REAL-TIME CHESS GAME ANALYZER

Serden Sait Eranil

Veysel Colak

Firdevs Su Aydin

Outline

- Motivation
- Project Steps & Work Distribution
- Project Pipeline (ROS Nodes & Topics)
- Deeper Look Into Project Steps
- Accuracy and Delay Results
- Demo
- Future Scope

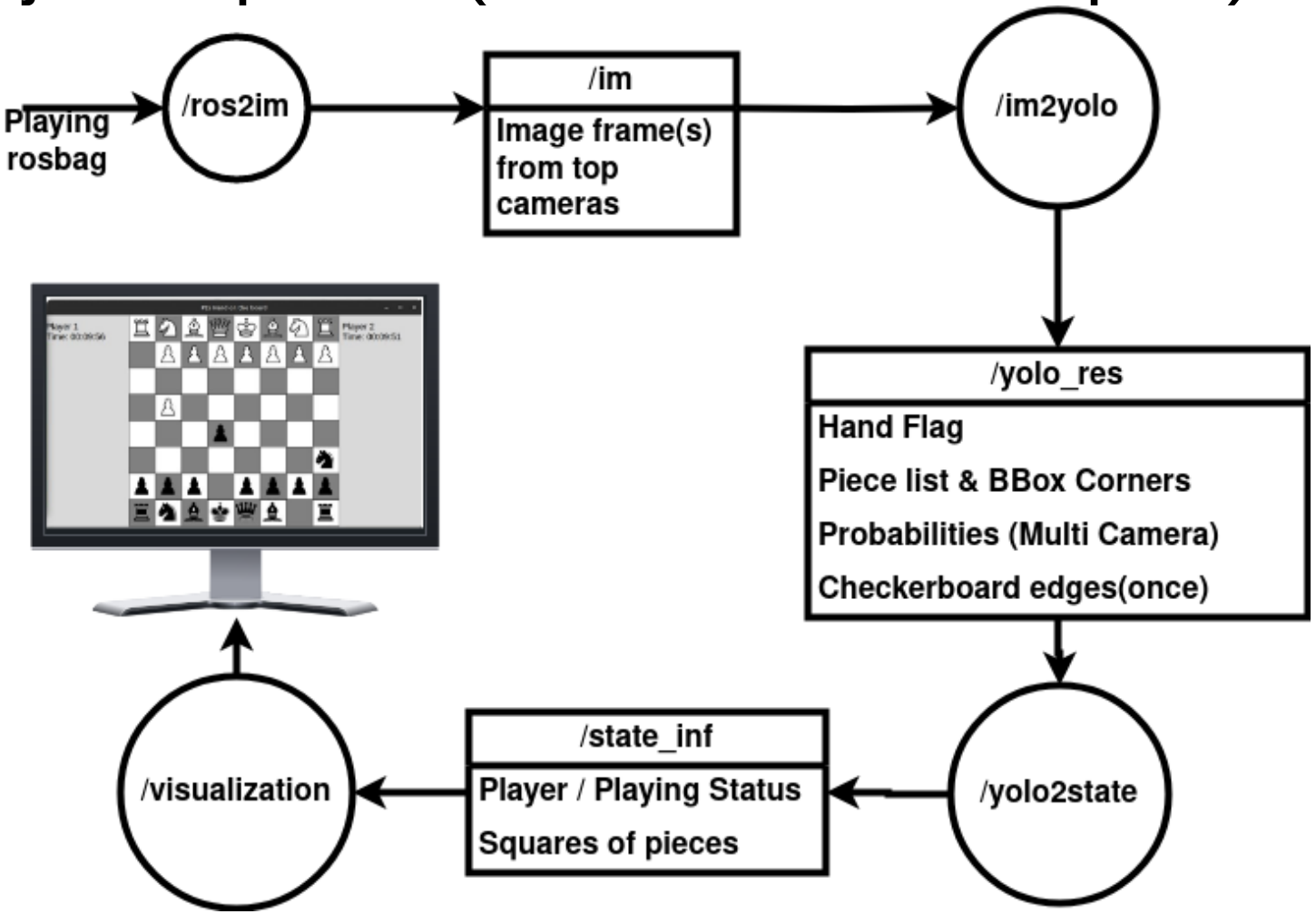
Motivation

- Automate recording of the moves
- Instead understand the moves with camera and save them

Project Steps & Work Distribution

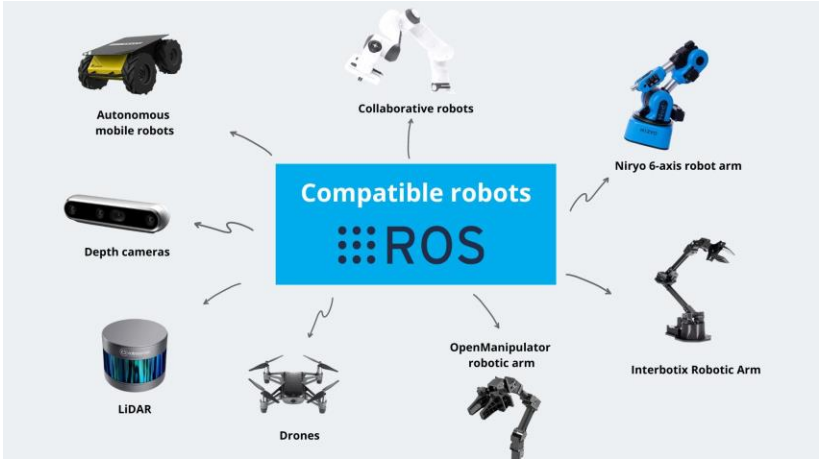
Data Collection & Annotation	YOLO Model Selection	Piece, Checkerboard and Hand Detection
Serden Su Veysel	Su Serden	Su Serden
Game State Management	**Visualization**	**ROS Pipeline & Workspace**
Veysel	Serden	Veysel

Project Pipeline (ROS Nodes & Topics)



Deeper Look Into Project Steps

ROS



- Robot Operating System is…
  - a robotics workframe.
  - a library for C and Python.
  - a middleware between distributed system.

- "Nodes", "Topics", "Services", "Actions" …

What does ROS bring?

- The "communicating nodes" structure

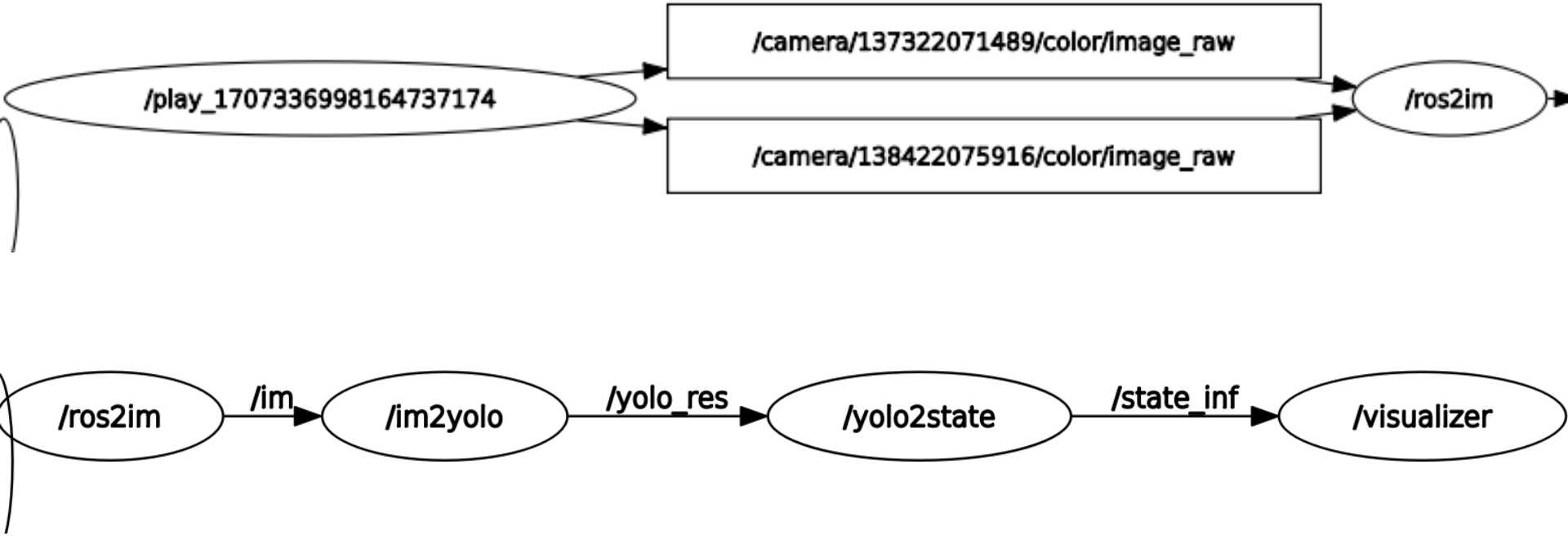
- In robotics domain also...

But there are two of them!

- ROS has two versions: ROS & ROS2.
  - No Masters
  - Message QoS profiling

- Due to lab computers and bag files, we have moved our project to ROS1

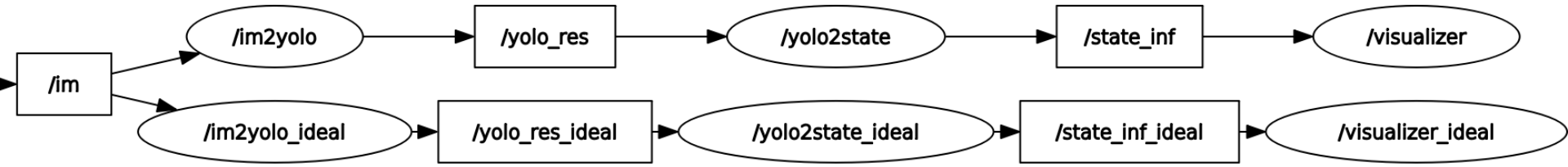
ROS Implementation (with Camera Node)



ROS Implementation (with Test Node)

- Utilize the ROS node2node communication

- On parallel use a "~perfectly arranged determiner"
  - Use the branch for each step after "/im" topic



ROS Reading Camera (/ros2im)

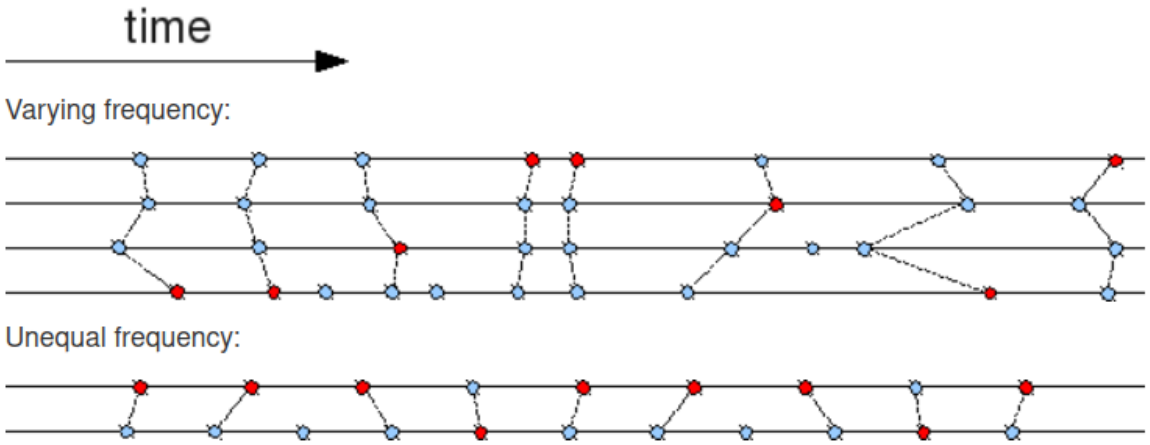
- Other nodes subscribe 1 topic
  - But /ros2im subscribes 2 topics
  - They should be combined.

- Approximate Time Synchroniser

Time Synchronization in ROS

- TimeSynchronizer:
  - Perfect but Unimplementable

- ApproximateTimeSynchronizer:
  - Frame loss possible but works



DATA ANNOTATION TRAINING AND TEST

Data Collection and Annotation

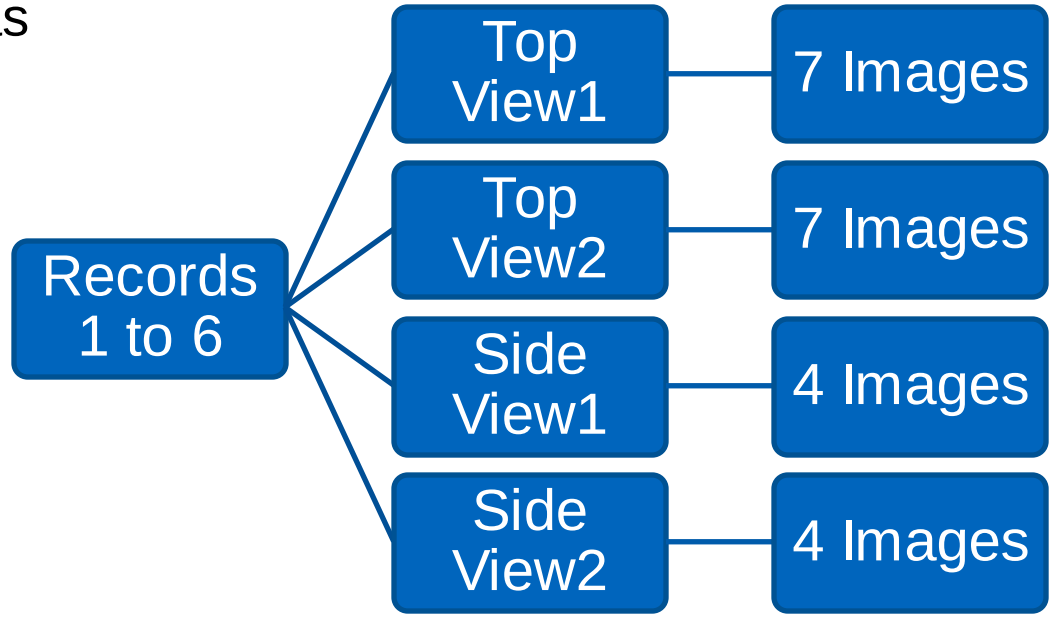
- 8 Recordings
- 4 Cameras: 2 top views, 2 side views




Data Collection and Annotation

Training Dataset:

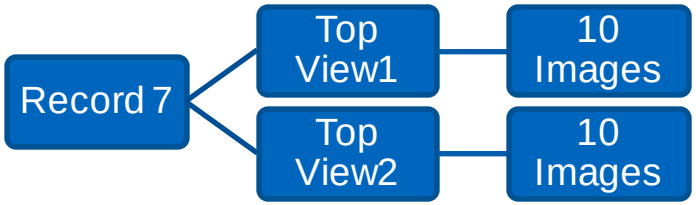
- 6 recordings, 4 cameras
- Total of 132 training images



Data Collection and Annotation

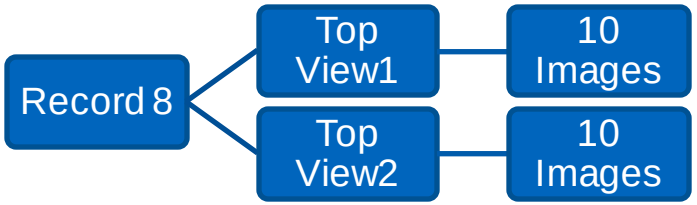
Validation Dataset:

- 1 recording, 2 top views
- Total of 20 validation images



Test Dataset:

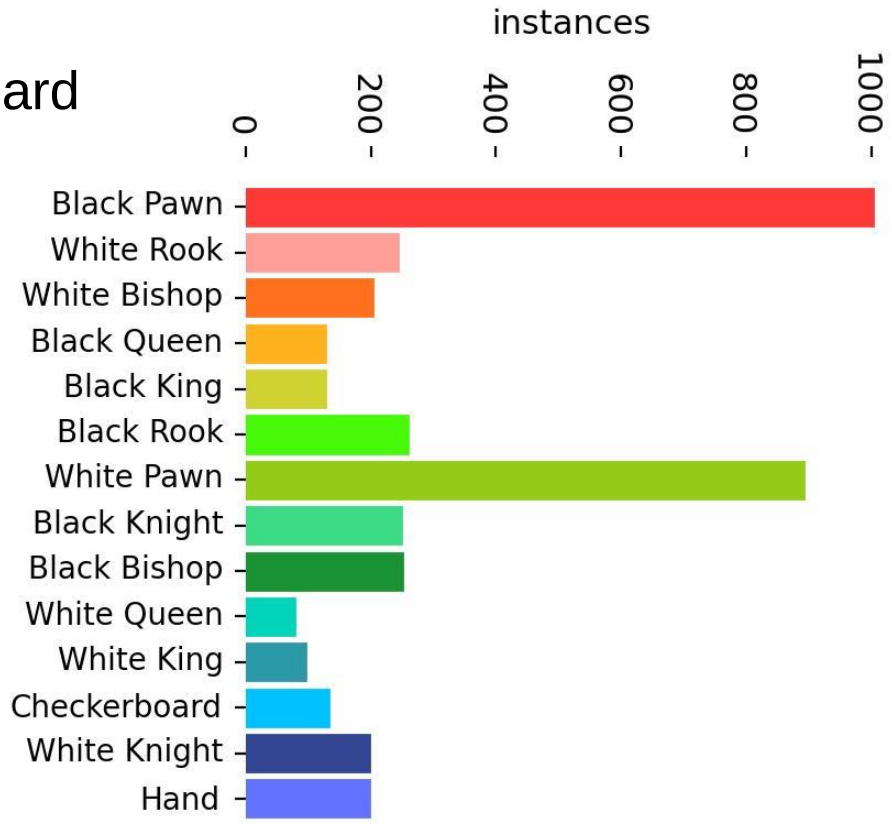
- 1 recording, 2 top views
- Total of 20 test images



Data Collection and Annotation

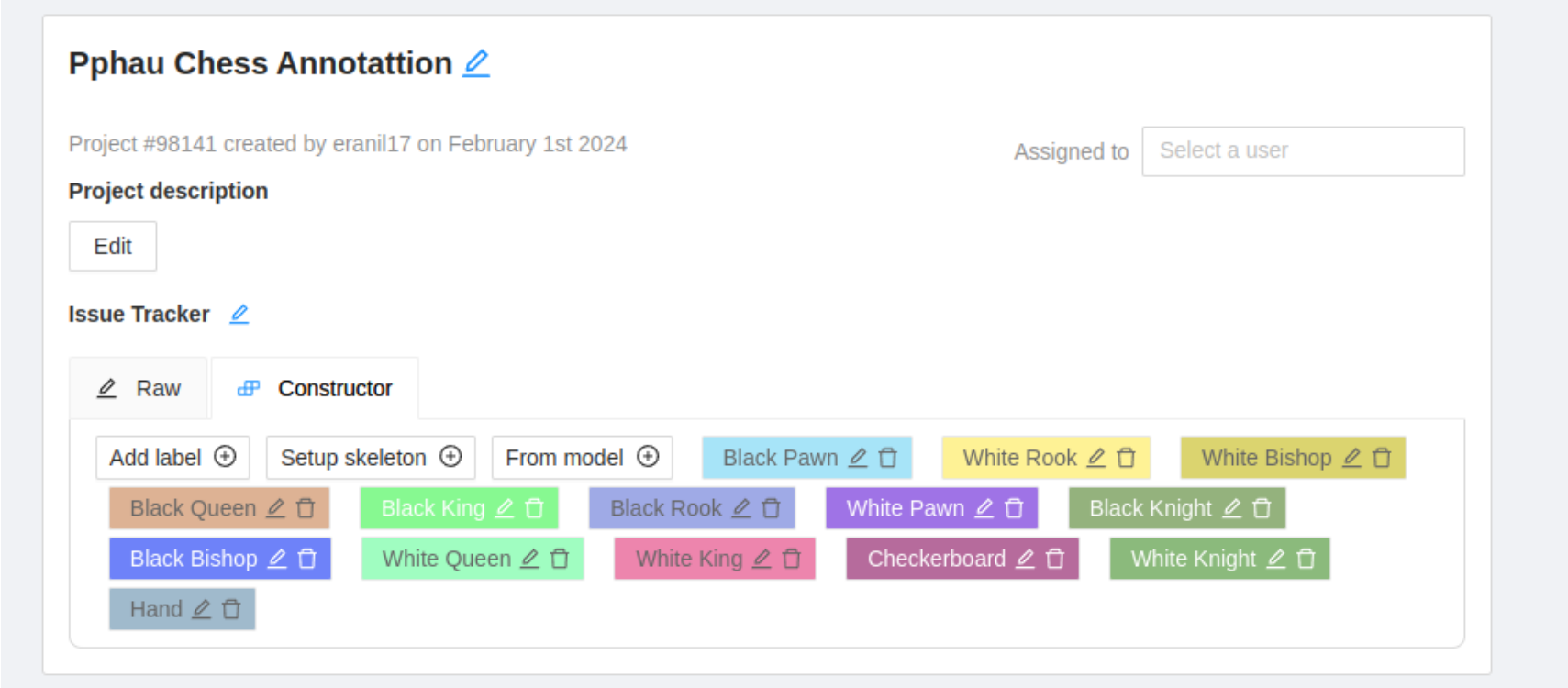
Data Distribution:

- 12 piece + hand + checkerboard
- Not evenly distributed



CVAT: Computer Vision Annotation Tool

- Easy-to-use, open-source, collaborative web-based tool
- Allows the use of state-of-the-art segmentation methods (e.g., Segment Anything, HRNET)

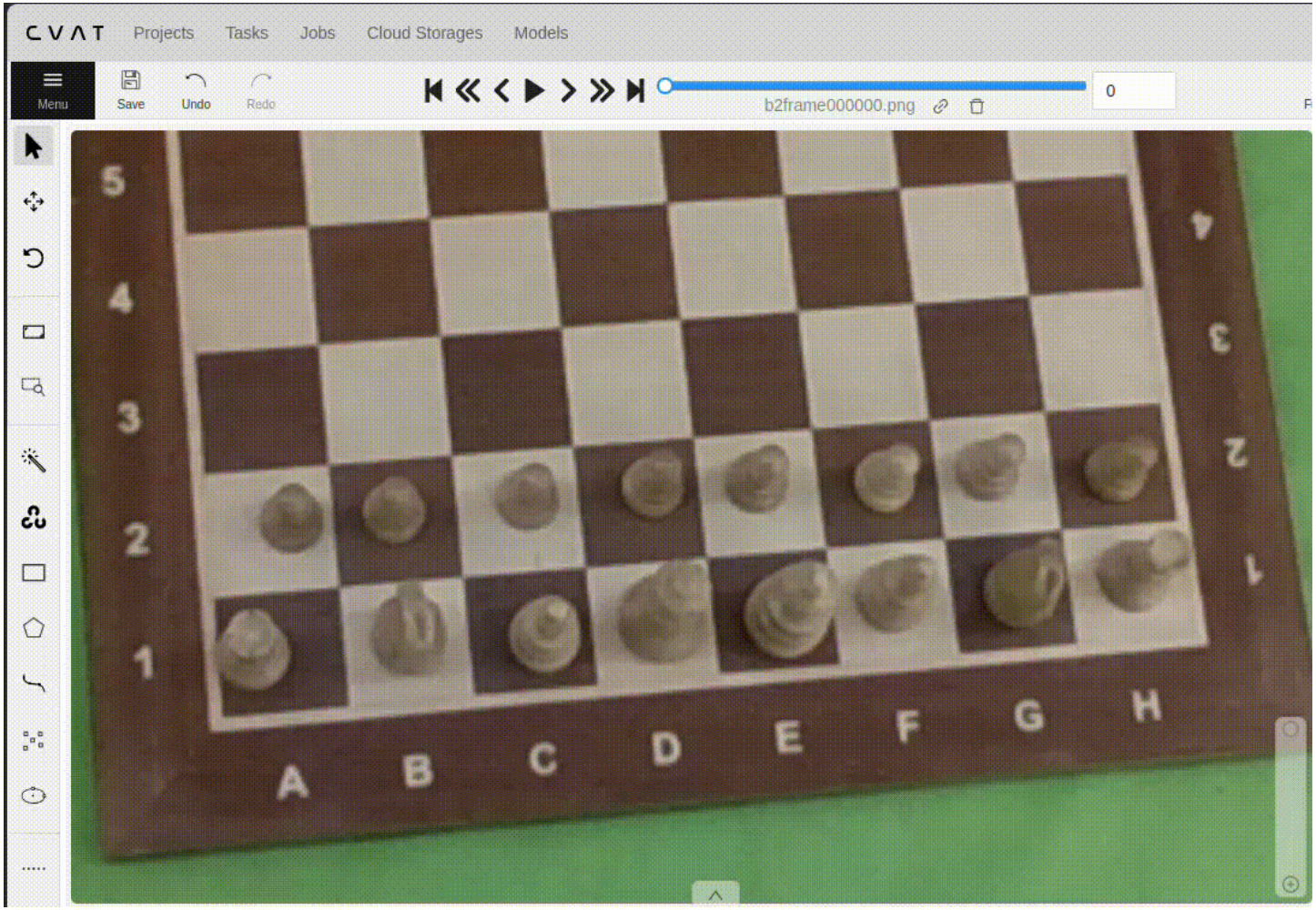


Why didn't we use MIVOS?

- Poor quality of the pieces
- Insufficient support of instances
- Fast hand movements
- Too chaotic piece distribution

Solution: Through use of CVAT, add some manual input

How did we do the annotation in CVAT: GIF



Final annotation outcome per frame

YOLO MODELS

YOLO Model Selection

- 4 sizes of YOLO v8 model:
  - Nano
  - Small
  - Medium
  - Large

Model	size (pixels)	mAP^{val} 50-95	Speed CPU ONNX (ms)	Speed A100 TensorRT (ms)	params (M)	FLOPs (B)
YOLOv8n	640	37.3	80.4	0.99	3.2	8.7
YOLOv8s	640	44.9	128.4	1.20	11.2	28.6
YOLOv8m	640	50.2	234.7	1.83	25.9	78.9
YOLOv8l	640	52.9	375.2	2.39	43.7	165.2

Figure: Parameters of YOLO models. Adapted from https://github.com/ultralytics/ultralytics

YOLO Model Selection

Training:

- Avoid overfitting.

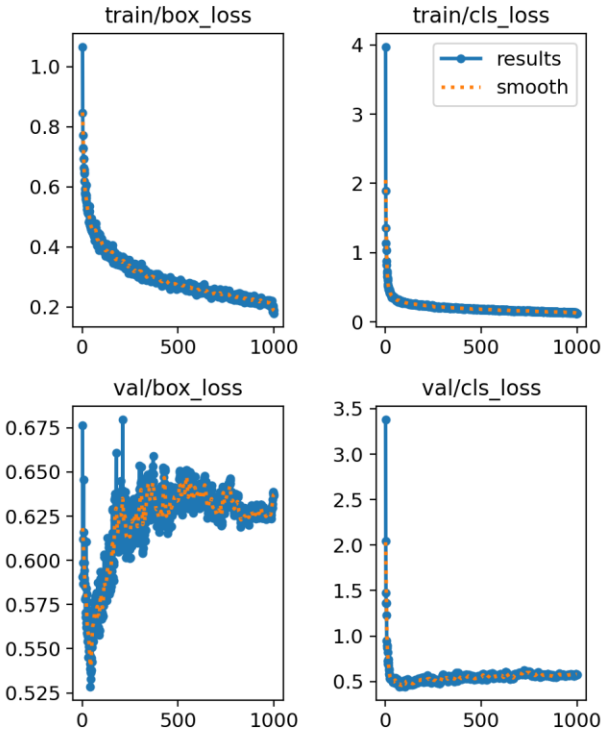


Figure1: Training curves for small model.

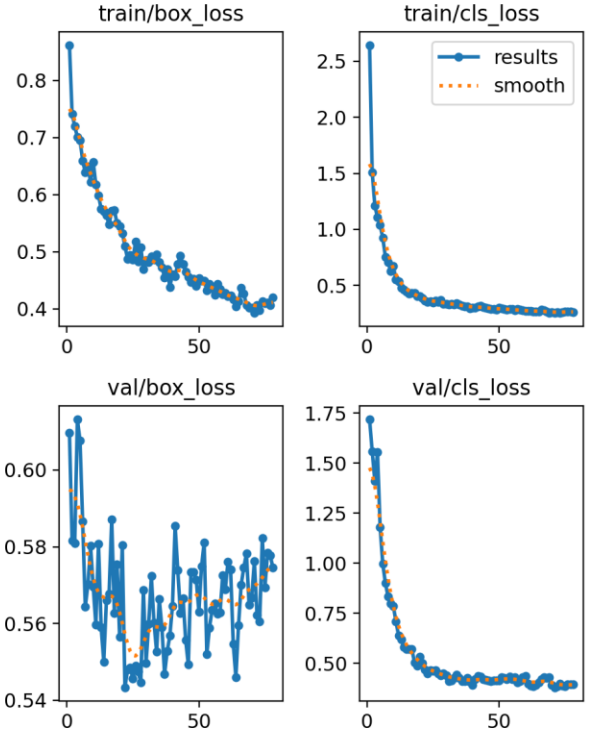


Figure2: Training curves for medium model.

YOLO Model Selection

Test: Confusion matrices

- Some pieces are confused with each other.

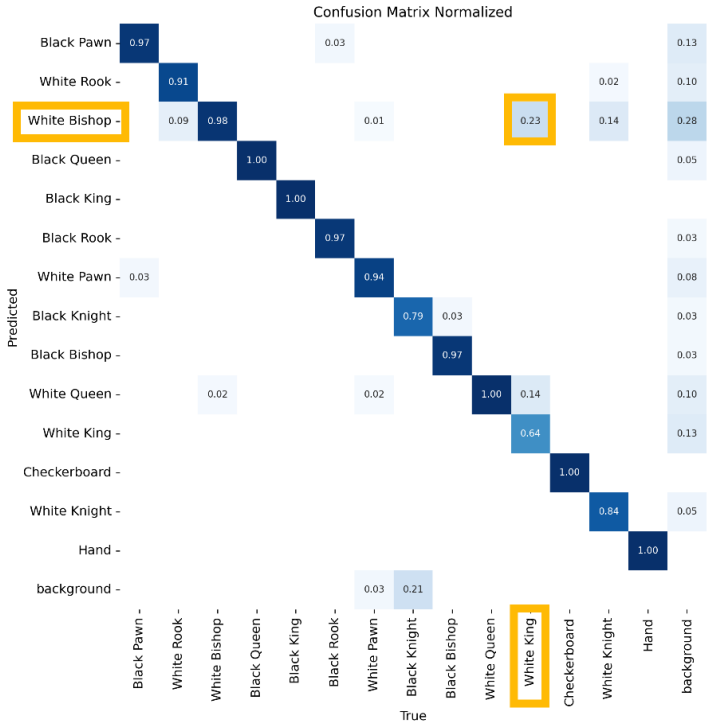


Figure 1: Confusion matrix for large model with test data

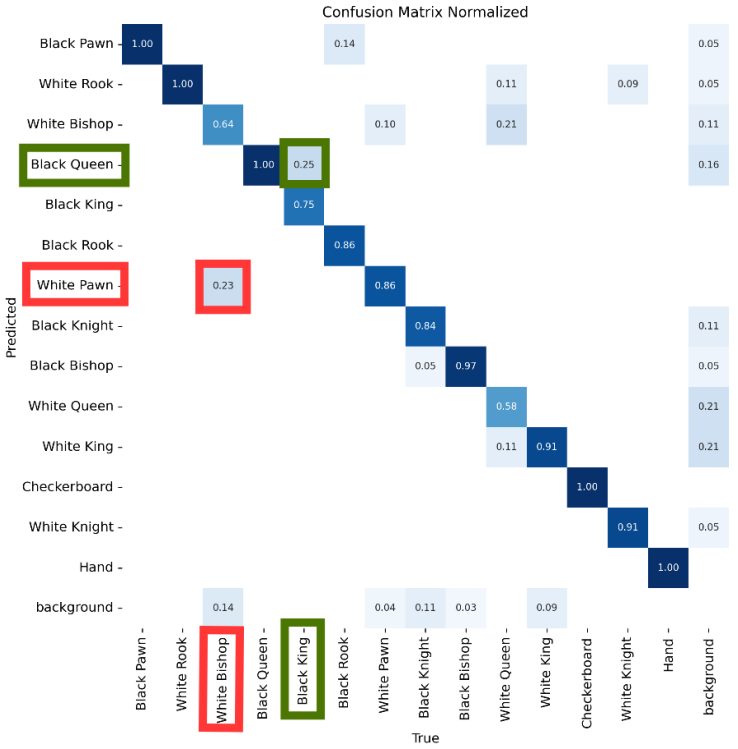


Figure 2: Confusion matrix for small model with test data

YOLO Model Selection

Test: F1 Scores

- Kings and Queens are harder to detect.

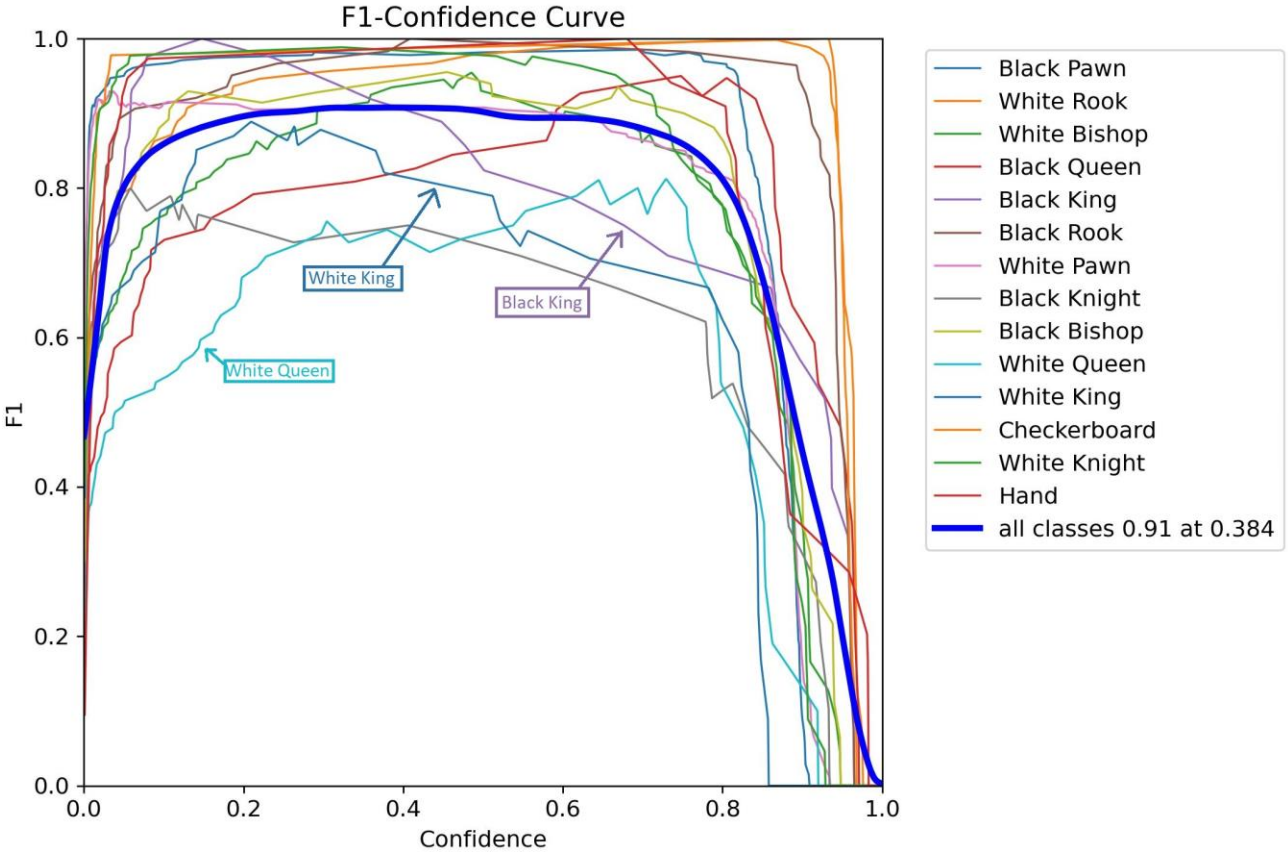
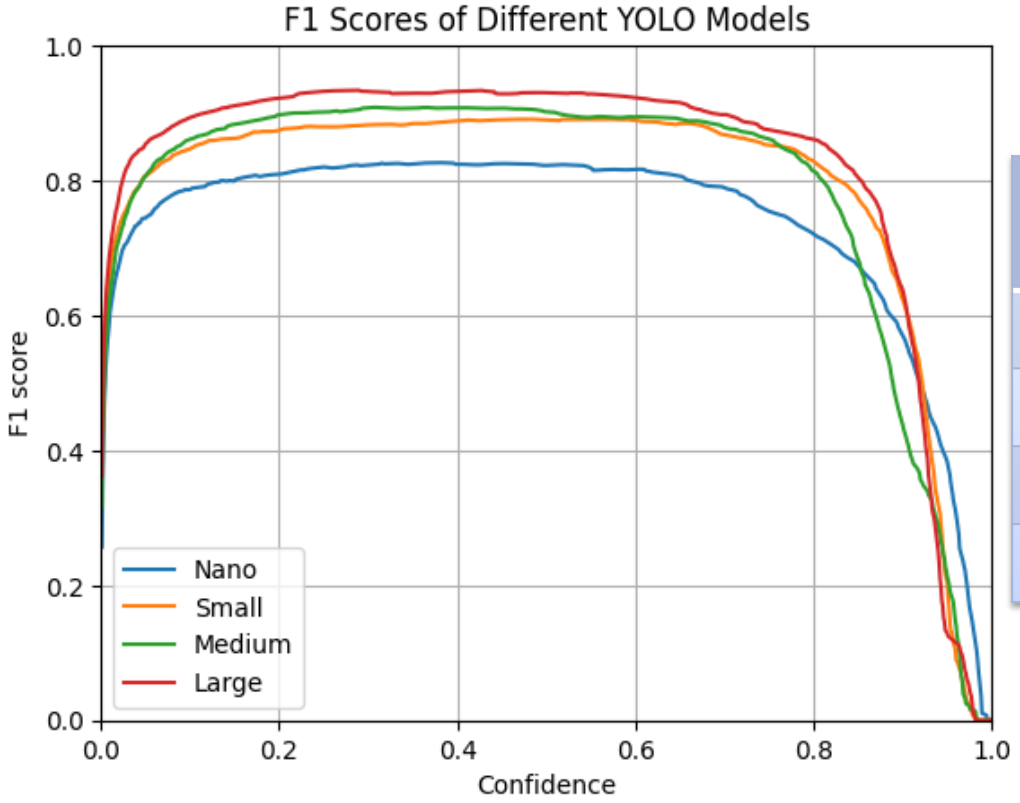


Figure: F1 Score of medium model

YOLO Model Selection

Test: F1 Scores

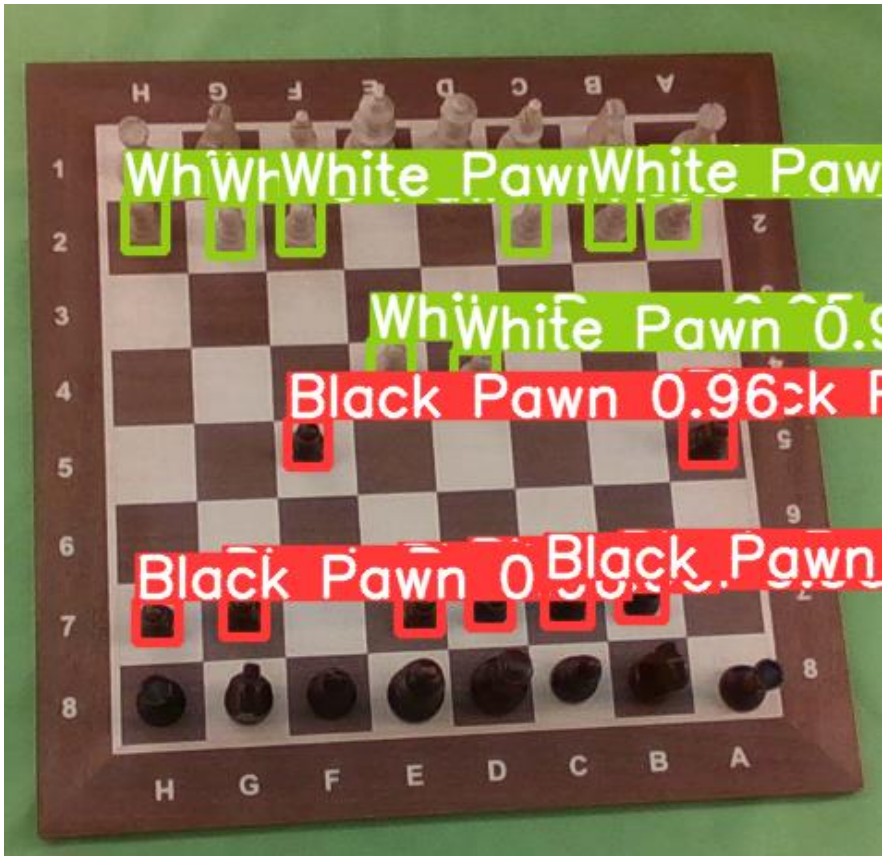
- Confidence threshold is determined by F1 Score.
- Medium model is selected.



Model	Max F1 S.	Confidence
Nano	0.81	0.374
Small	0.89	0.553
Medium	0.91	0.384
Large	0.93	0.274

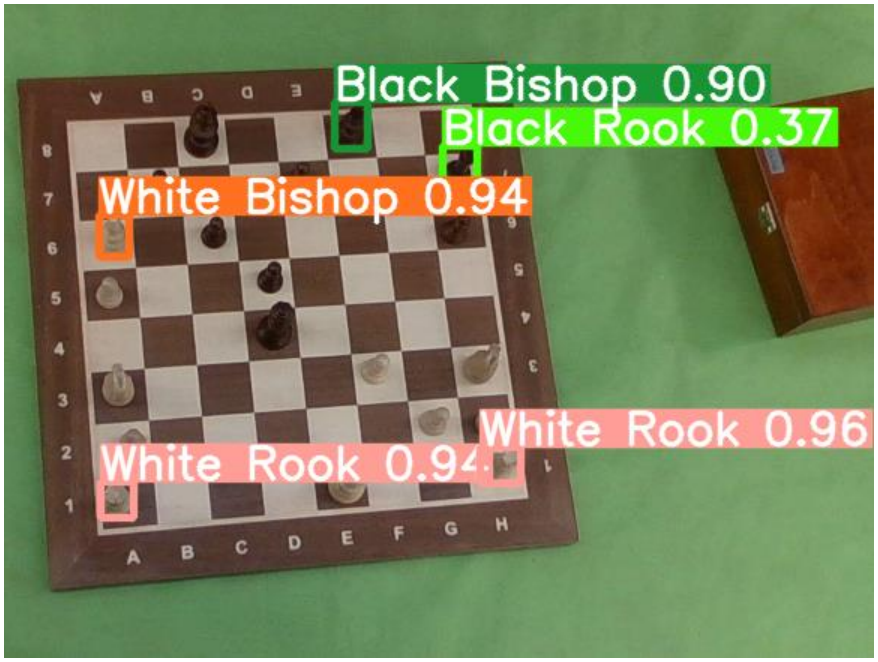
YOLO Model Selection

- Some detection examples: Only pawns



YOLO Model Selection

- Some detection examples: Rooks and Bishops

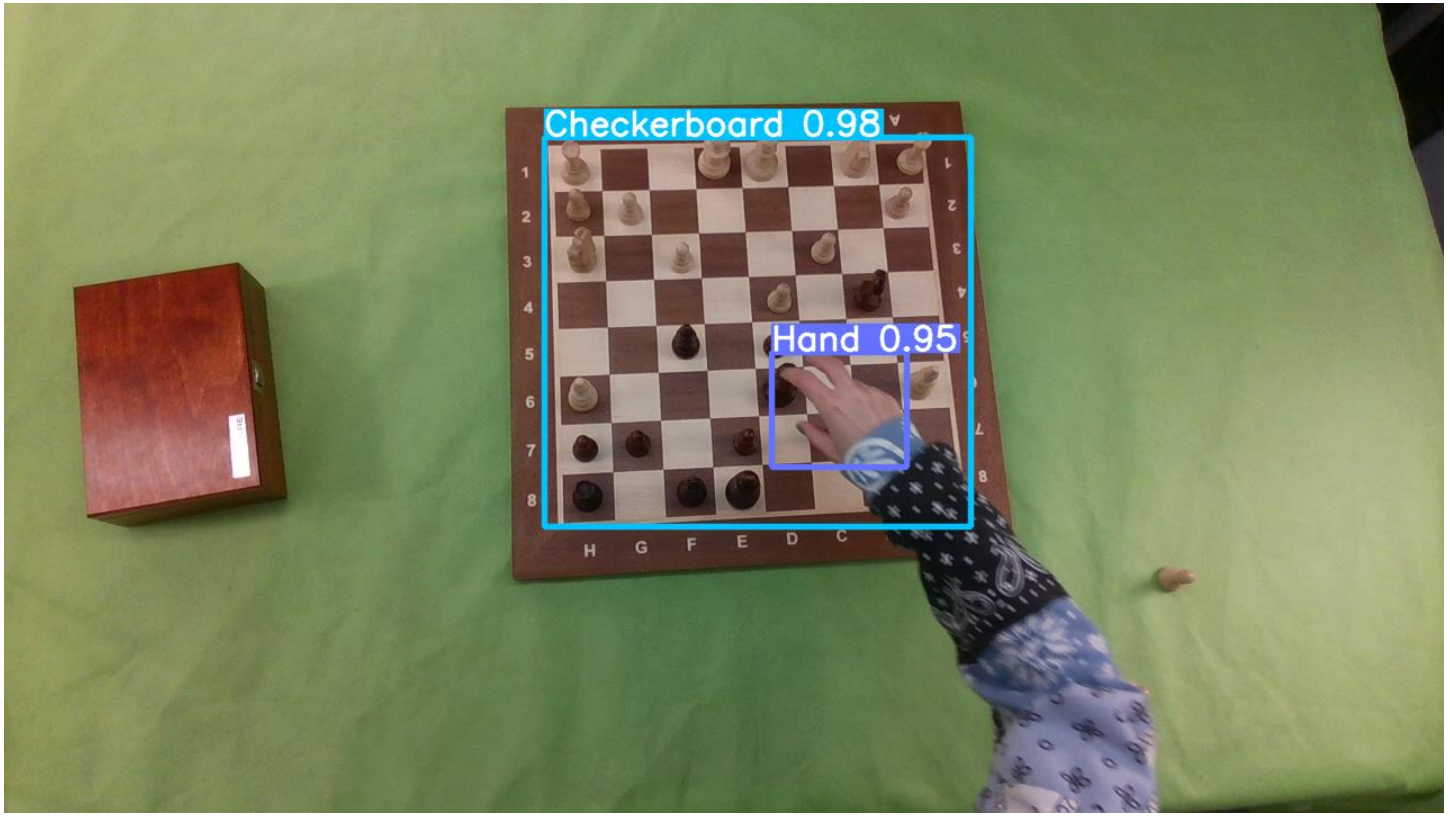


YOLO Model Selection

- Some detection examples: Knights, Kings and Queens

YOLO Model Selection

- Some detection examples: Checkerboard and Hands



YOLO Model Selection

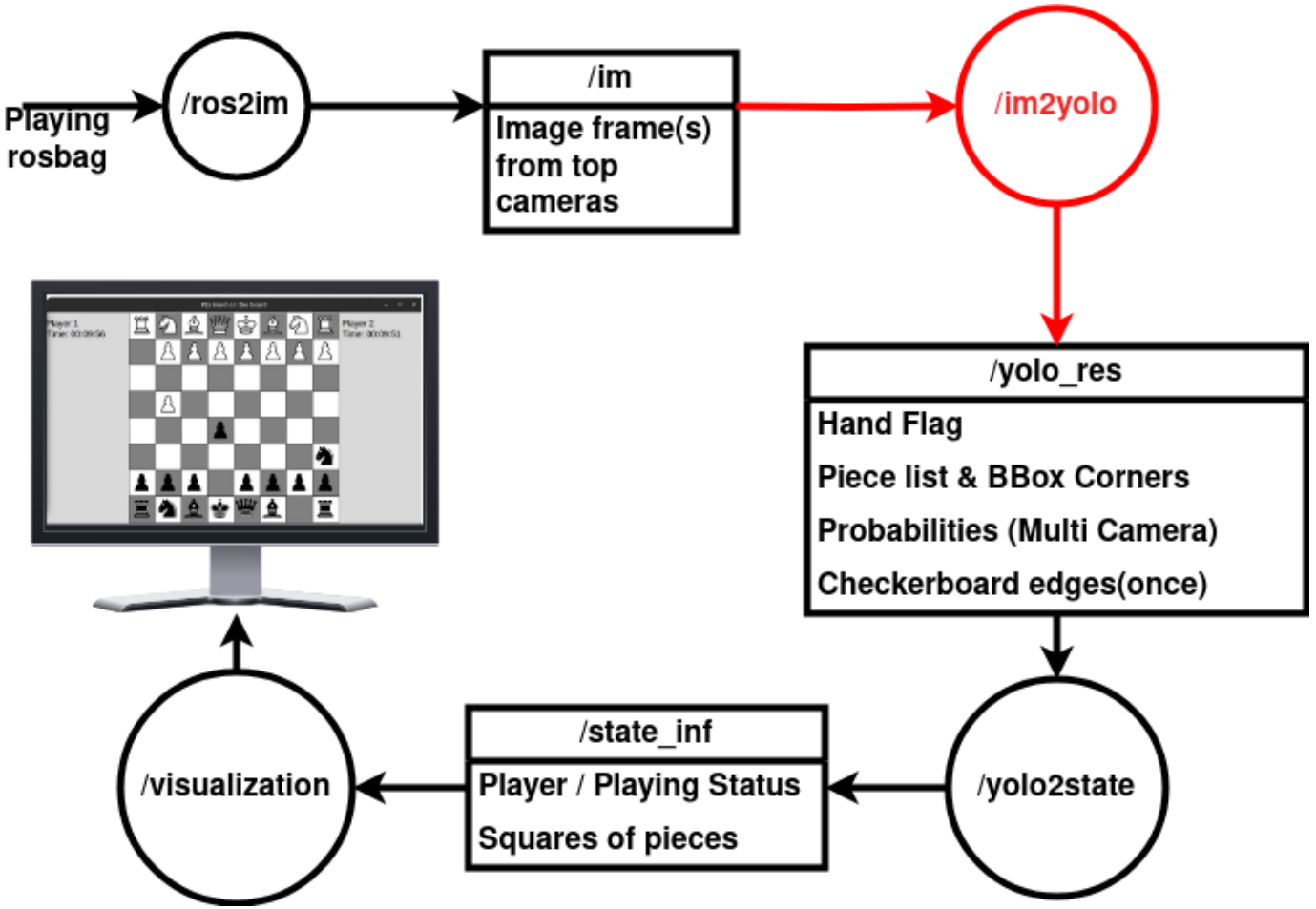
Detection Challenges:

- White bishops are hard to detect.
- Queens and kings are sometimes confused for each other.

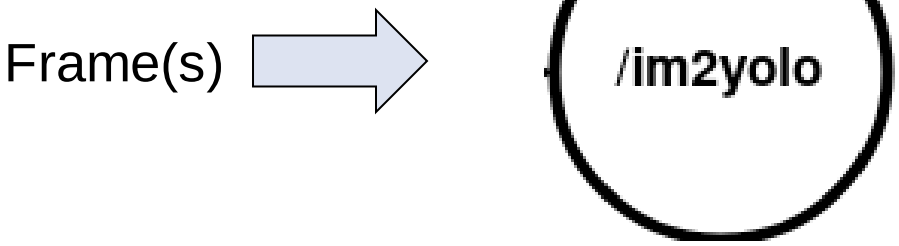
Figure 1: Wrongly detected white queen

Figure 2: Undetected white bishop

Piece, Checkerboard and Hand Detection

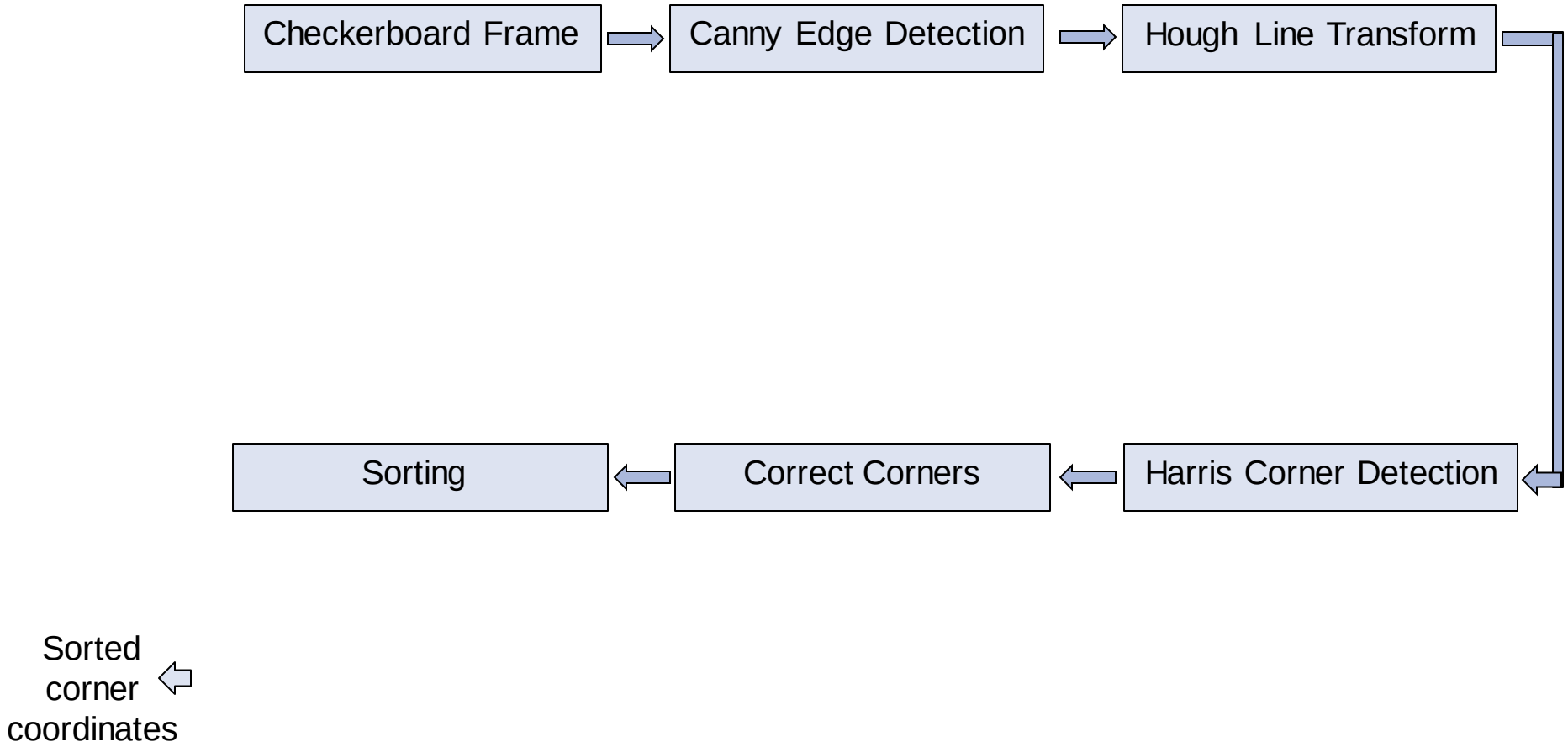


Piece, Checkerboard and Hand Detection

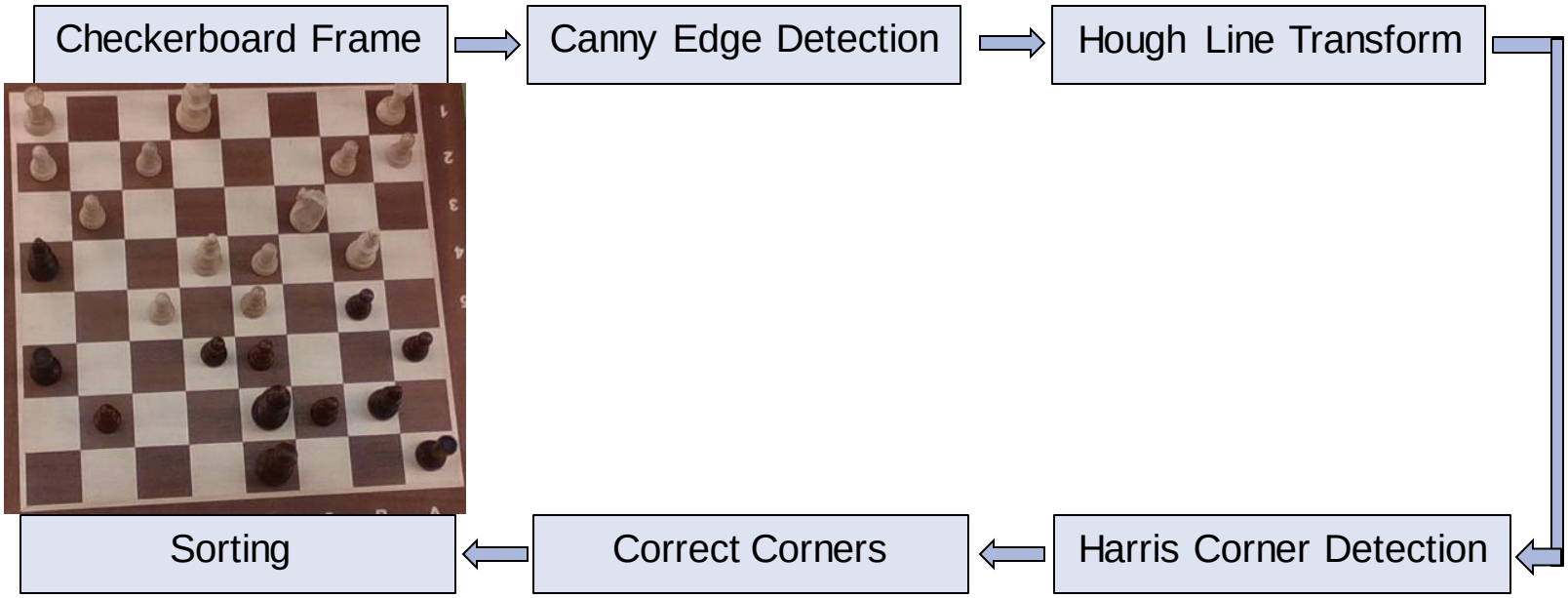


- Checkerboard coordinates
- Hand flag
- Piece classes
- Piece bounding boxes
- Piece probabilities

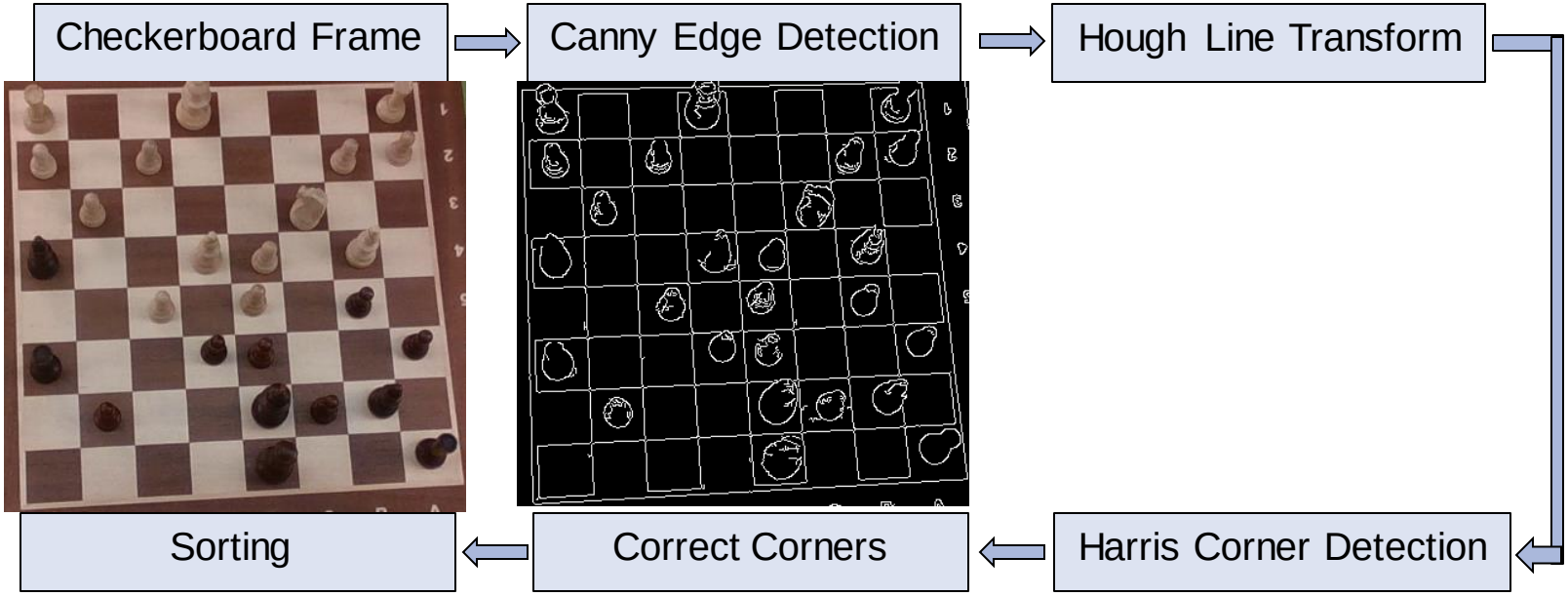
Checkerboard Corner Detection



Checkerboard Corner Detection

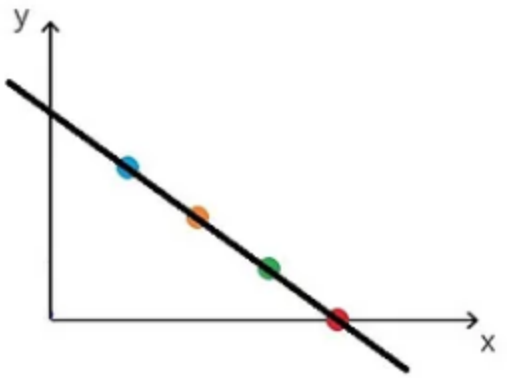


Checkerboard Corner Detection



Checkerboard Corner Detection

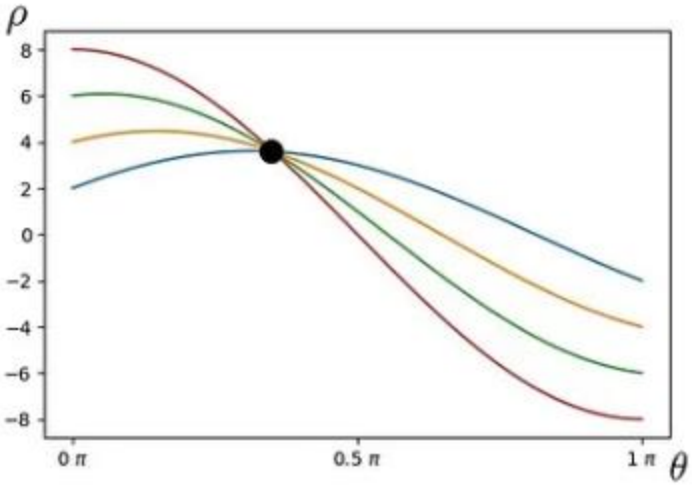
- Hough-Line Transform:
  - Point -> Sinusoid
  - Intersection of sinusoids means line



Points which form a line

Figure 1: Line on image domain. Adapted from https://shorturl.at/lBOX5

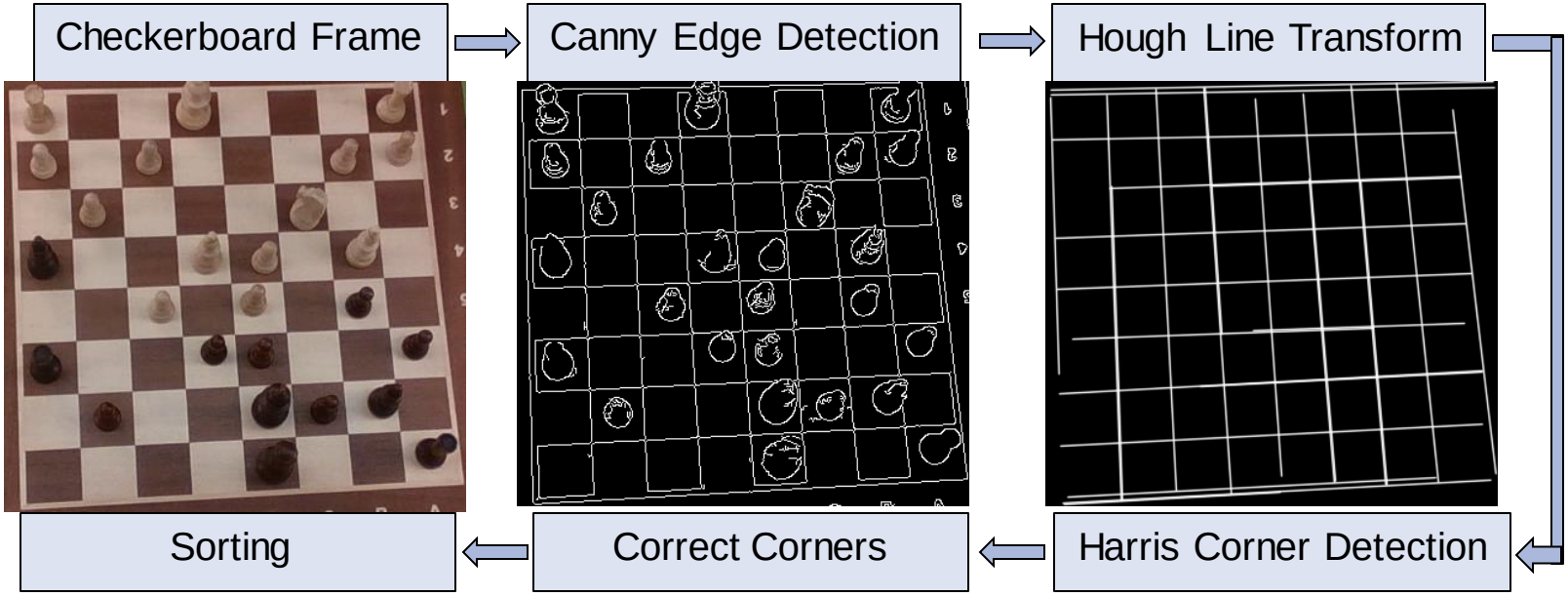
Hough Transform



Bunch of sinusoids intersecting at one point

Figure 2: Sinusoids on image domain. Adapted from https://shorturl.at/lBOX5

Checkerboard Corner Detection



Sorted corner coordinates

Harris Corner Detection: Basic Idea

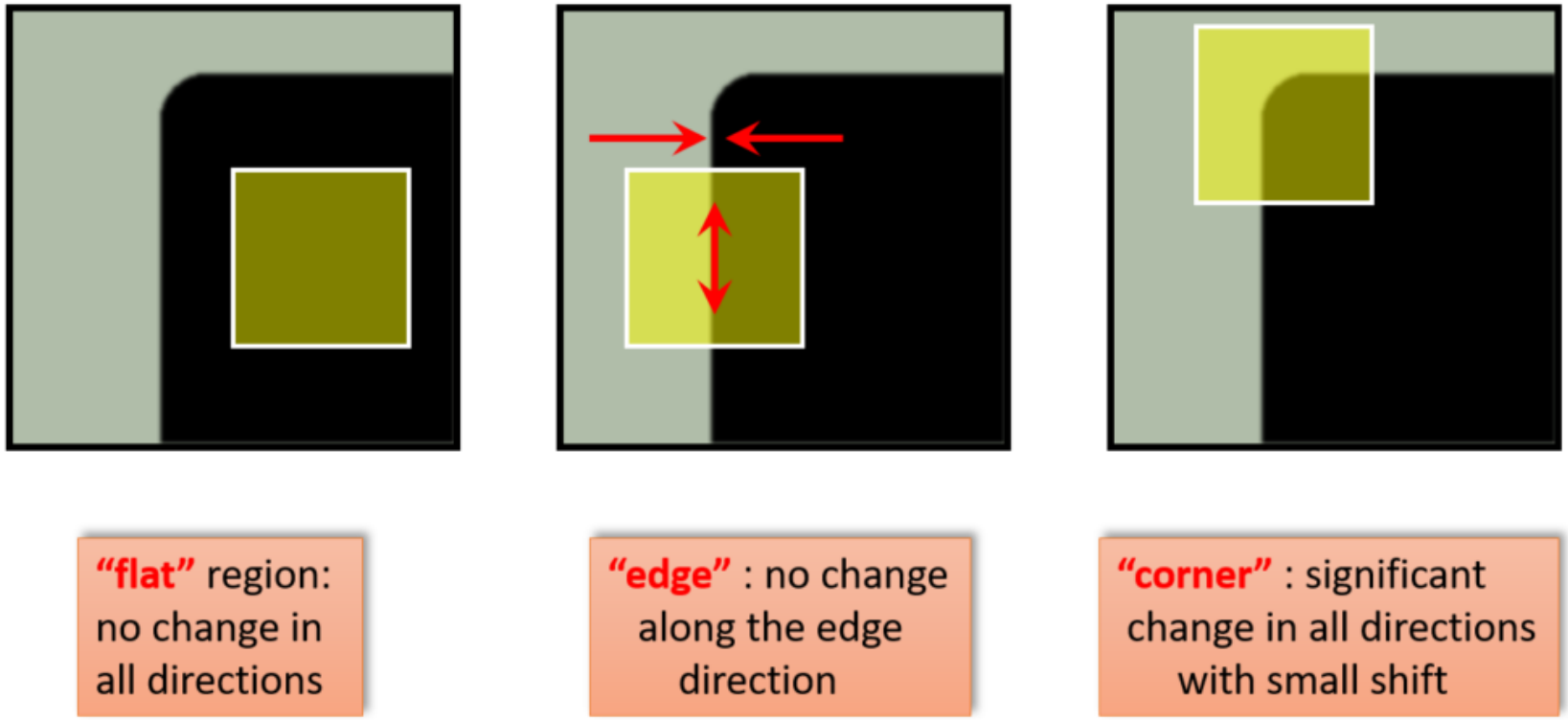
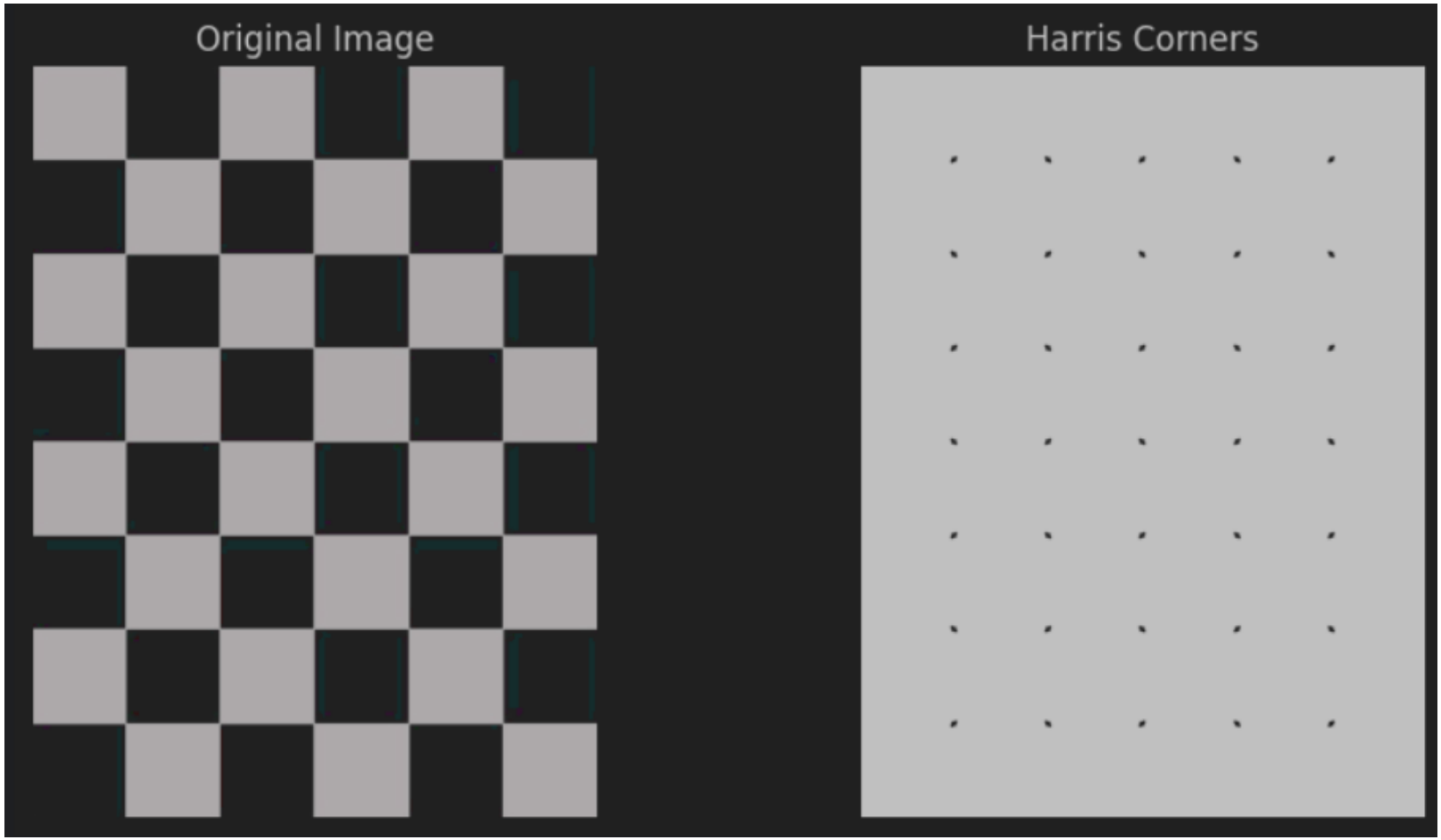
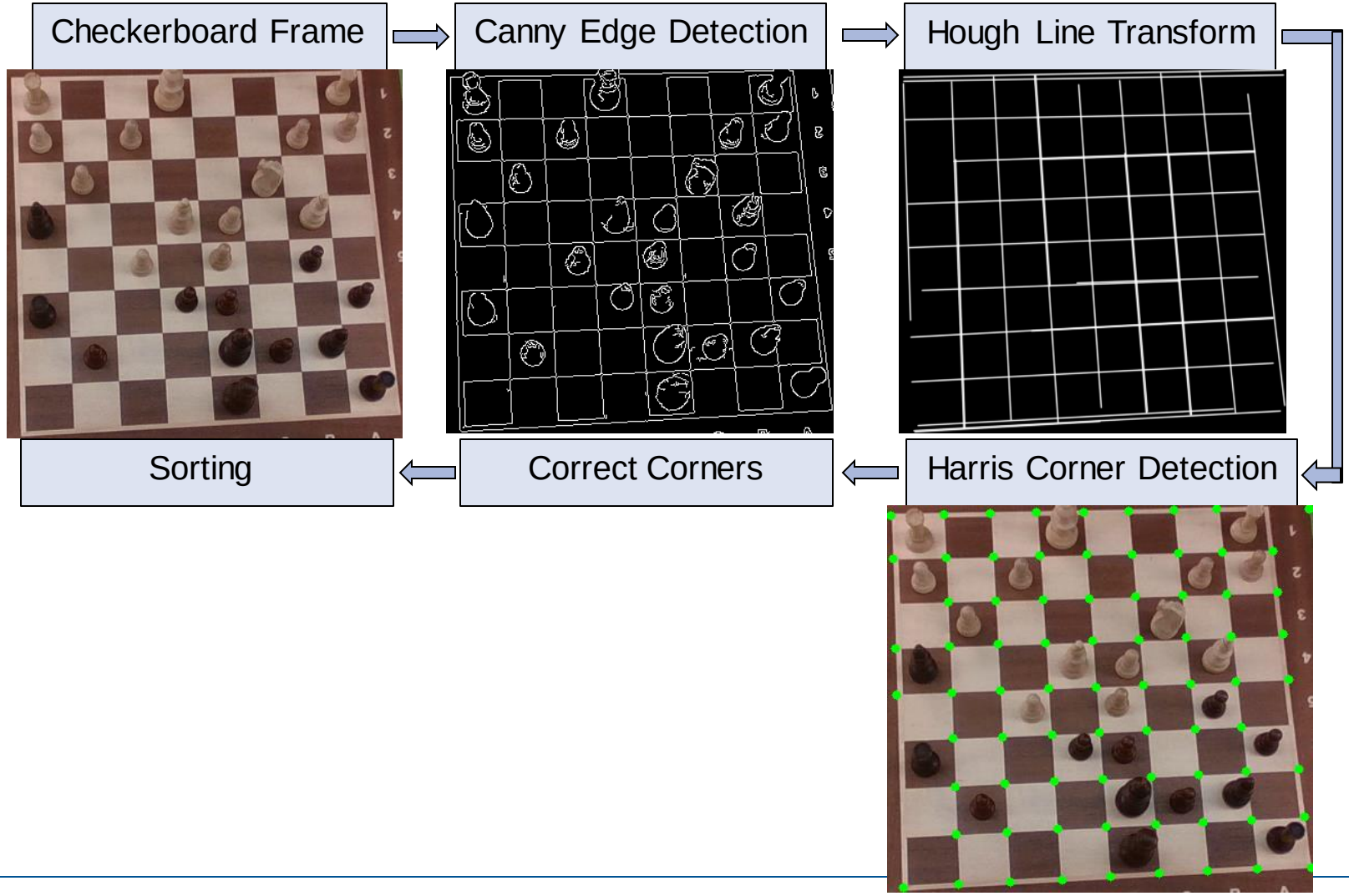


Figure: Harris Corner Detection. Adapted from https://shorturl.at/jtwQ8

Harris Corner Detection: Good for chess

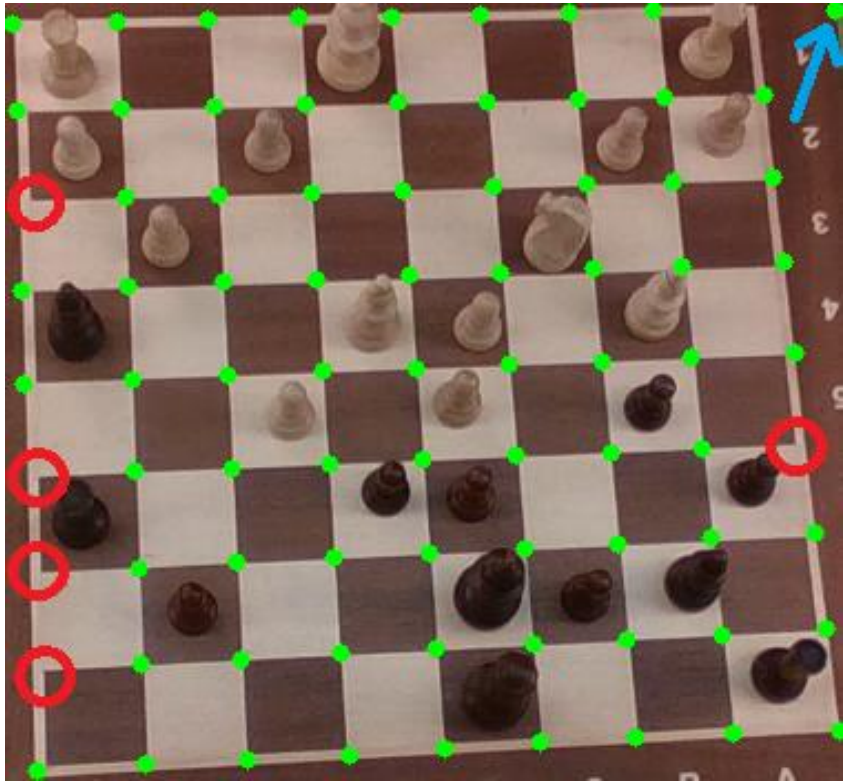


Checkerboard Corner Detection



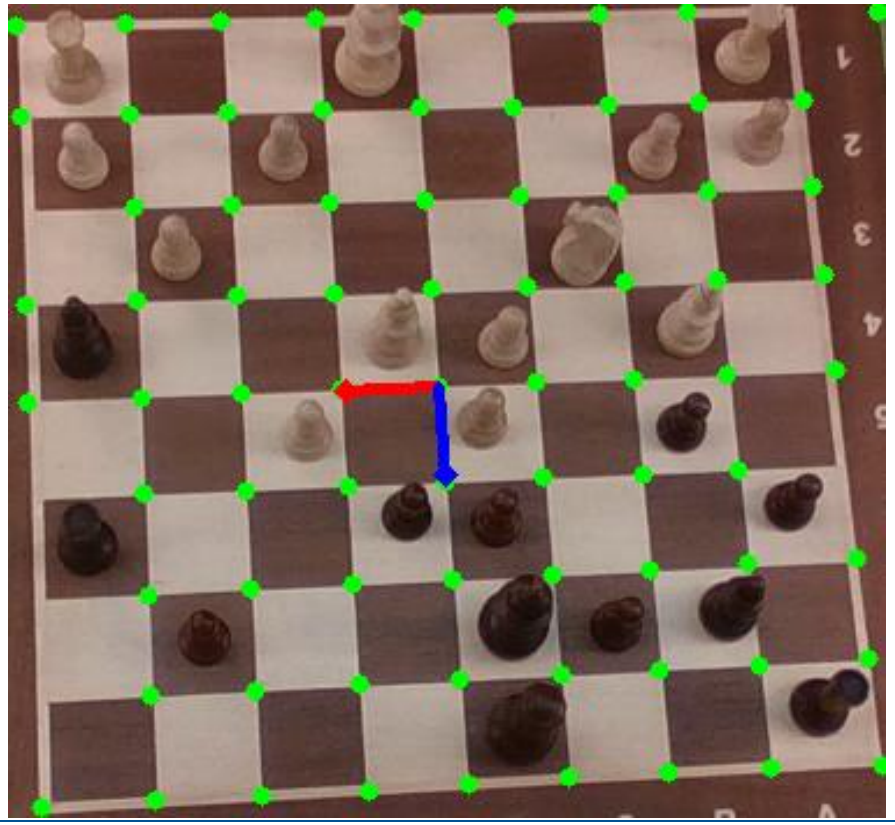
Checkerboard Corner Detection

- What if some corners are not detected? Or some corners are wrongly detected?



Checkerboard Corner Detection

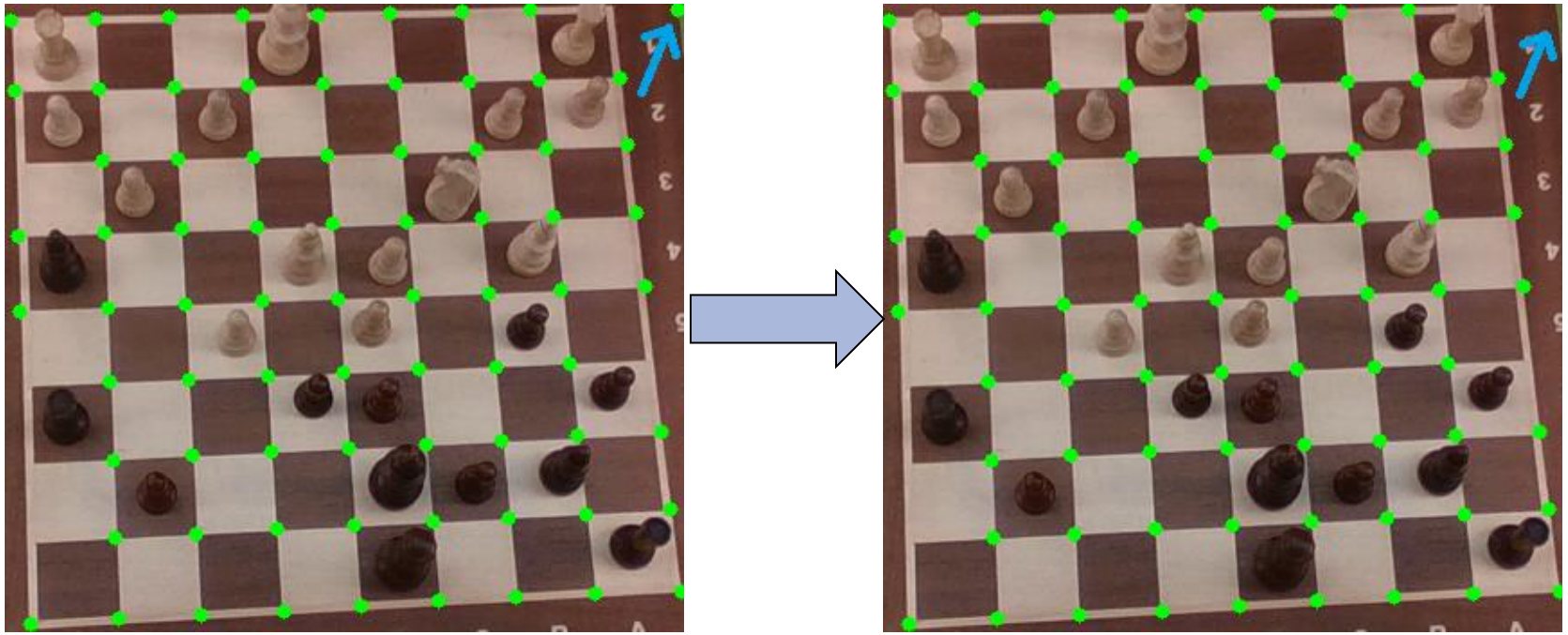
- Predict vertical and horizontal edge vectors.
- Median of existing vertical and horizontal vectors.



Checkerboard Corner Detection

For excess corners

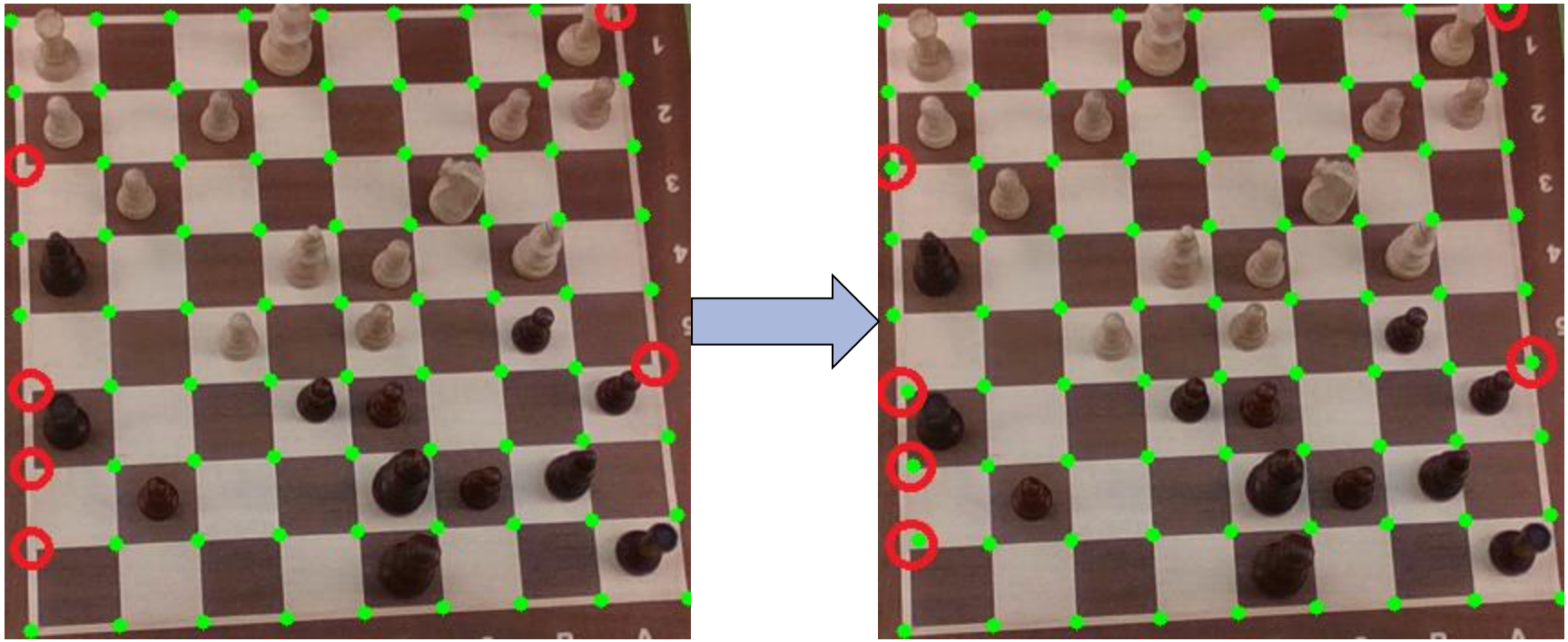
- Look at detected corner locations
- If detected corner does not match with any of predicted corners, delete the corner.



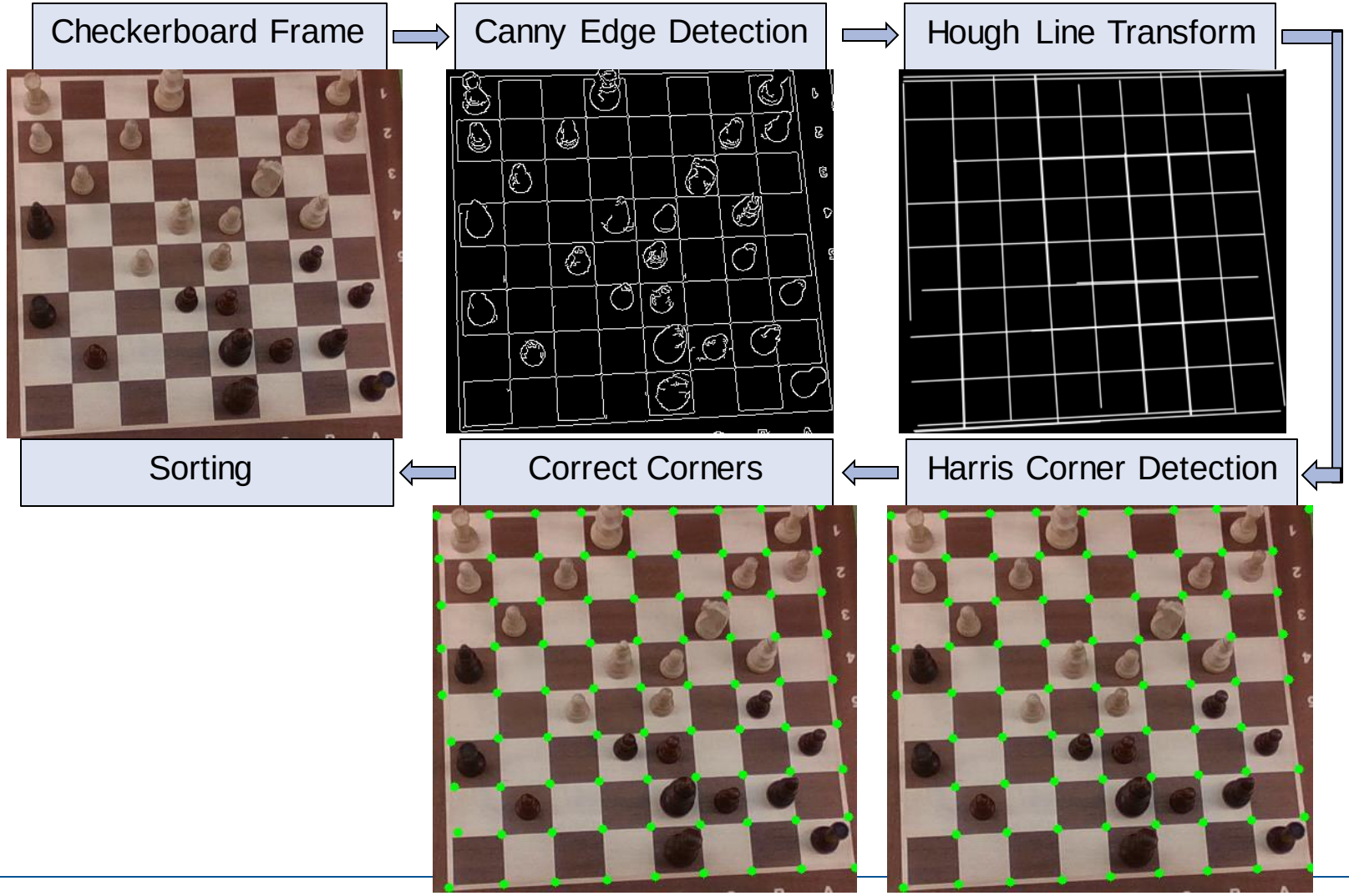
Checkerboard Corner Detection

For missing corners

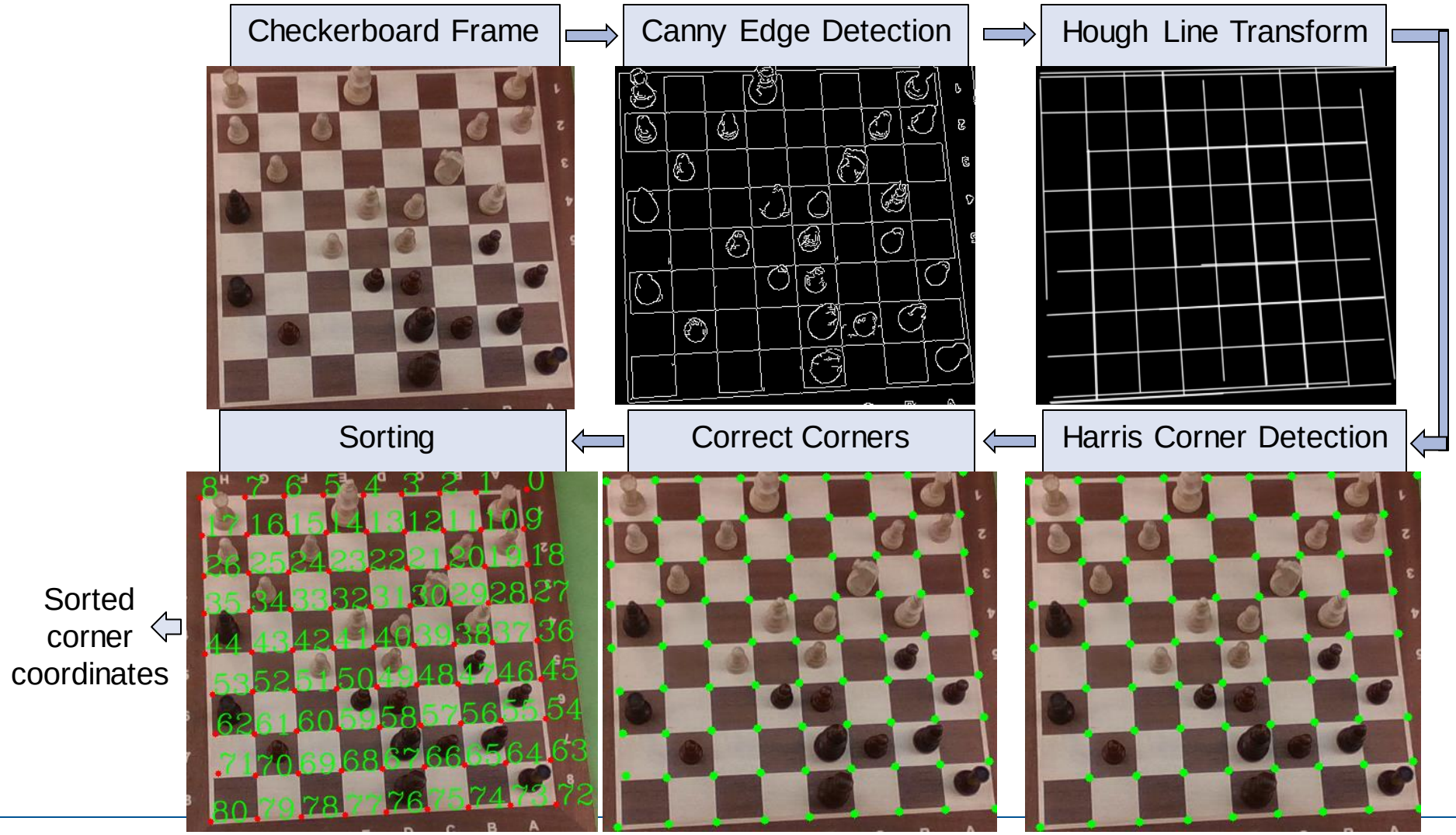
- Look at expected corner locations.
- If corner exists pass, if corner does not exist add a corner.



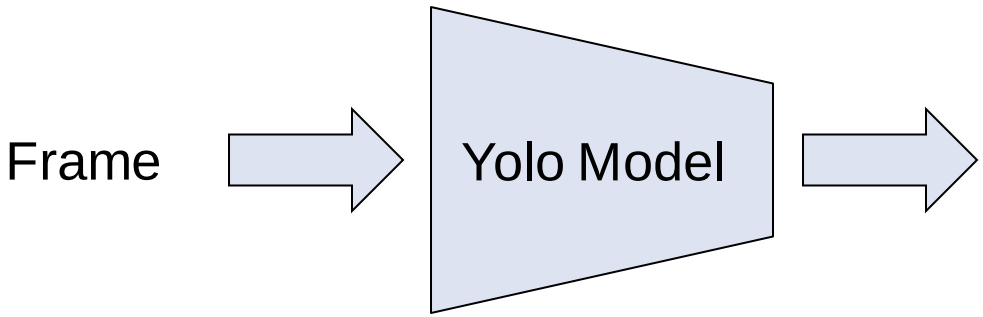
Checkerboard Corner Detection



Checkerboard Corner Detection



Piece Detection



- Piece classes
- Piece bounding boxes
- Piece probabilities

Hand Detection: Intersection Over Union

- Hand Flag = $\begin{cases} 1 & \text{if IoU} > \text{threshold} \\ 0 & \text{otherwise} \end{cases}$

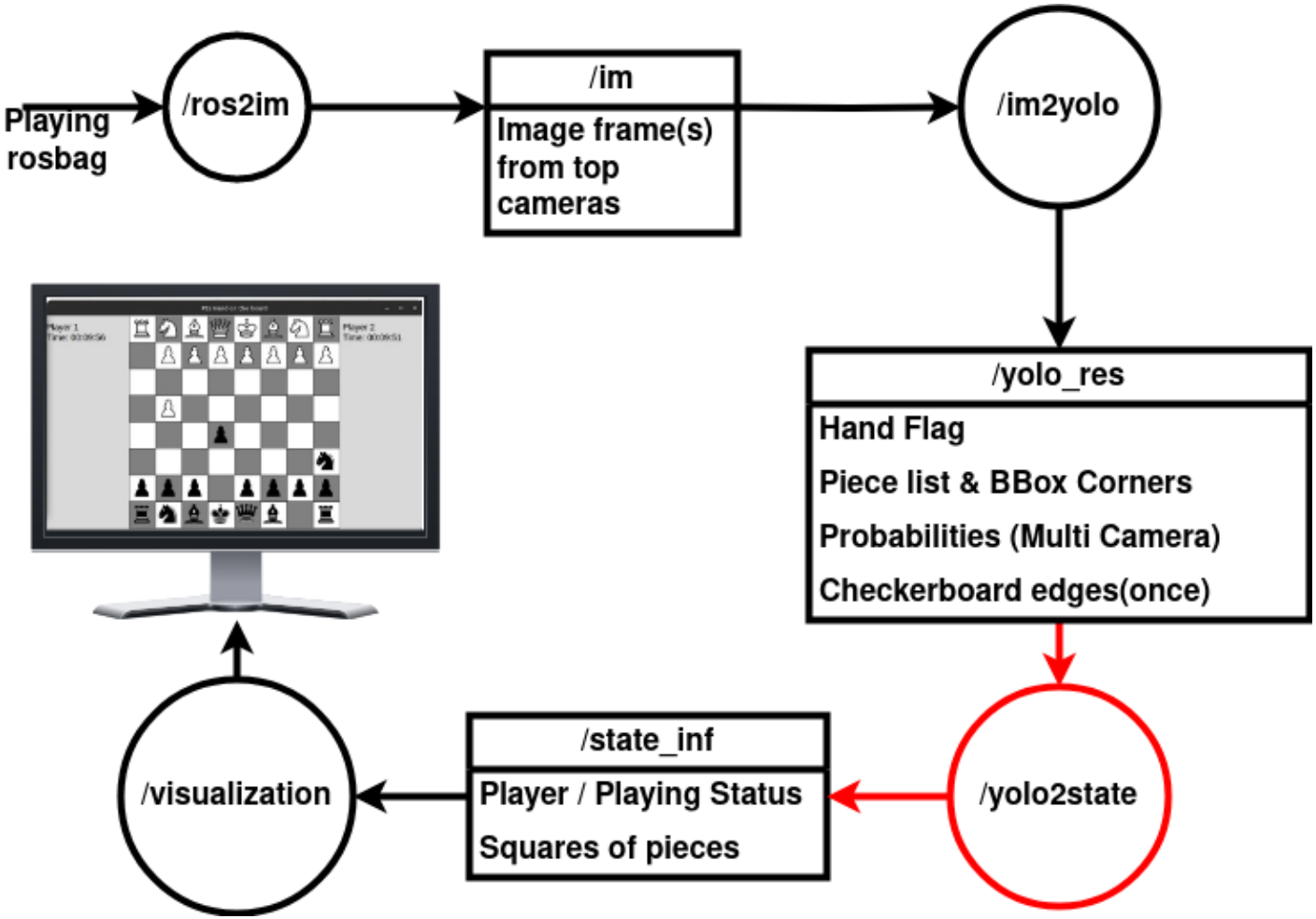
$$\text{IoU} = \frac{\text{Area of Overlap}}{\text{Area of Union}}$$

Intersection over Hand (IoH)

To understand what ratio of hand is over the checkerboard

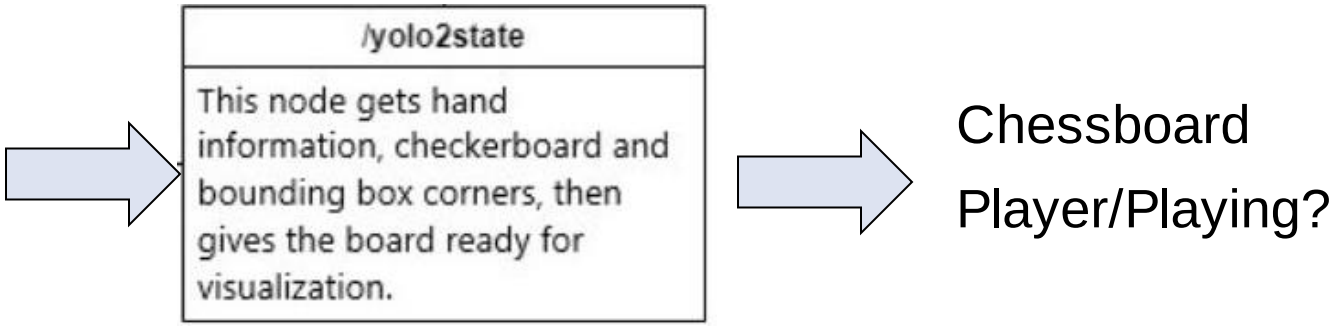
GAME STATE

Game State Management

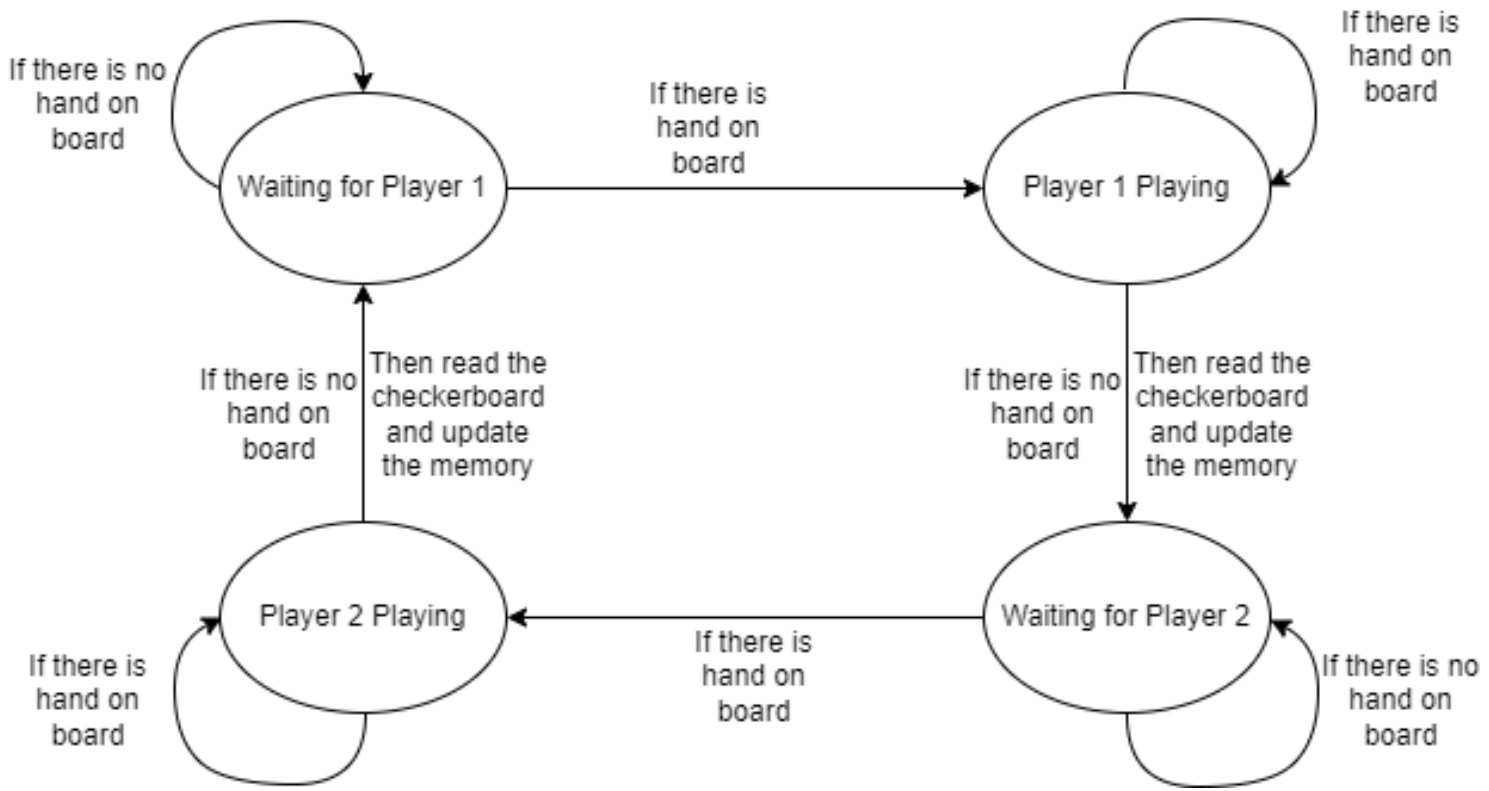


Game State Management

- Checkerboard coordinates
- Hand flag
- Piece classes
- Piece bounding boxes
- Piece probabilities



Game State Management – State Machine



Game State Management

- Read the board only at the end of a move!

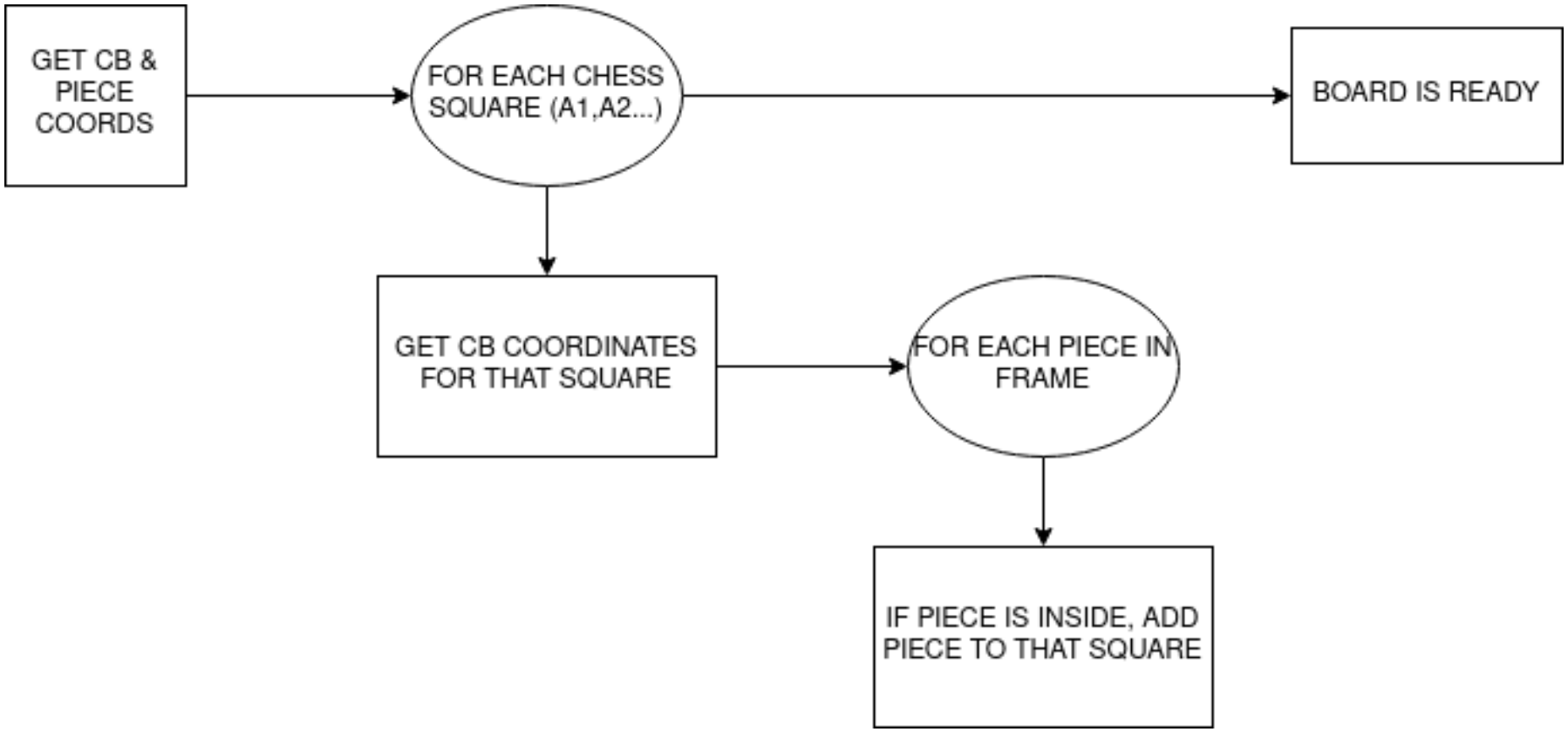
- The reasons for the decision:
  - Avoid piece detection errors due to blockage of hand
  - Fasten the code

Placing the Pieces into the Chessboard

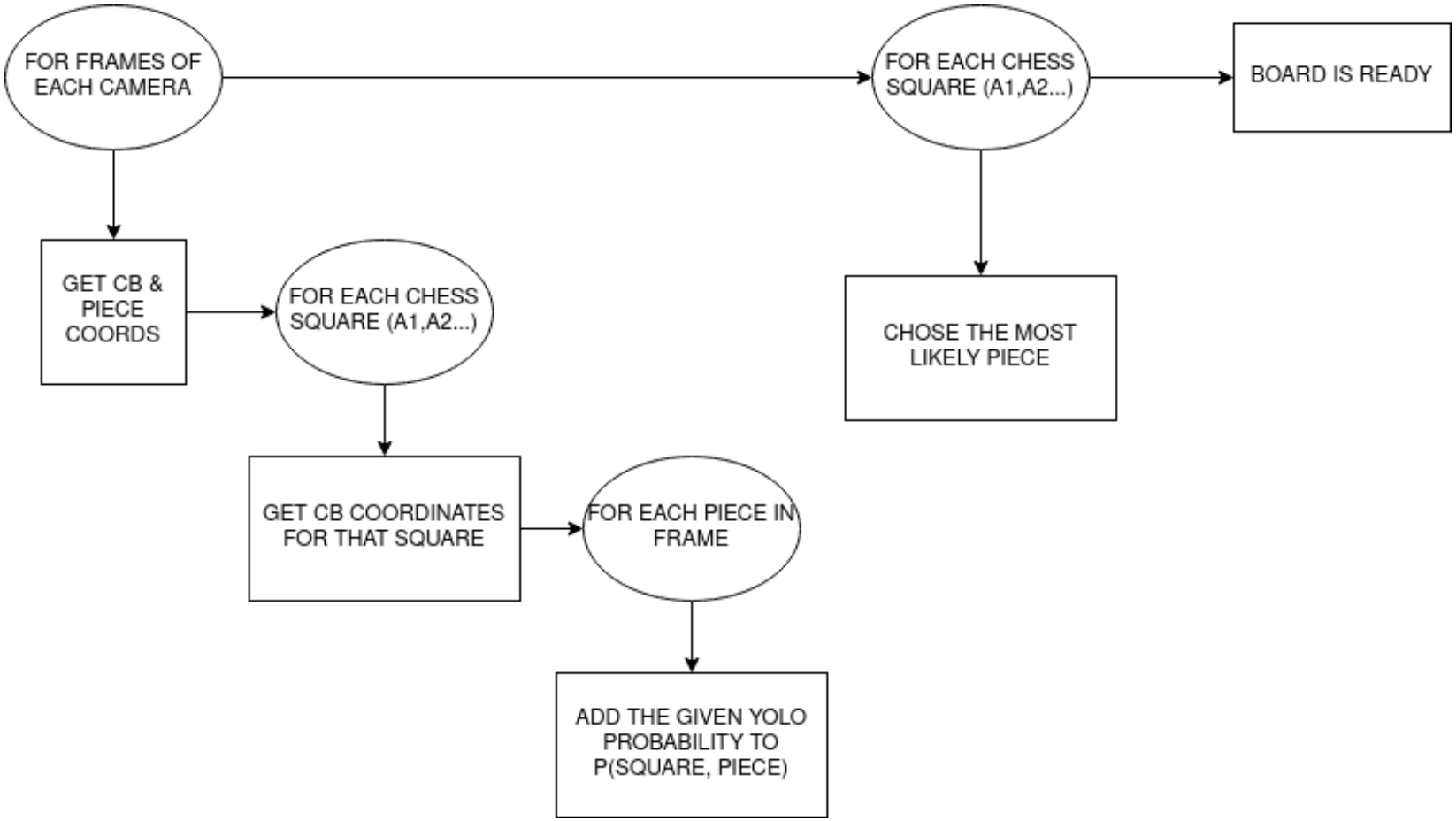
- We should evaluate the BBox and Checkerboard corners in order to determine the state of the board.

- The implementation differs for single and multicamera scenarios.

Chessboard Placement – Single Camera



Chessboard Placement – Multicamera

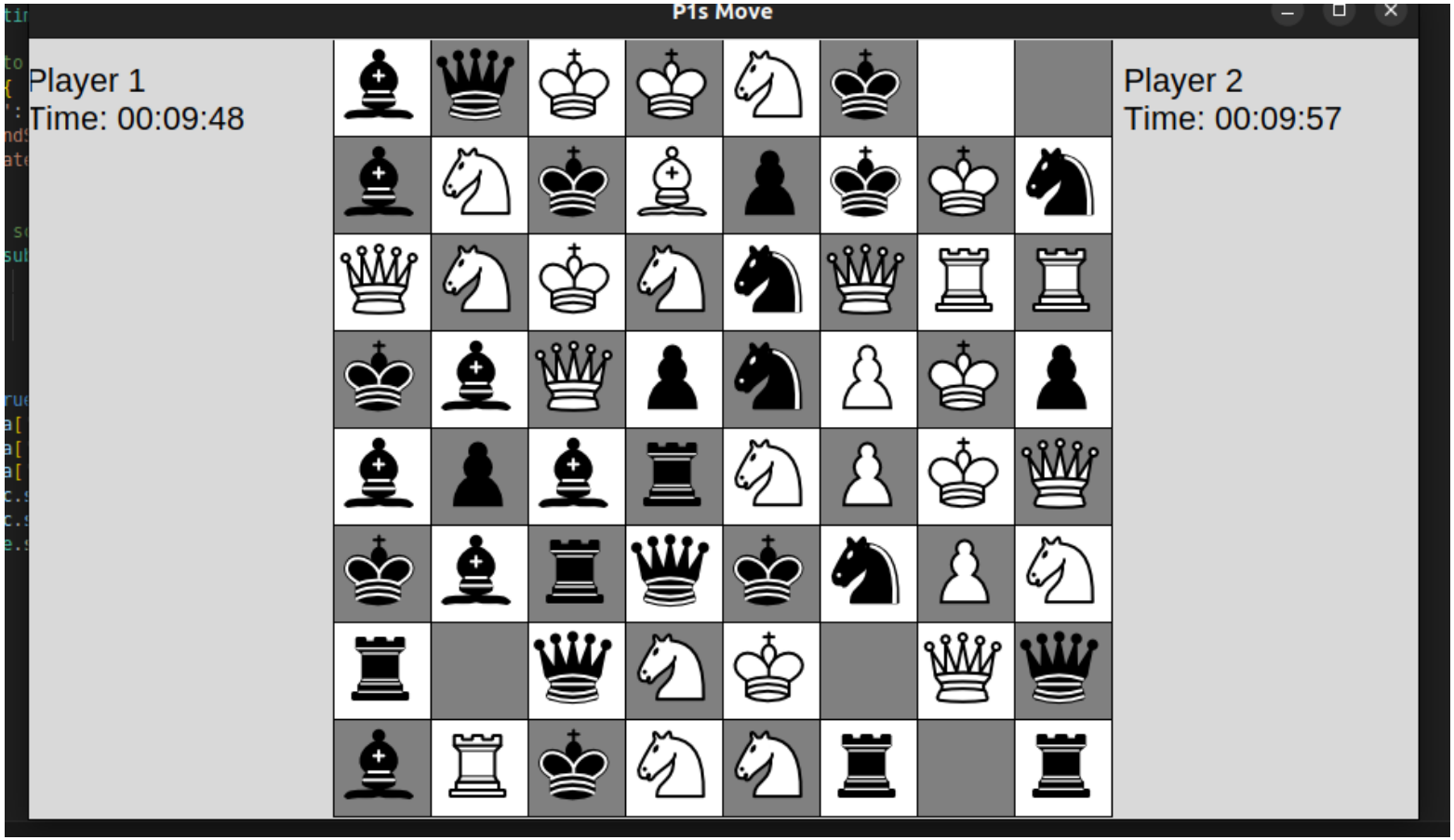


VISUALIZATION

Real-time visualization with Tkinter

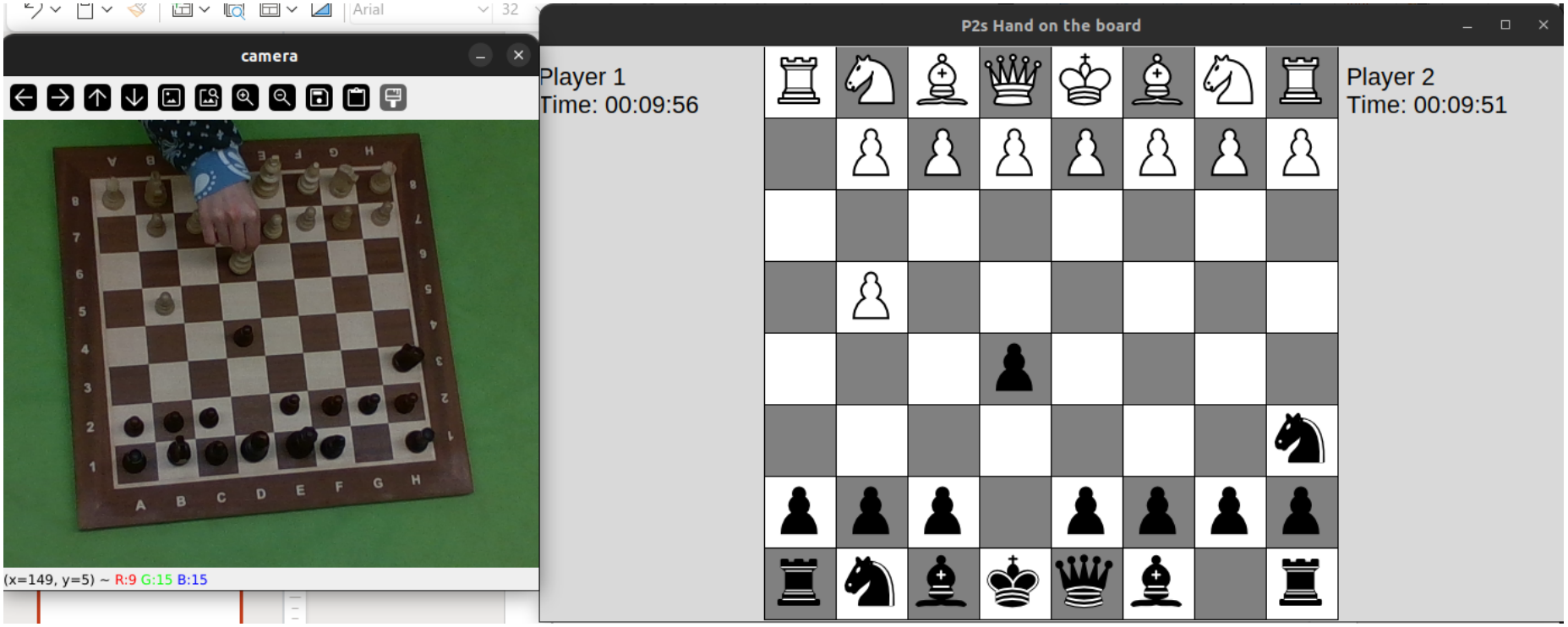
- Visualization is a subprocess.
- Send game state to subprocess through open pipe.
- Thread checks if data is on pipe and draws it.
- Run in infinite loop

Real-time tracking of remaining times.



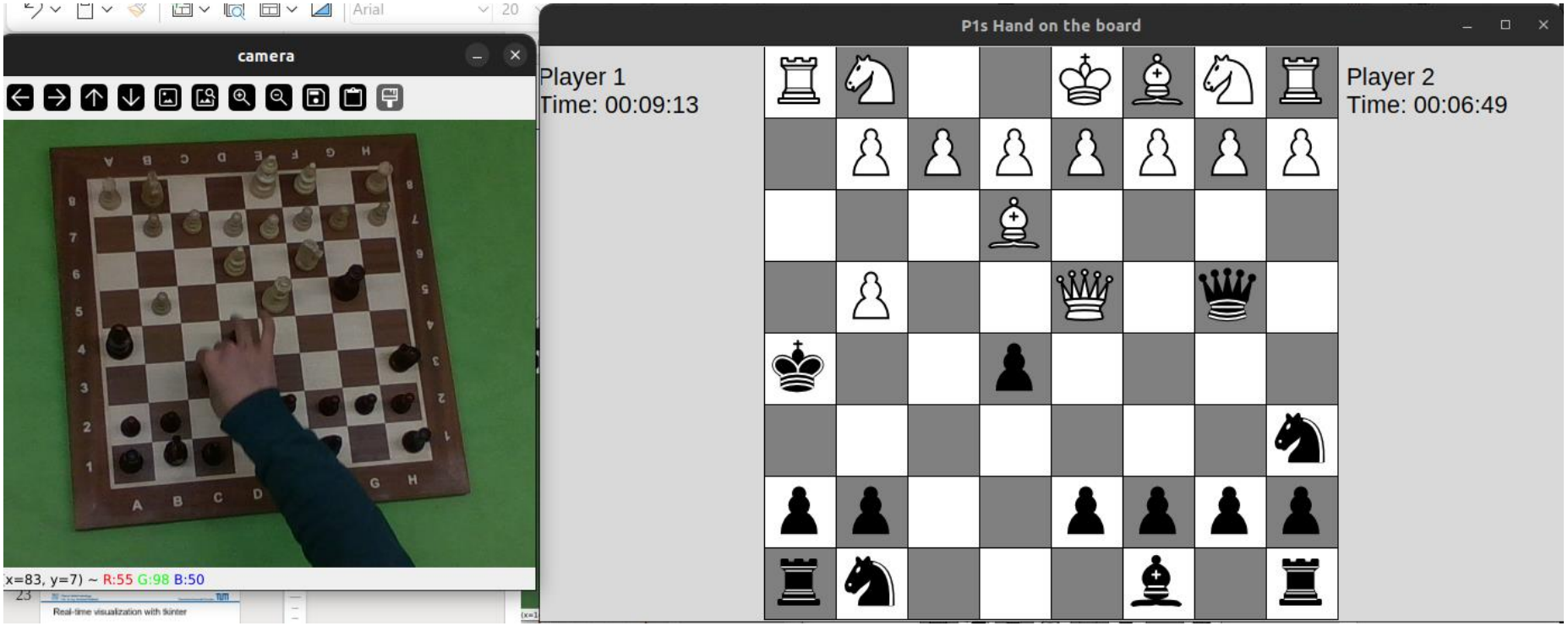
Hand State on Title

- 4 State: P2s Move, P1s Move, P1s Hand on the board, P2s Hand on the board
- Updated in the Tkinter window in real-time



Hand State on Title

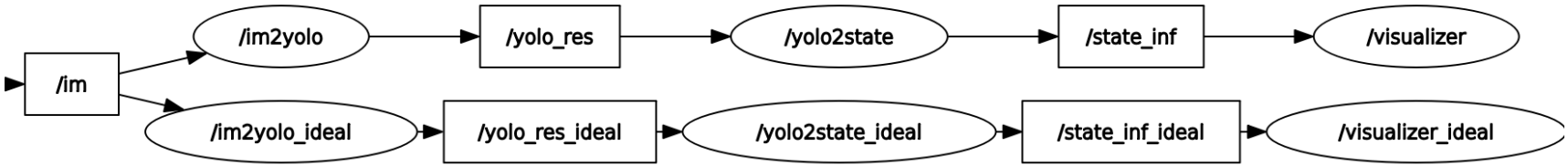
- 4 State: P2s Move, P1s Move, P1s Hand on the board, P2s Hand on the board
- Updated in the Tkinter window in real-time



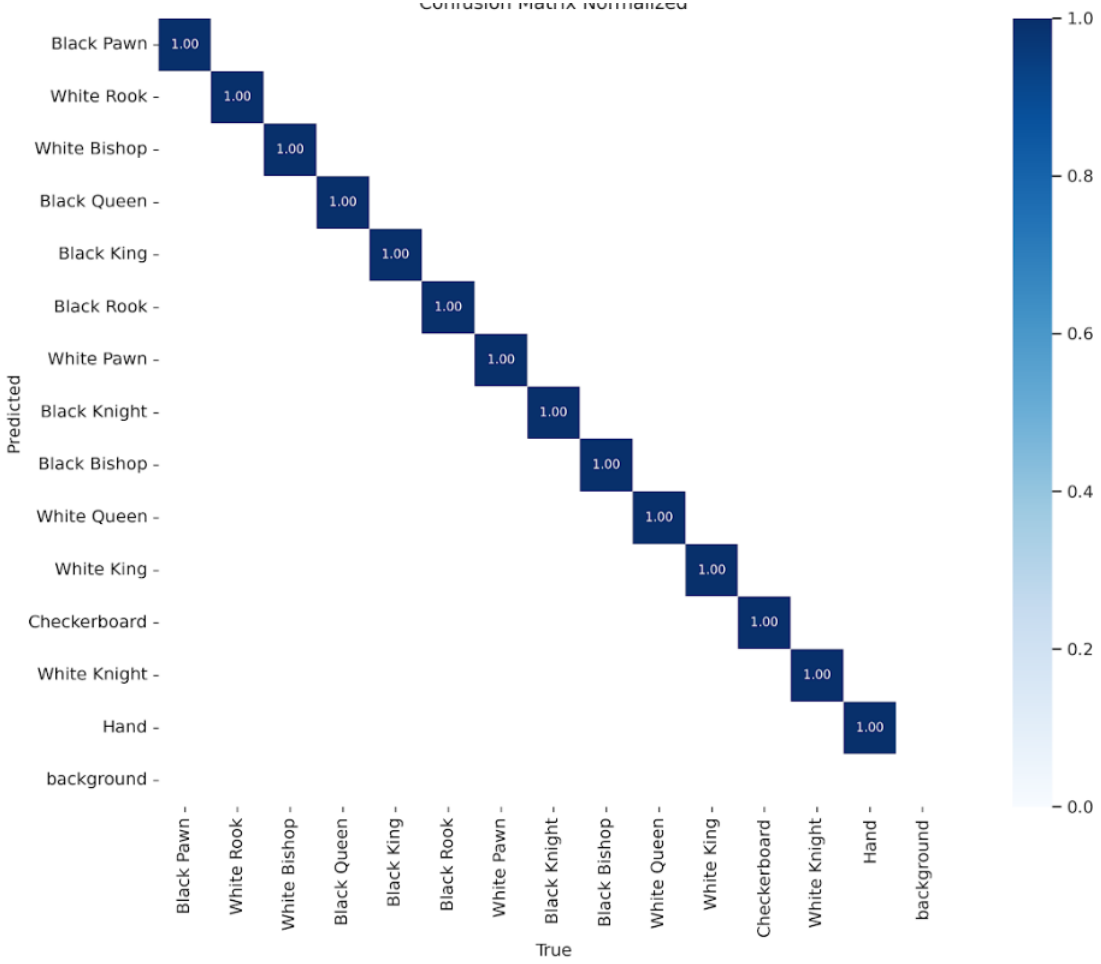
PERFORMANCE COMPARISIONS

Accuracy calculation on run-time

- Use a perfect YOLO model
  - Every piece is correctly classified
  - Now we can test accuracy by frame

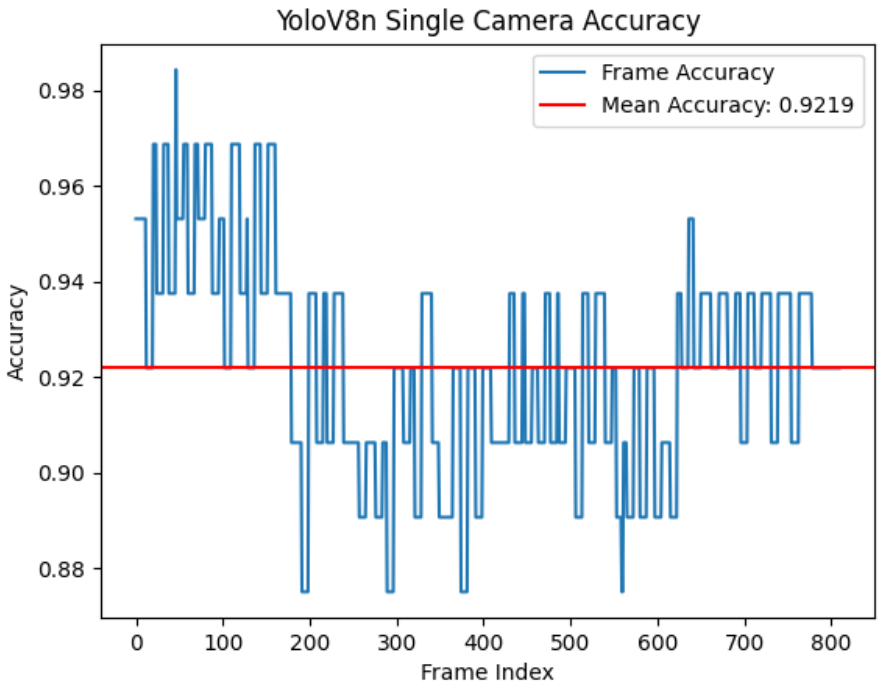


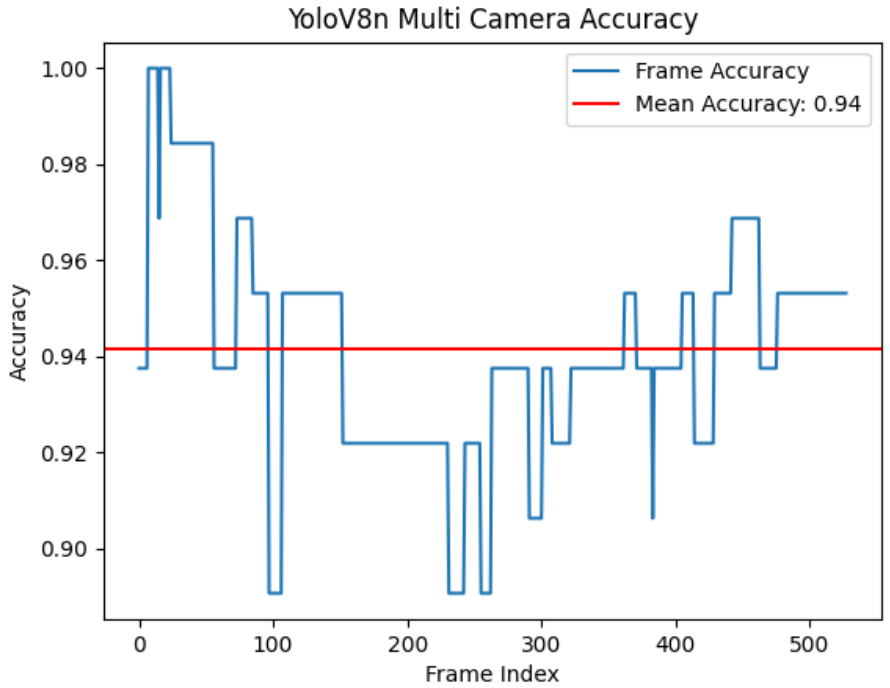
Confusion Matrix for overfitted model



Accuracy Results for YoloV8n (nano)

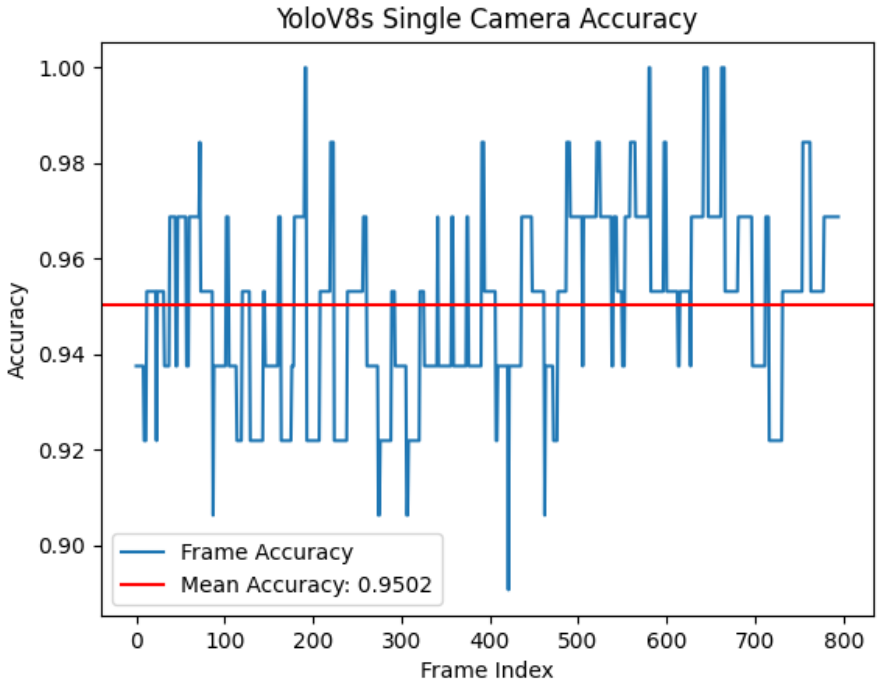
- From 0.92 to 0.94 with the cost of delay
- Mean performance is 0.9219

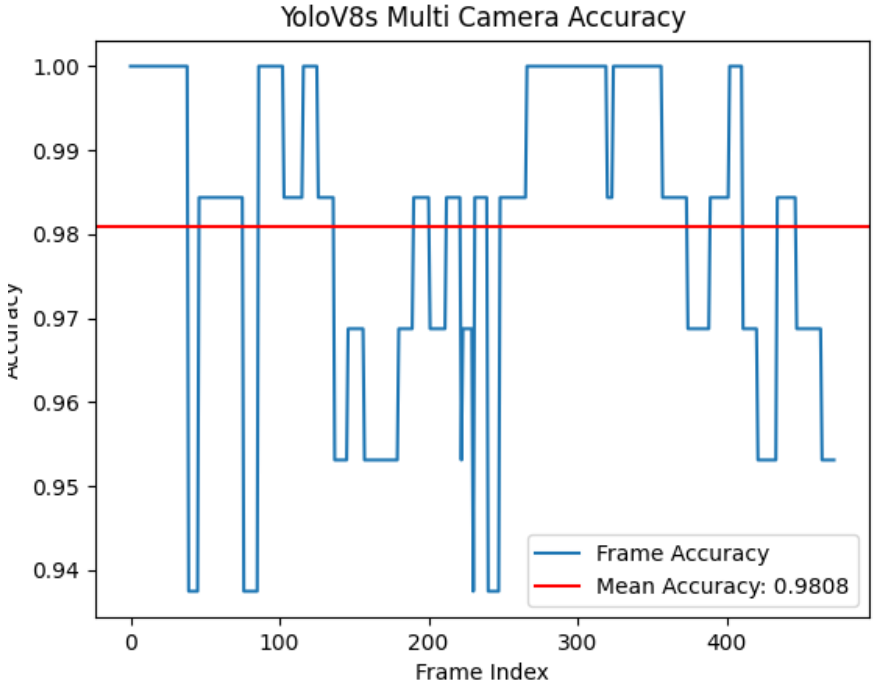




Accuracy Results for YoloV8s (small)

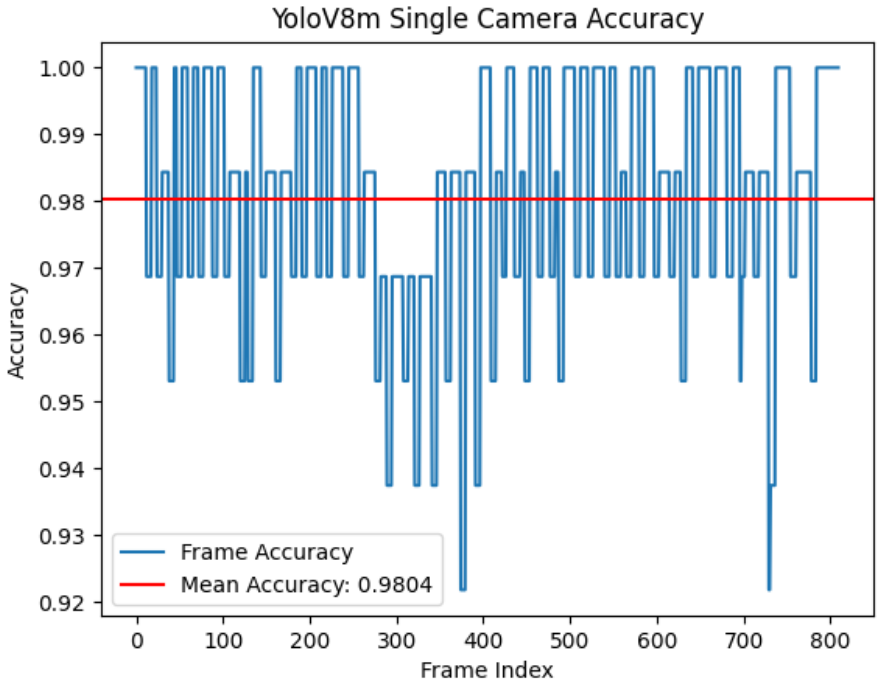
- Single camera accuracy moved to 0.95 from 0.92 (YoloV8n)
- Multi camera accuracy moved to 0.98 from 0.94 (YoloV8n)

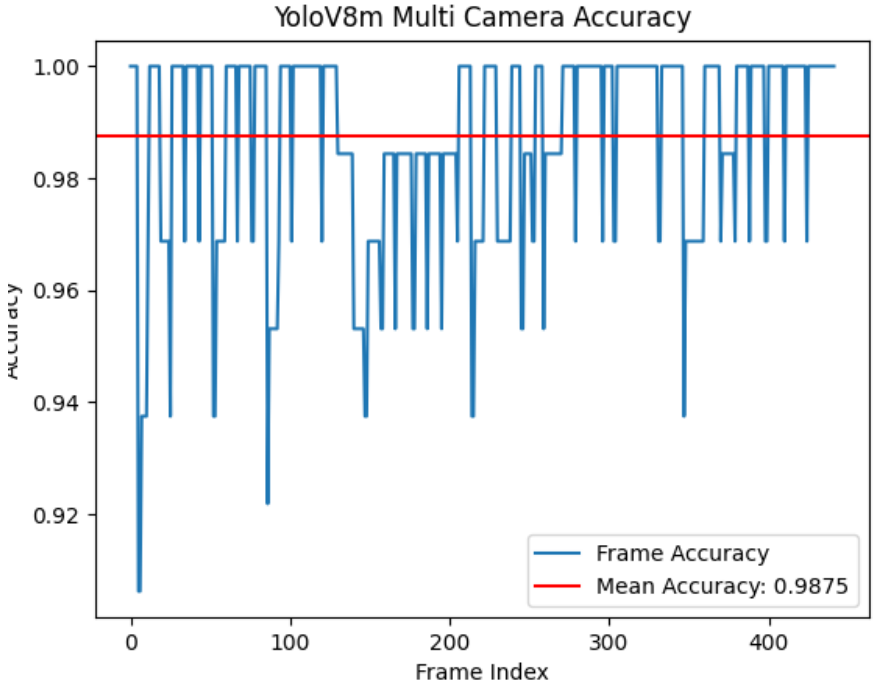




Accuracy Results for YoloV8m (medium)

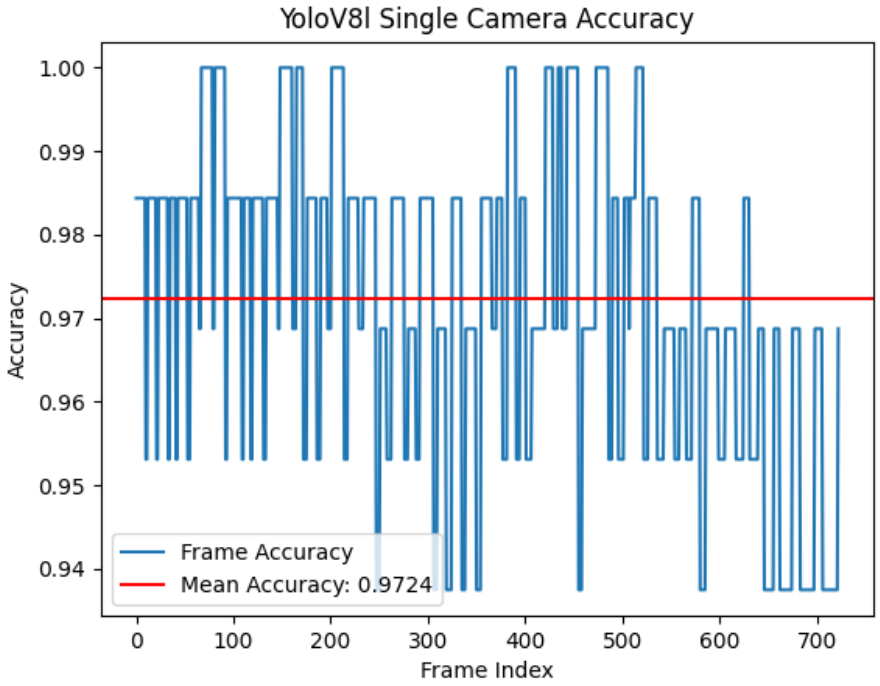
- Single camera accuracy moved to 0.98 from 0.95 (YoloV8s)
- Multi camera accuracy moved to 0.9875 from 0.9808 (YoloV8s)

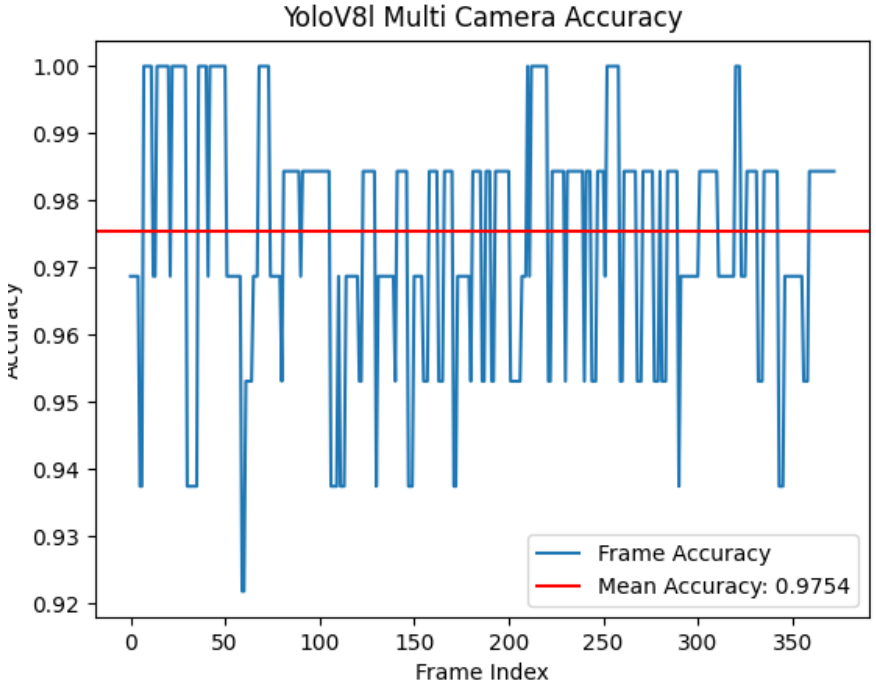




Accuracy Results for YoloV8l (large)

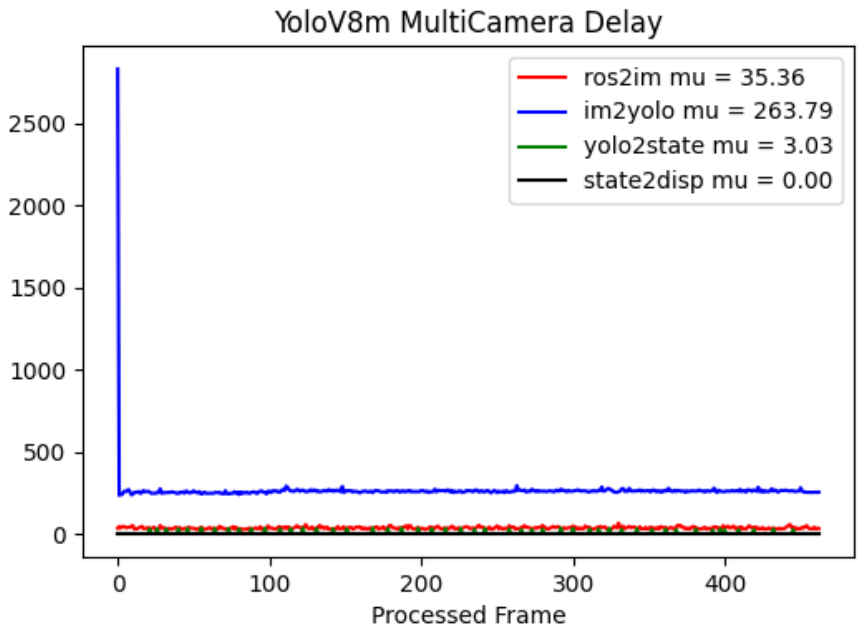
- Single camera accuracy moved to 0.97 from 0.98 (YoloV8m)
- Multi camera accuracy moved to 0.9754 rom 0.9875 (YoloV8m)
- Large model performs worse since we don't have enough data

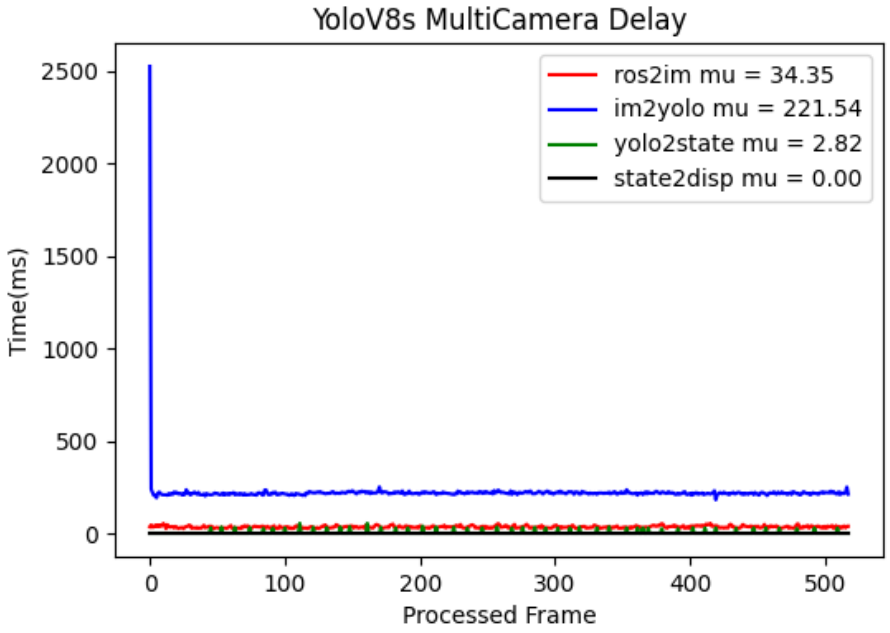




Multi-view Camera: does it pay off?

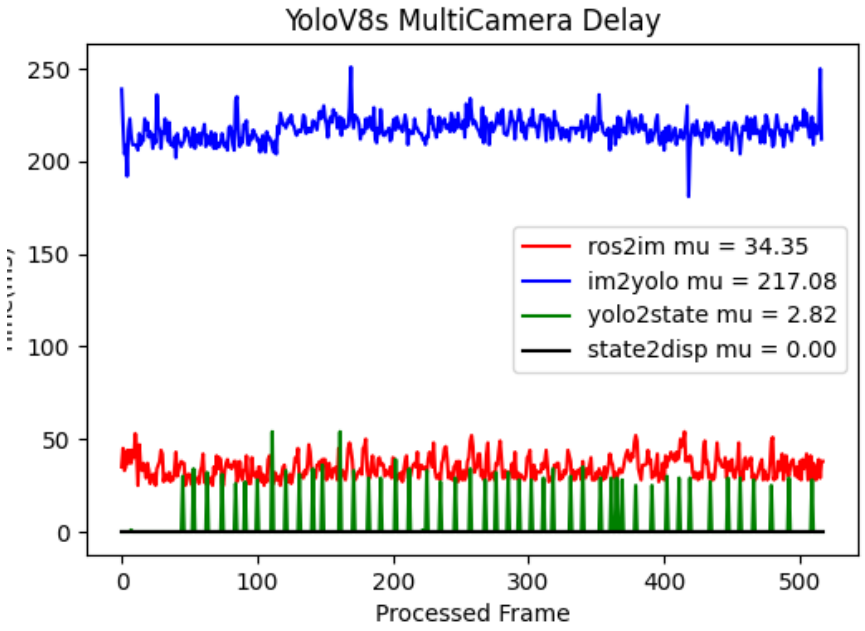
- It brings a marginal improvement over accuracy.
- However, we need to detect corners at the beginning two times
- Also with initialization, first delay is higher.

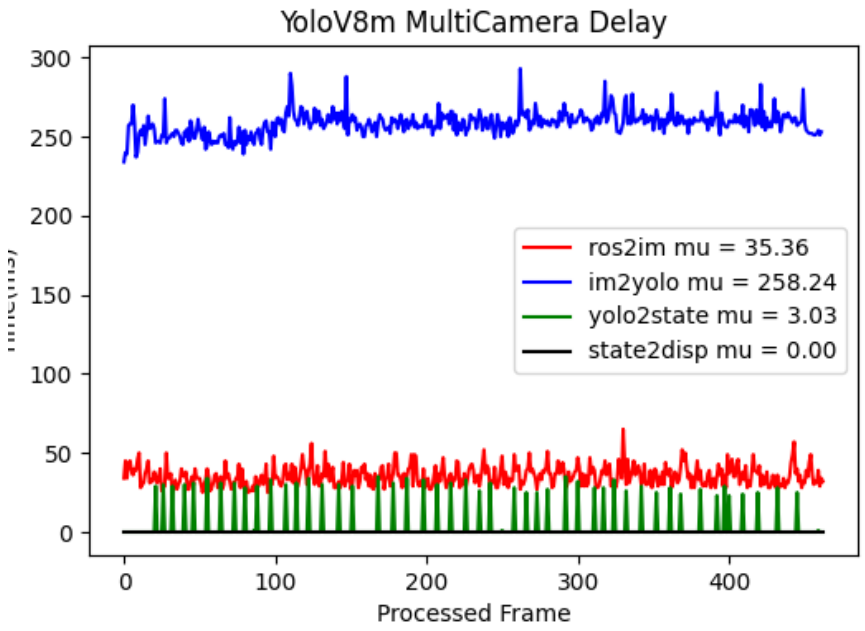




Multi-view Camera: stable after first frame

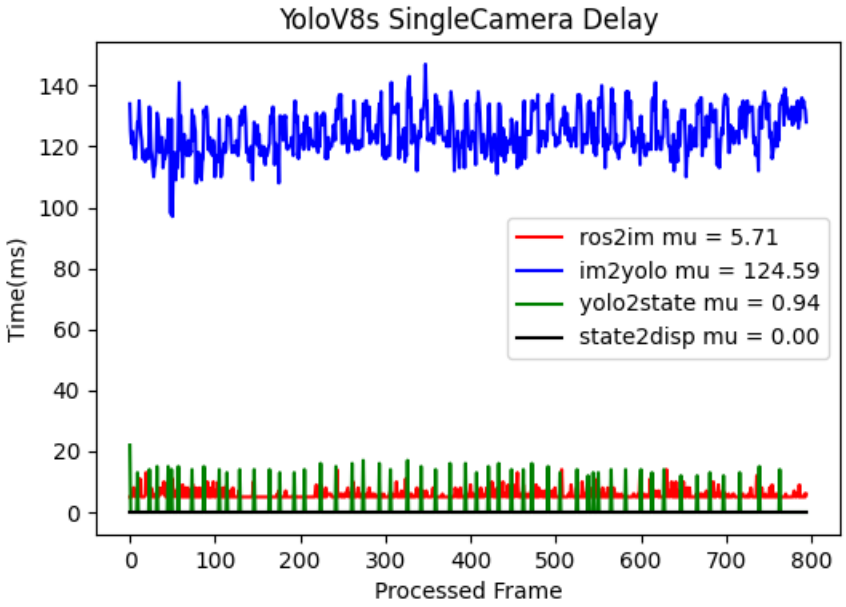
- After initialization, it processes the frames in comparable times
- Since medium model has more parameter, it takes more time to compute prediction than that of the small model

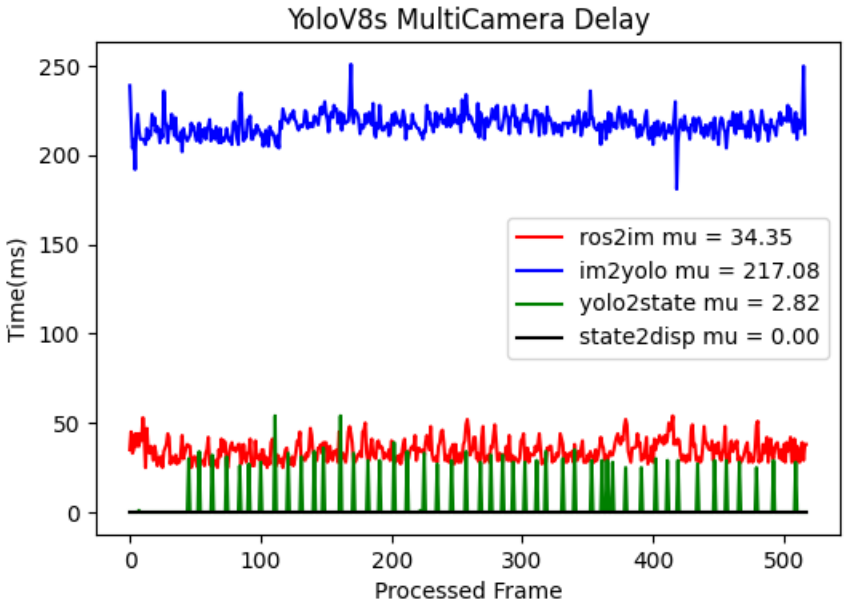




Delay: Multi vs. Single-Camera

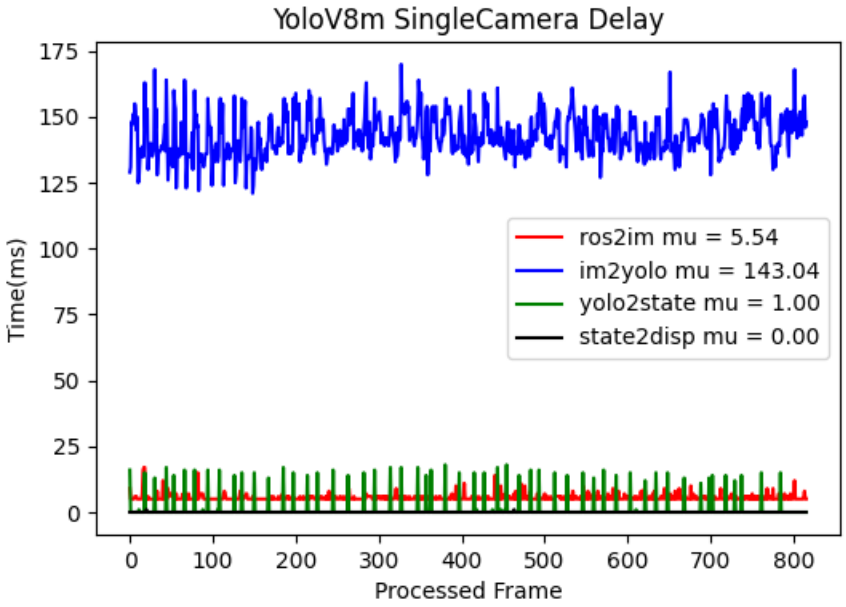
- Second camera is sequentially processed with the first one
- That causes the delay to almost double for every node
- Timings are given for small model

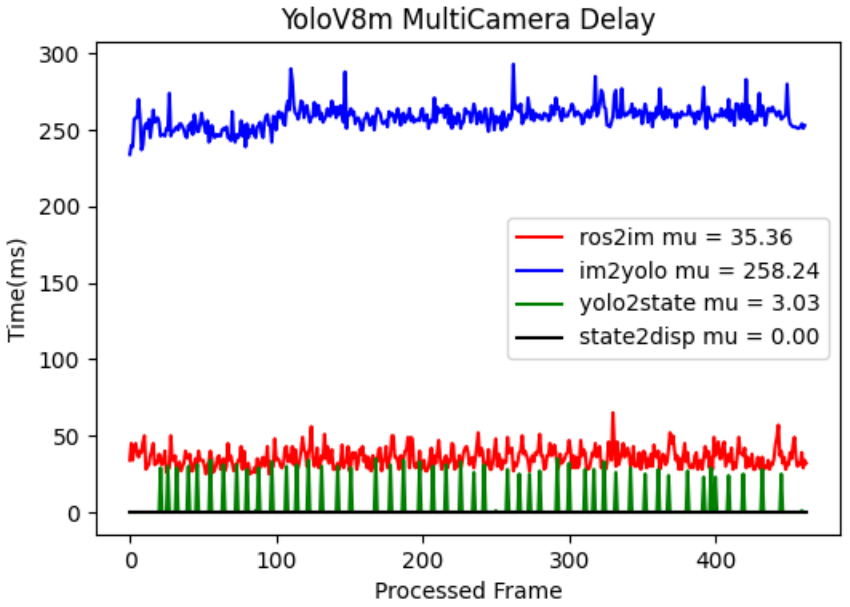




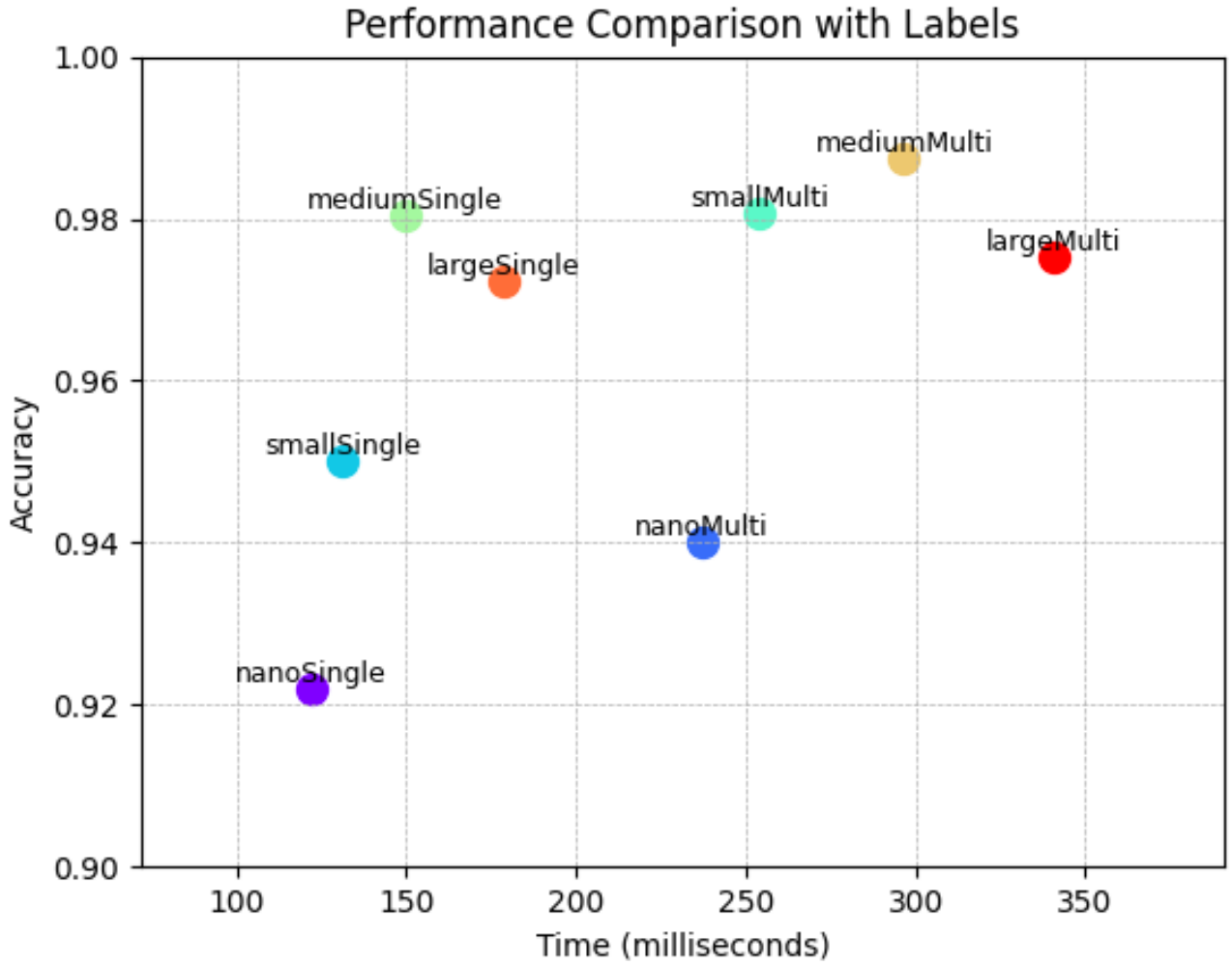
Delay: Multi vs. Single-Camera

- Second camera is sequentially processed with the first one
- That causes the delay to almost double for every node
- Timings are given for medium model (increased w.r.t. to small)

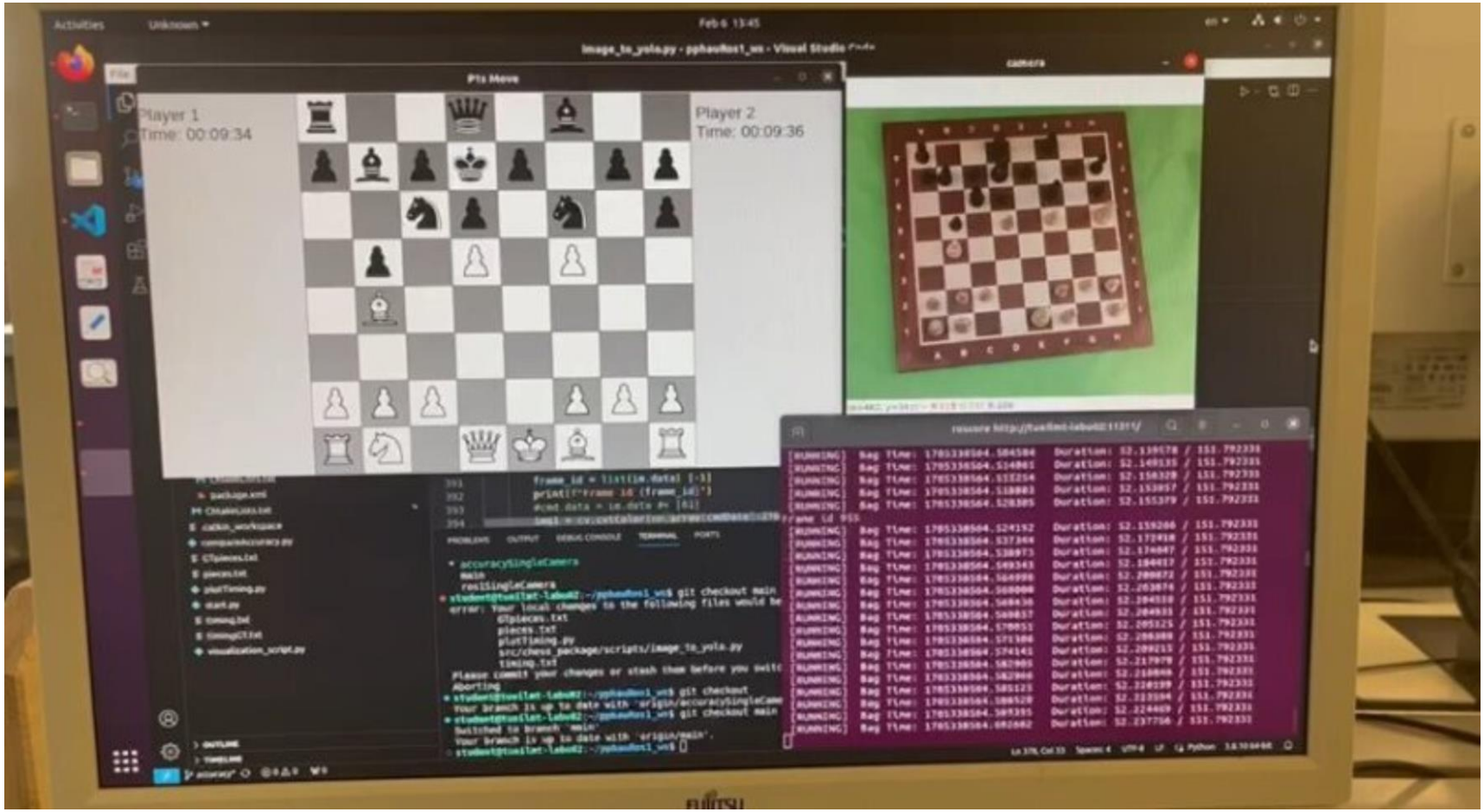




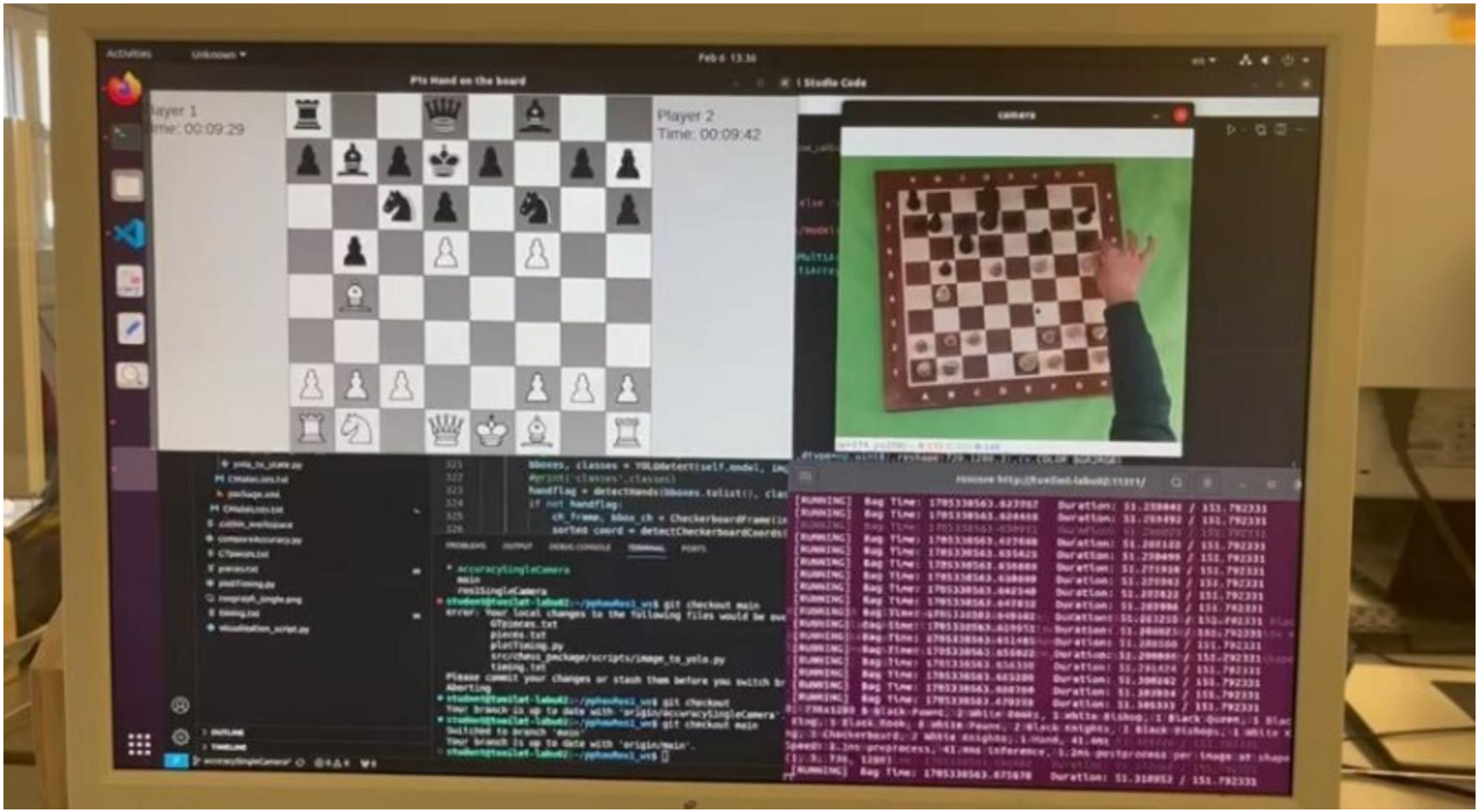
The trade-off of speed and accuracy



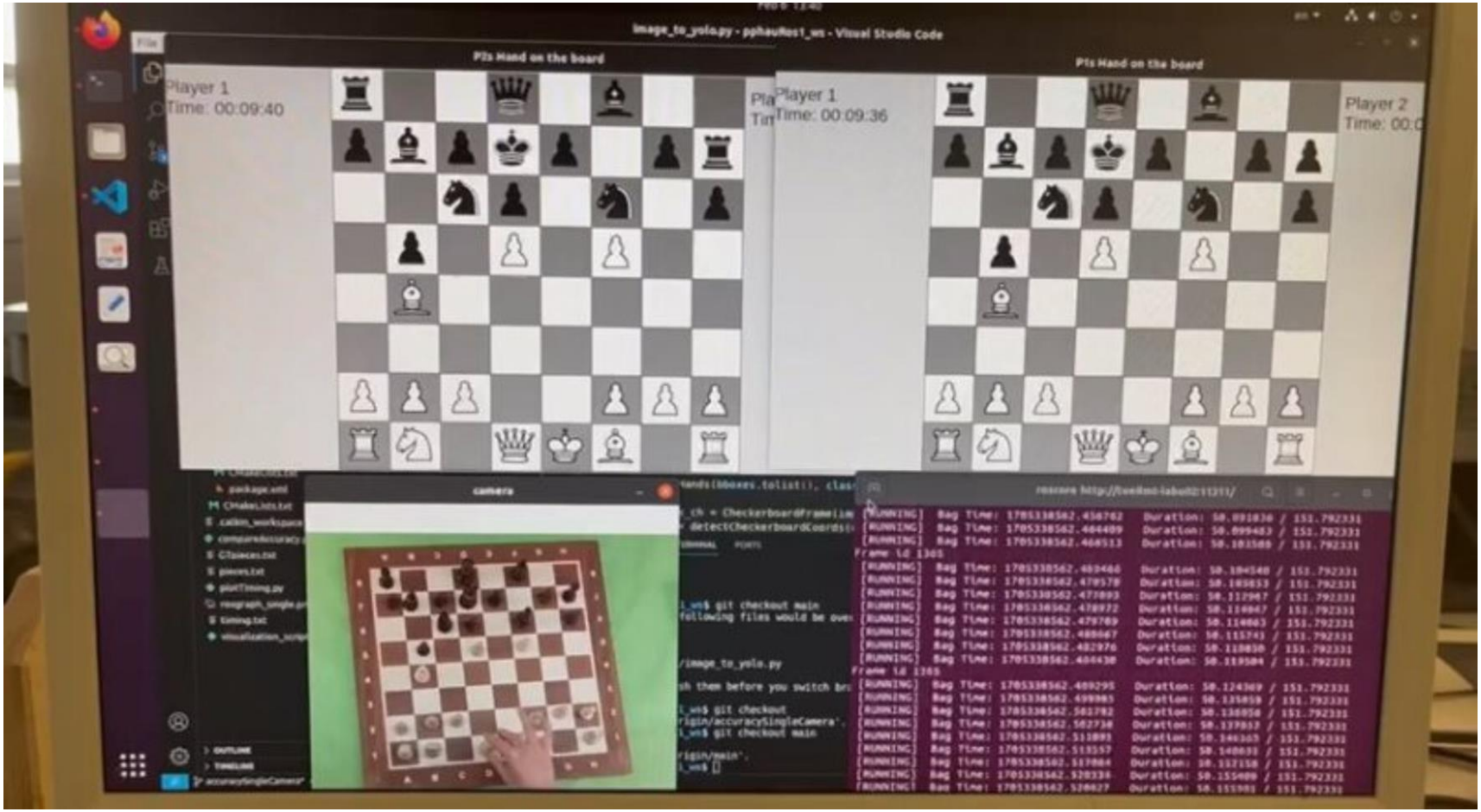
DEMO: YoloV8m, MultiCamera



DEMO: YoloV8m, SingleCamera

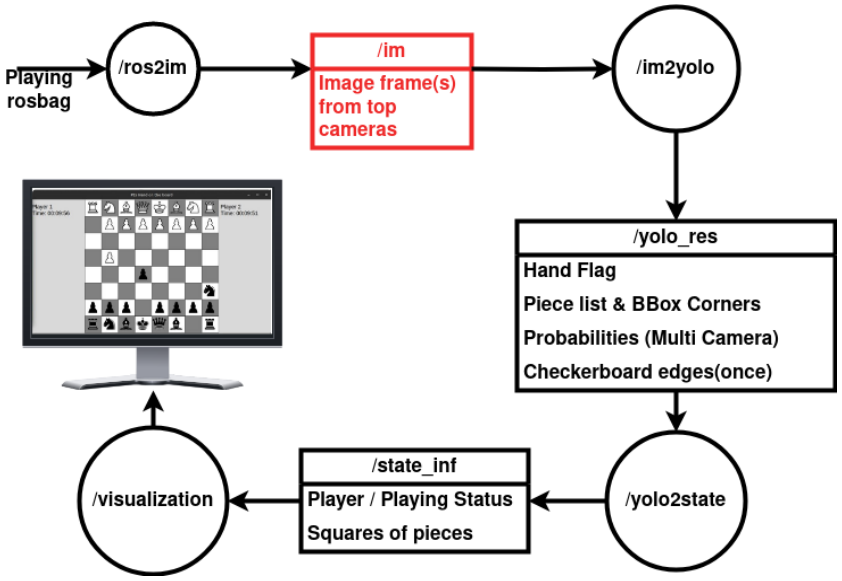


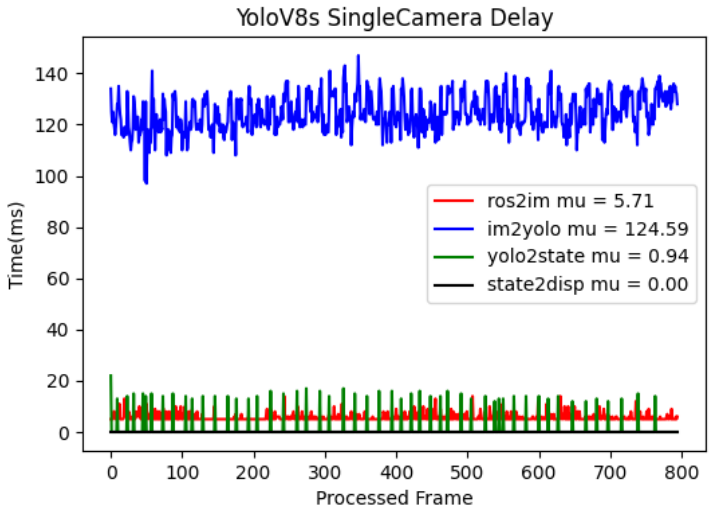
DEMO: Single Camera with Ground Truth



FUTURE WORK

- One node creates too much delay comparatively
- Buffer of /im topic can become unstabilized
  - Suggestion: Re-arrange the node-topic pipeline for app. eq. Delays
  - Challenge: Each code snippet should be taken into account





QUESTIONS