 'b', '.', '.', 'p', '.'],  
  ['.', '.', 'p', 'p', '.', '.', '.'],  
  ['n', '.', 'P', 'p', '.', '.', '.'],  
  ['.', 'p', 'N', 'P', 'P', '.', '.'],  
  ['P', 'P', 'P', 'K', 'P', 'P'],  
  ['R', 'Q', 'B', 'R']  
,  
[  
  ['r', '.', '.', 'k', 'b', 'n', 'N'],  
  ['p', 'p', 'p', 'b', '.', '.', 'p', '.'],  
  ['.', 'p', 'p', 'P', 'p', '.', '.'],  
  ['n', 'P', 'P', 'P', 'P', 'q'],  
  ['.', 'N', 'P', 'P', 'K', 'P', 'P'],  
  ['P', 'P', 'P', 'K', 'P', 'P'],  
  ['R', 'Q', 'B', 'R']  
,  
[  
  ['r', '.', '.', 'k', 'b', 'n', 'N'],  
  ['p', 'p', 'p', 'b', '.', '.', 'p', '.'],  
  ['.', 'p', 'p', 'P', 'p', '.', '.'],  
  ['n', 'P', 'P', 'P', 'P', 'q'],  
  ['.', 'N', 'P', 'P', 'P', 'P'],  
  ['P', 'P', 'P', 'K', 'P', 'P'],  
  ['R', 'Q', 'B', 'R']  
]
```



1... Qh4+ 2. e3

PGN format



Evaluation & Analysis

Result

จนจน อย่าลืมทำงาน

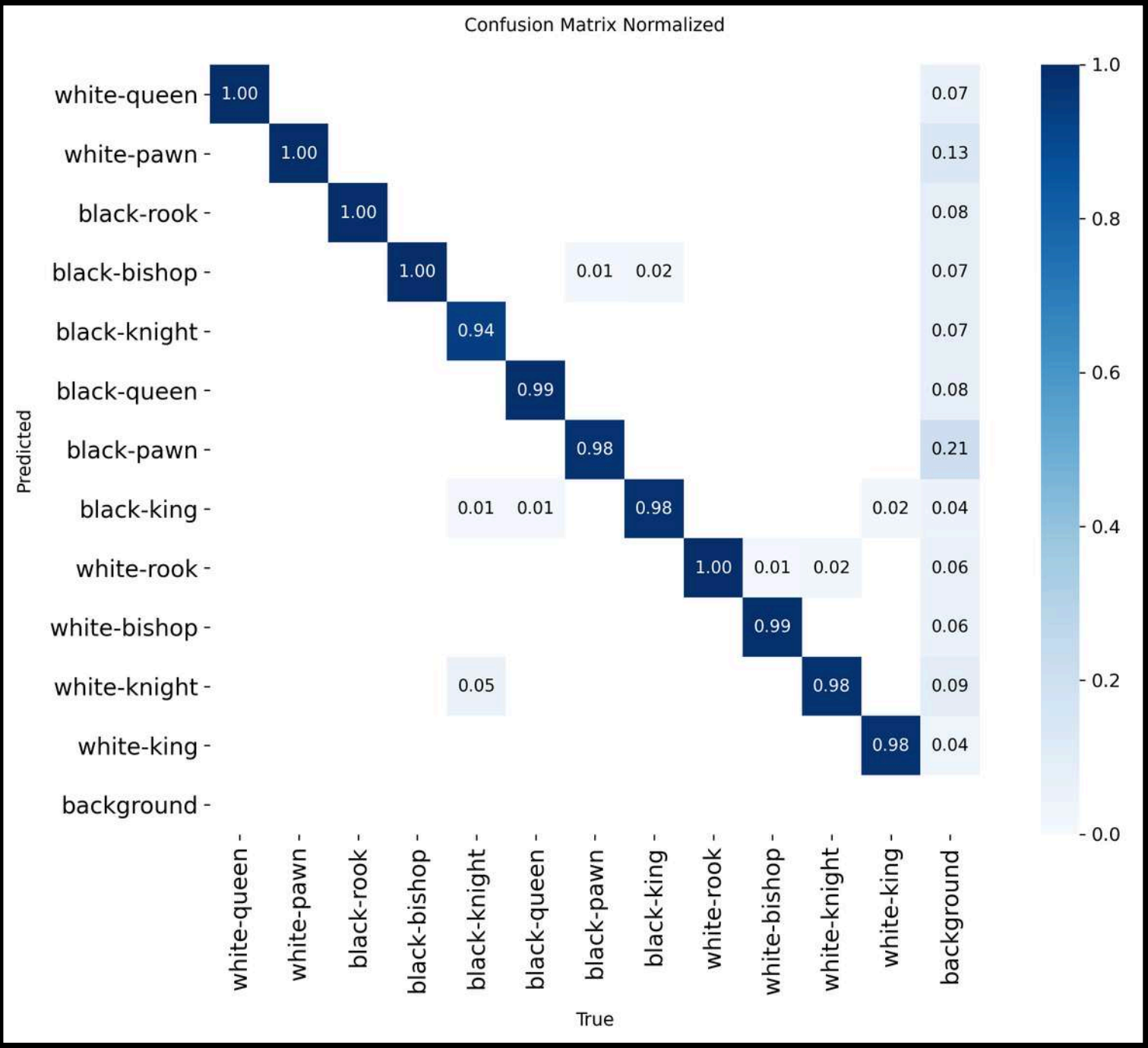
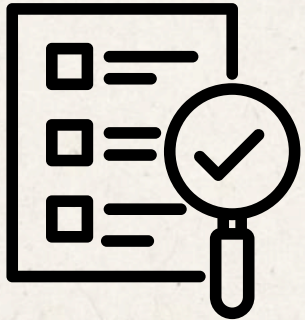


Score !!!

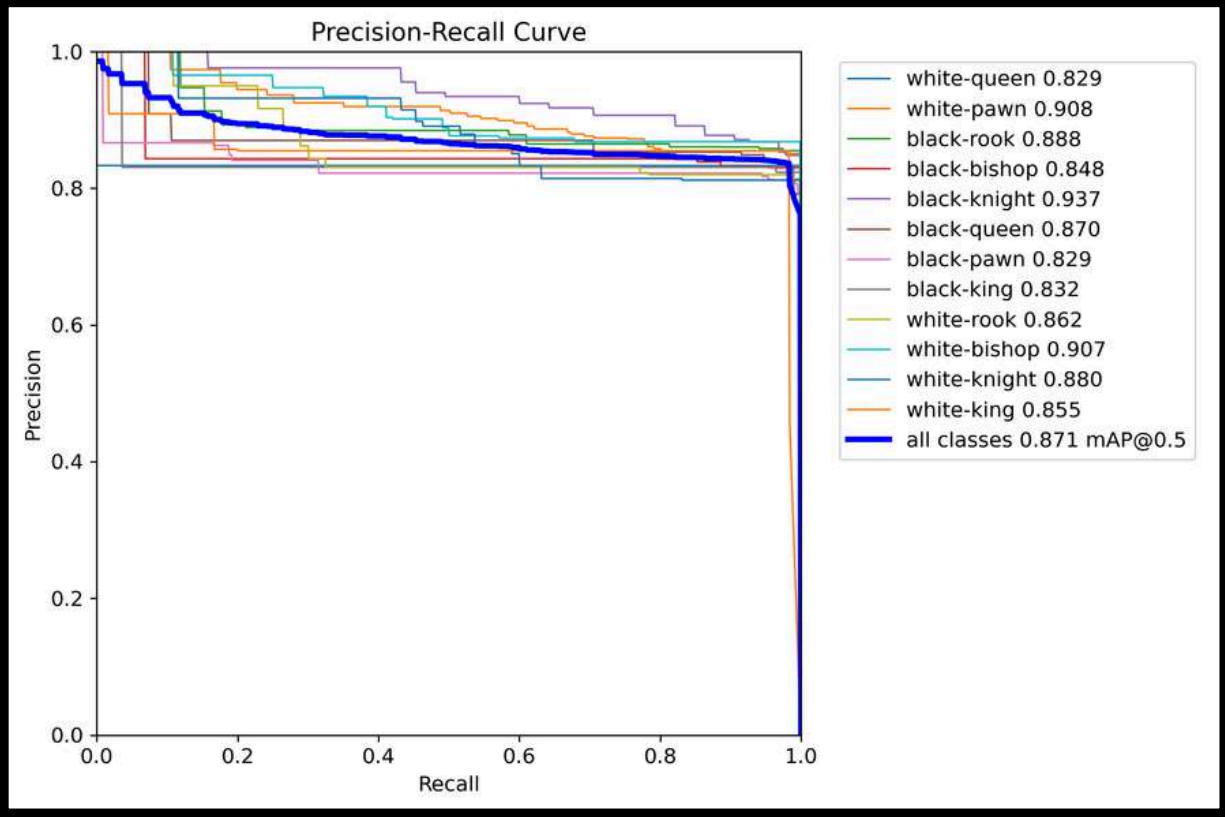
1.00



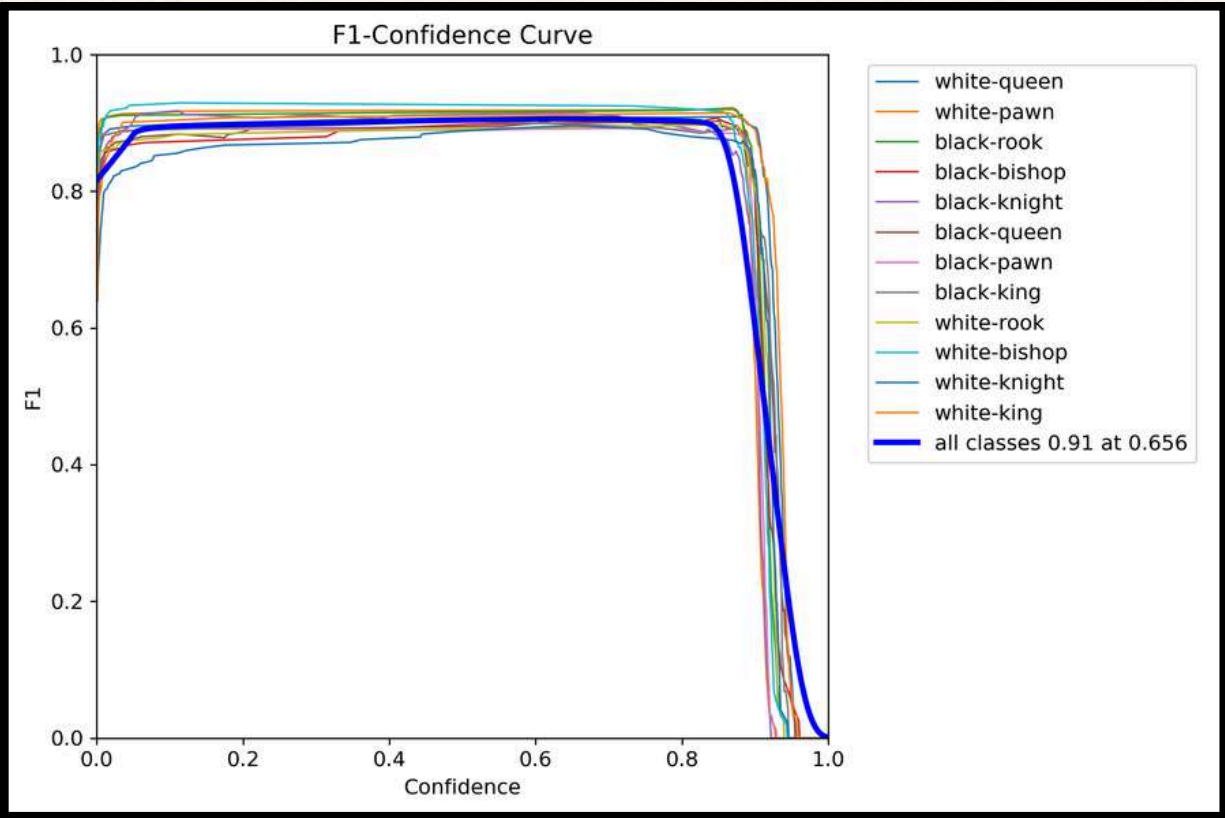
Evaluation



Confusion Matrix

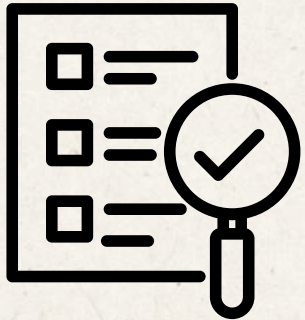


Precision-Recall Curve



F1-Confidence Curve

Evaluation



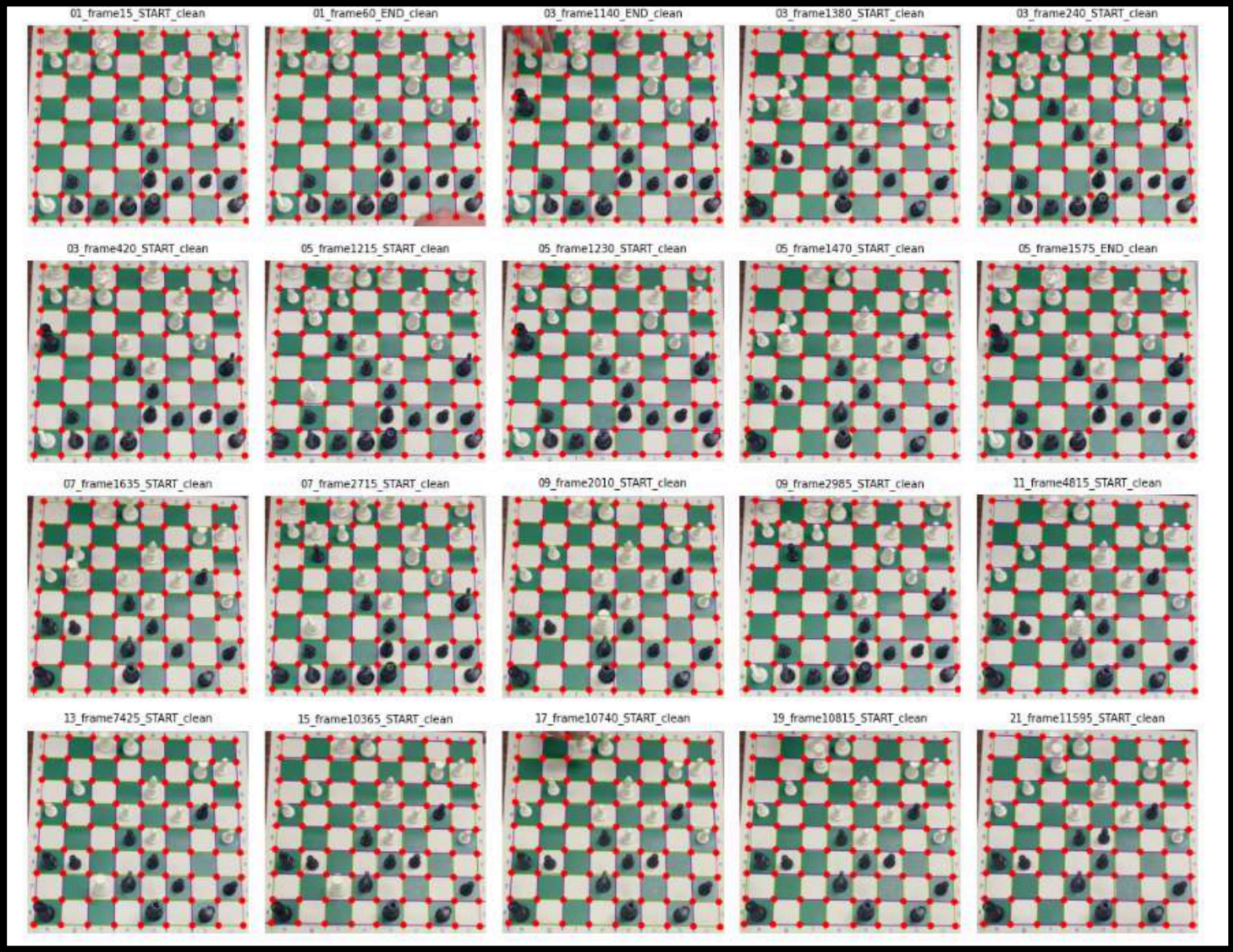
Class	Images	Instances	Box(P	R	mAP50	mAP50-95):
all	69	1338	0.835	0.993	0.871	0.791
white-queen	54	90	0.83	1	0.829	0.767
white-pawn	49	211	0.85	1	0.908	0.808
black-rook	59	118	0.847	1	0.888	0.802
black-bishop	52	88	0.818	1	0.848	0.763
black-knight	58	95	0.868	0.968	0.937	0.833
black-queen	59	95	0.832	0.988	0.87	0.808
black-pawn	47	235	0.81	0.991	0.829	0.74
black-king	56	56	0.822	0.989	0.832	0.753
white-rook	47	83	0.808	1	0.862	0.796
white-bishop	59	112	0.867	0.992	0.907	0.814
white-knight	58	95	0.812	1	0.88	0.803
white-king	59	60	0.851	0.983	0.855	0.801

Speed: 5.7ms preprocess, 23.5ms inference, 0.0ms loss, 5.0ms postprocess per image
Results saved to /content/runs/detect/val

mAP50: 0.870537594161959
mAP50-95: 0.7906270504917717
Precision: 0.834624295974549
Recall: 0.9926229111602661

mAP50: 0.870537594161959
mAP50-95: 0.7906270504917717
Precision: 0.834624295974549
Recall: 0.9926229111602661

Chess detection



Board detection



Analysis



What helps improve our pipeline?

- **Defense in Depth (Hybrid Approach)**
 - **YOLOv8n and Hand Detection:** ensures only clean frames are used.
 - **Model and Chess Ratio Algorithm:** Improves piece-to-cell mapping.
 - **Majority Vote Recheck:** Ensure no-move, no change.
 - **Double Deduplication :** Reduces duplicated frame
- **Diverse training:** Enhances the YOLO model's ability to accurately detect chess pieces under various lighting, perspectives, and piece colors.

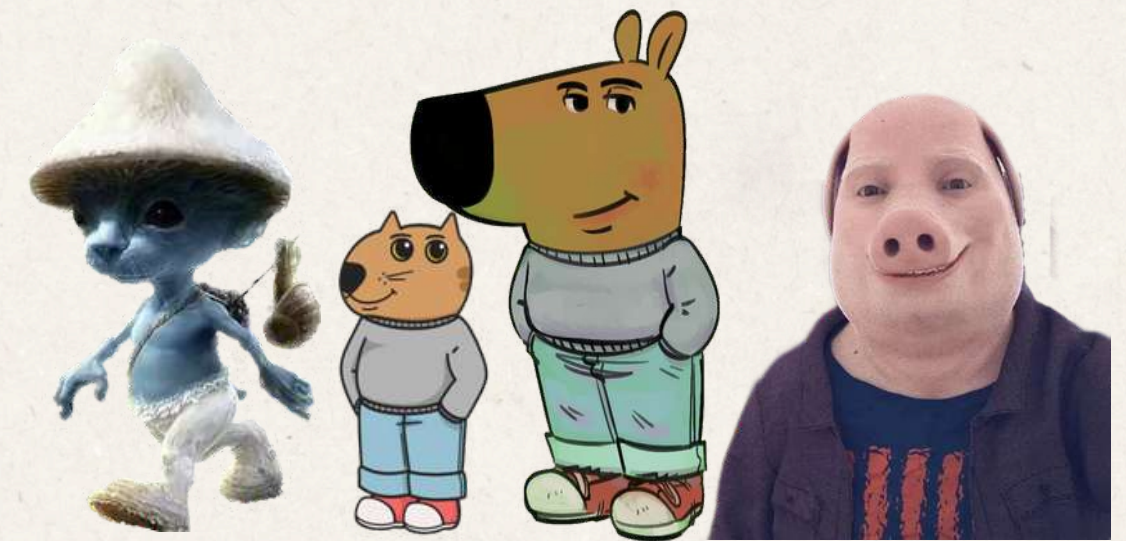
Analysis

Current Weakness and Further Improvement



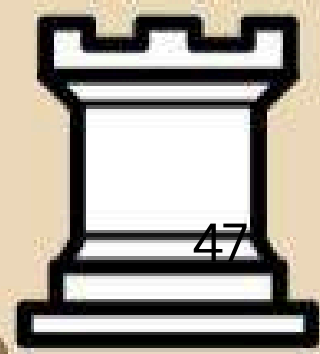
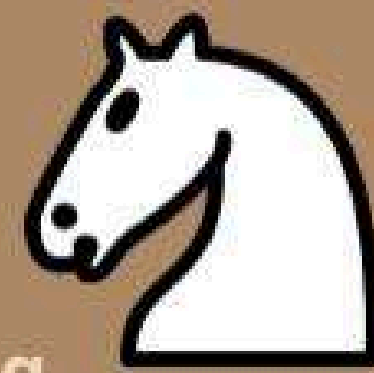
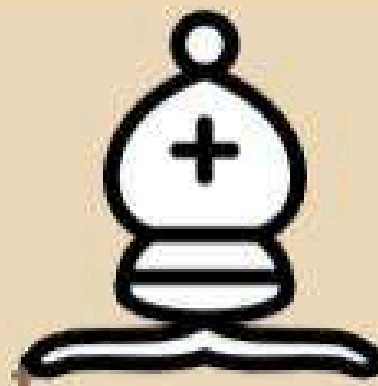
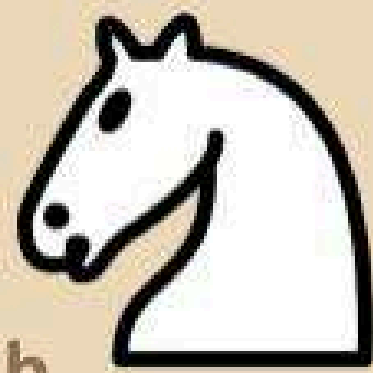
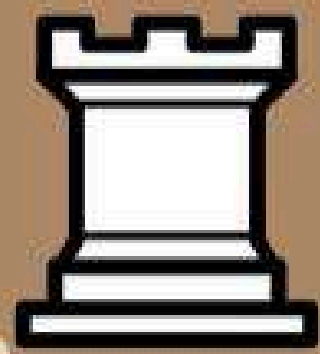
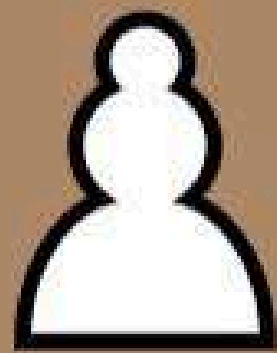
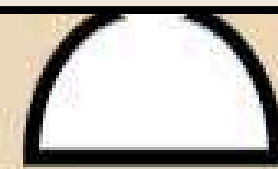
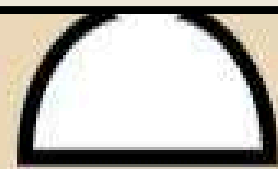
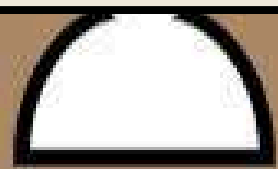
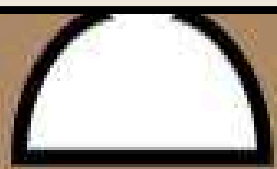
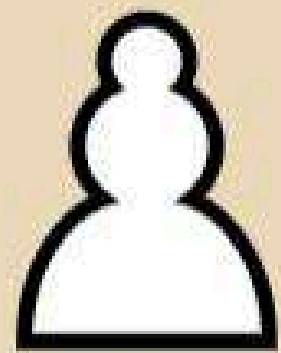
- Color Dependency: The board detection relies entirely on a specific green color mask, failing on other board styles.
 - Sensitive to steep camera angles.
 - Grid fails if corners are not visible.
 - **Sol** : Replace with a trained YOLO model to identify the board regardless of color or style.
- Orientation Weakness: The board rotation algorithm is too simple and will fail in specific endgame scenarios.
 - **Sol** : Use OCR to detect board Algebraic Notation.
- Logic Gaps: The hand-crafted PGN translator might not cover every case.
 - **Sol** : Use official pgn translator in python-chess.

Role Description



กิตติพัฒน์ จูสิงห์ 6531304621	ROI, Board, Chess Detection, Evaluation
ณรรณกฤษ ธีระรัชชานนท์ 6531315521	Chess Model Training, Slide forPresentation
รัชปศย์ แบนประเสริฐ 6531338021	Data Preparation, Slide forPresentation
อชิระ ลีน 6532196321	Everything

Thank you for your attention.



47

