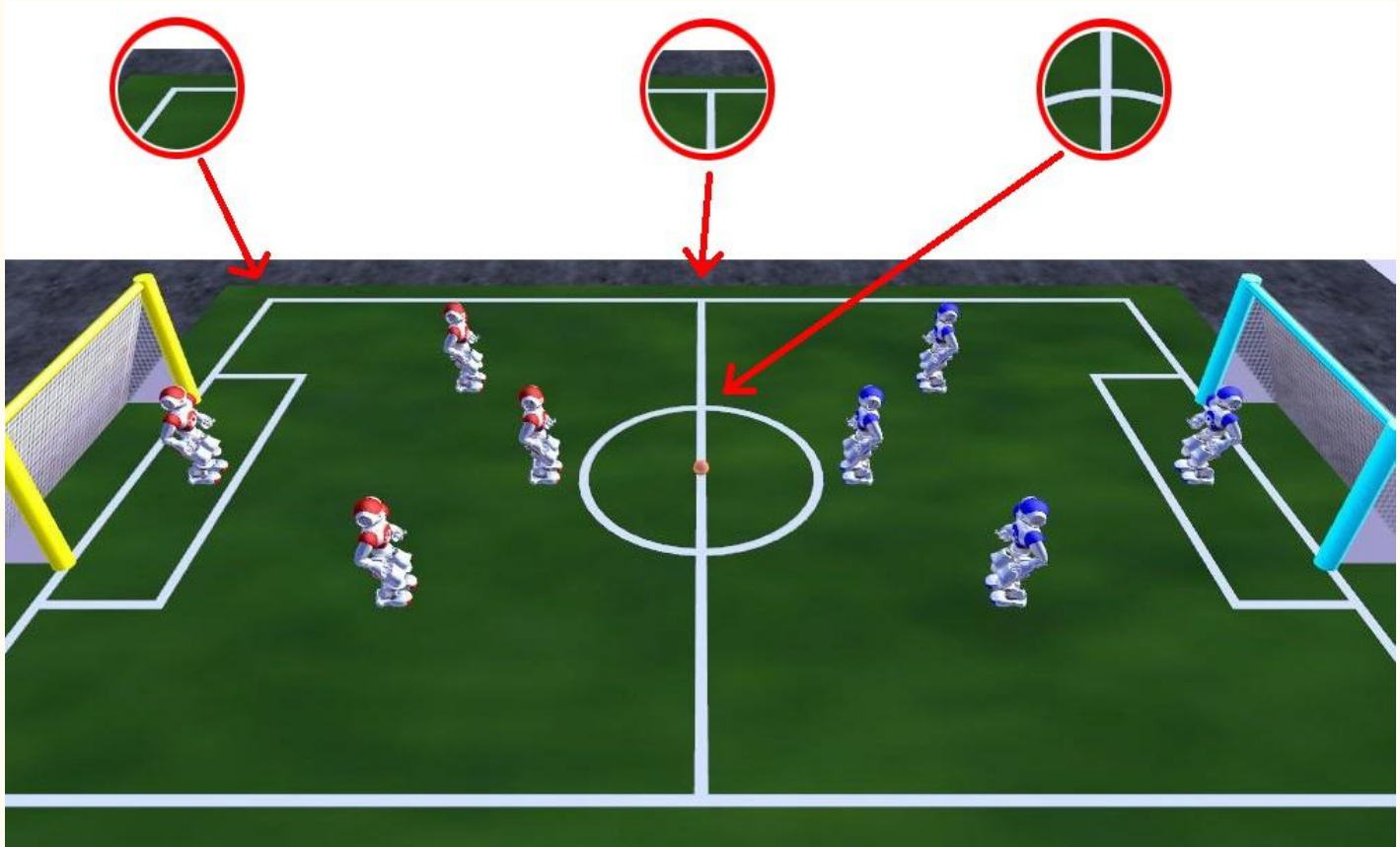
Line Intersection Detection for Robocup

—

By Fabien SANTOS-CESSAC, Clément BIN and Louis-Marie DALLEN

Objective

Detecting the 3 types of intersections



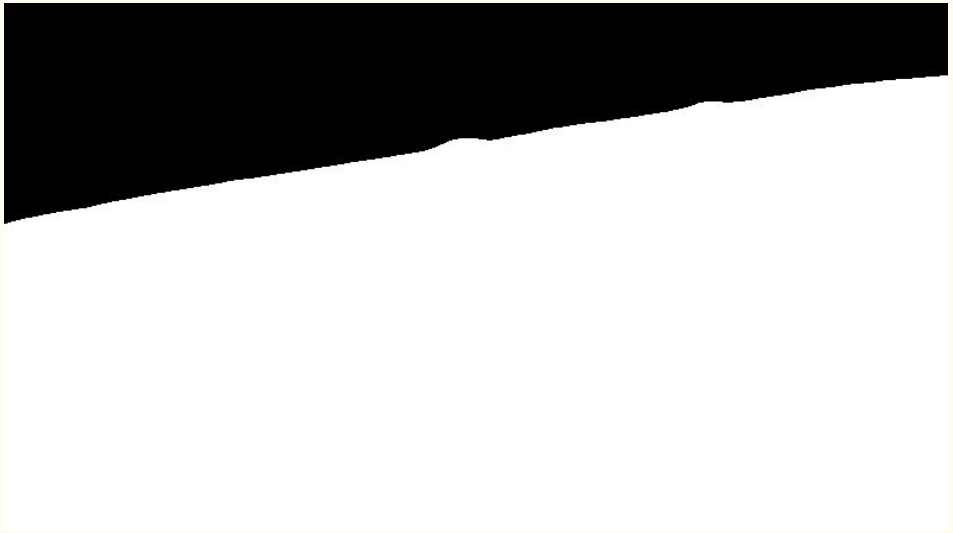
Plan

1. Noise Filtering
2. Line Detection
3. Intersection Detection and Identification

Noise Filtering

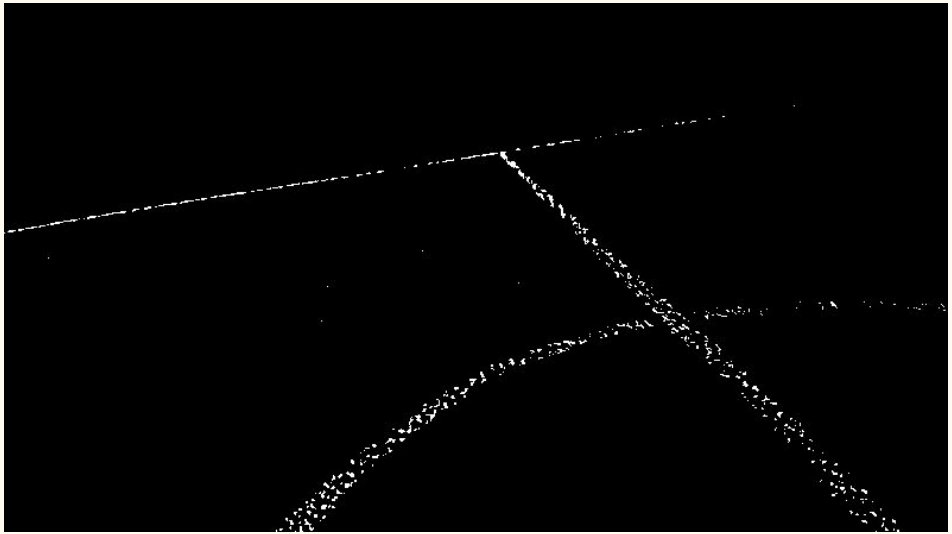
Background Deletion via Pitch Mask

Selection of green zones via a selective HSV filter followed by a median blurring filter and a small erosion.



RGB Threshold

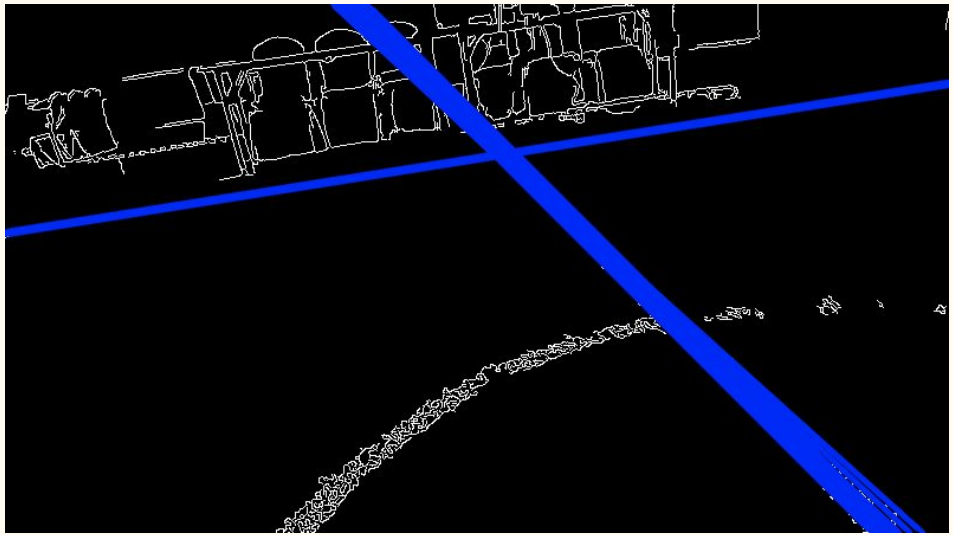
Keep only white via a selective RGB filter.



Line Detection

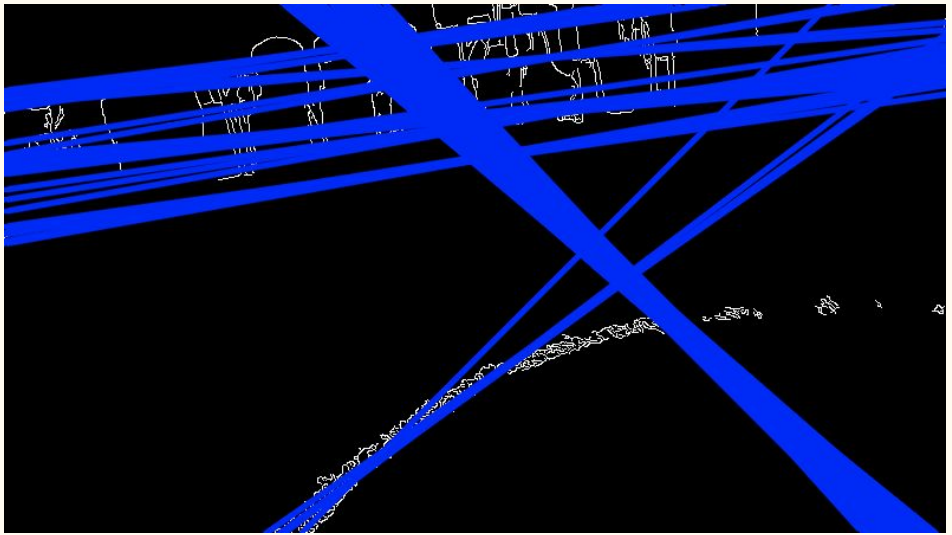
Hough Lines Function

The standard Hough Lines function from OpenCV draws full lines.



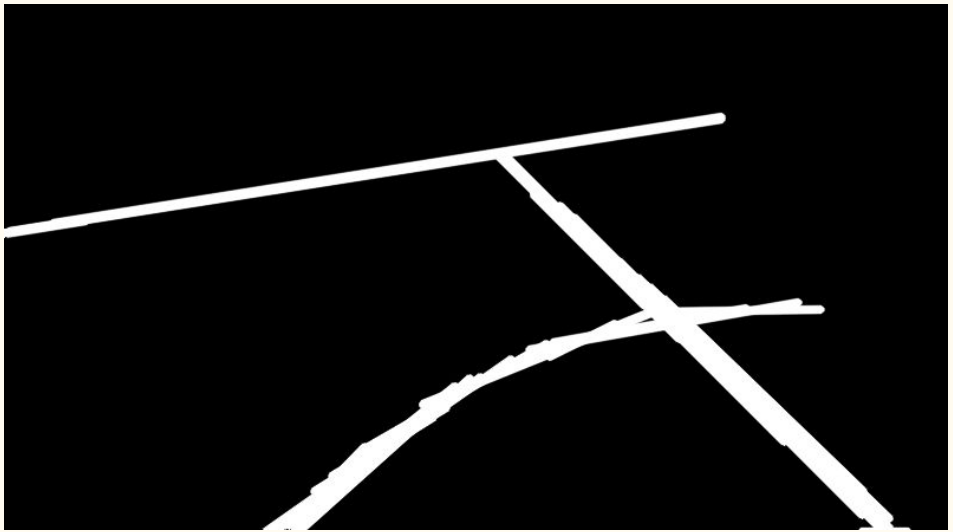
Hough Lines Function

But it has some drawbacks…



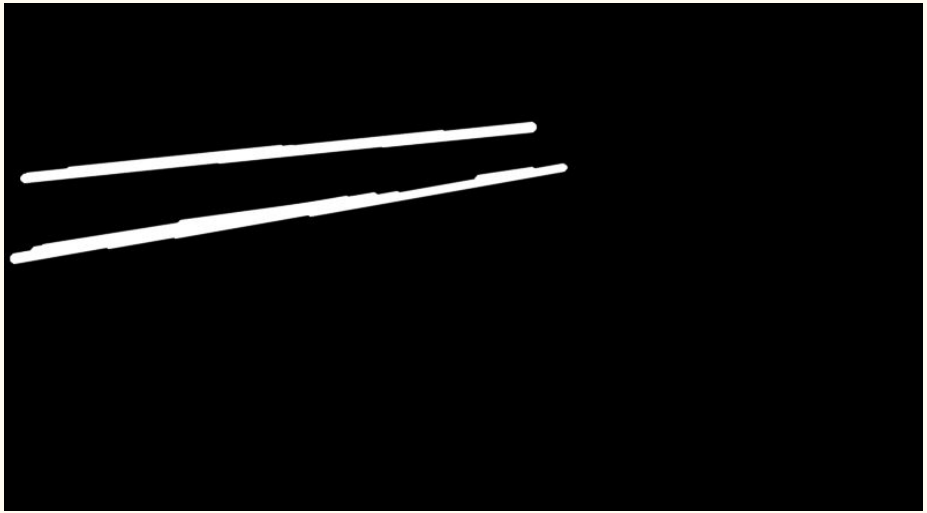
Probabilistic Hough Lines Function

The probabilistic Hough Lines function from OpenCV helps draw visible line segments.



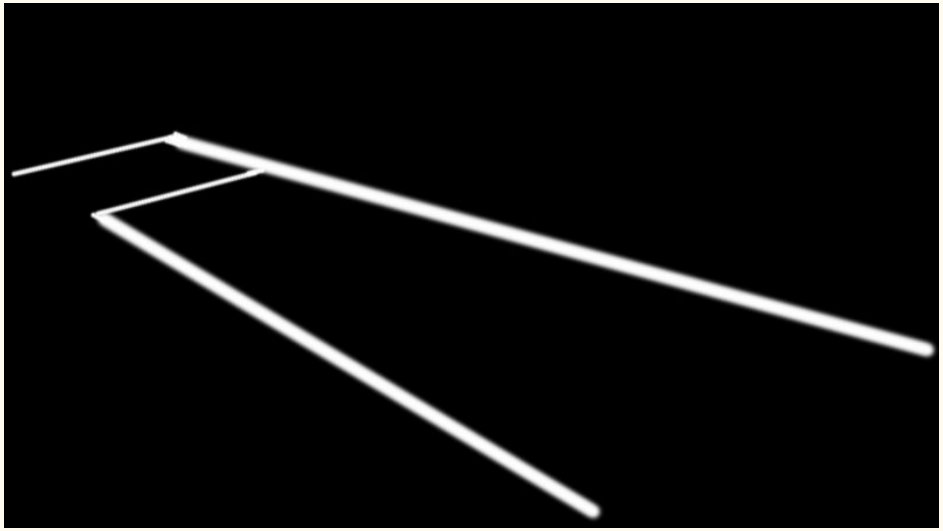
Probabilistic Hough Lines Function

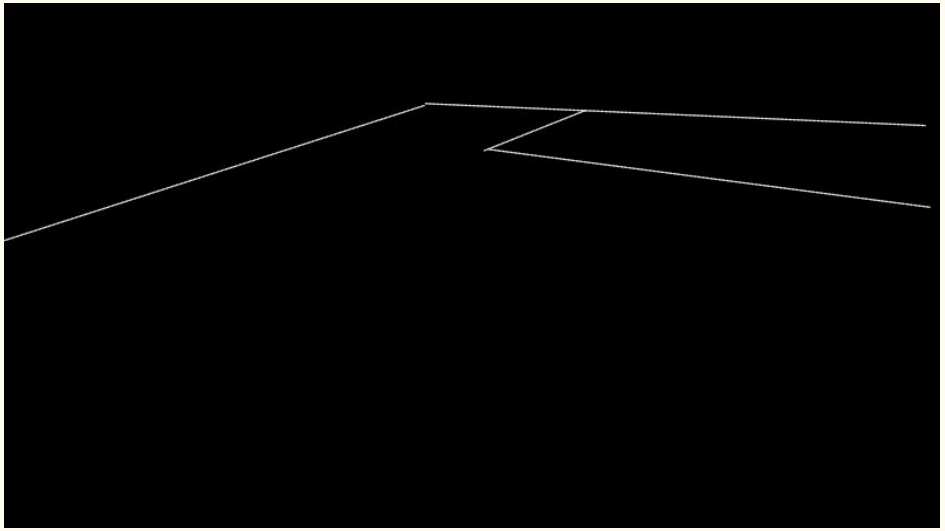
The limits of Hough Lines transform:



Alternative Method

Do nothing: assume that the line detection team provides clean pictures of the lines.

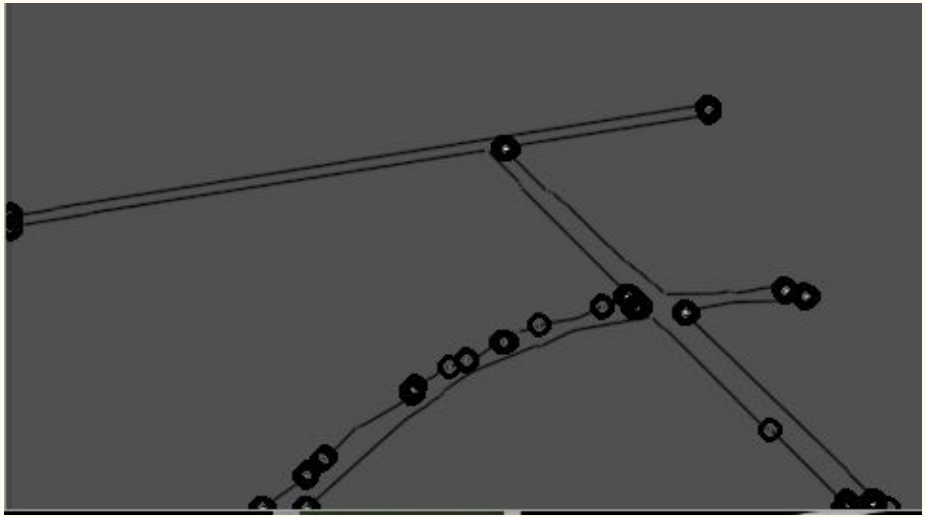




Intersections

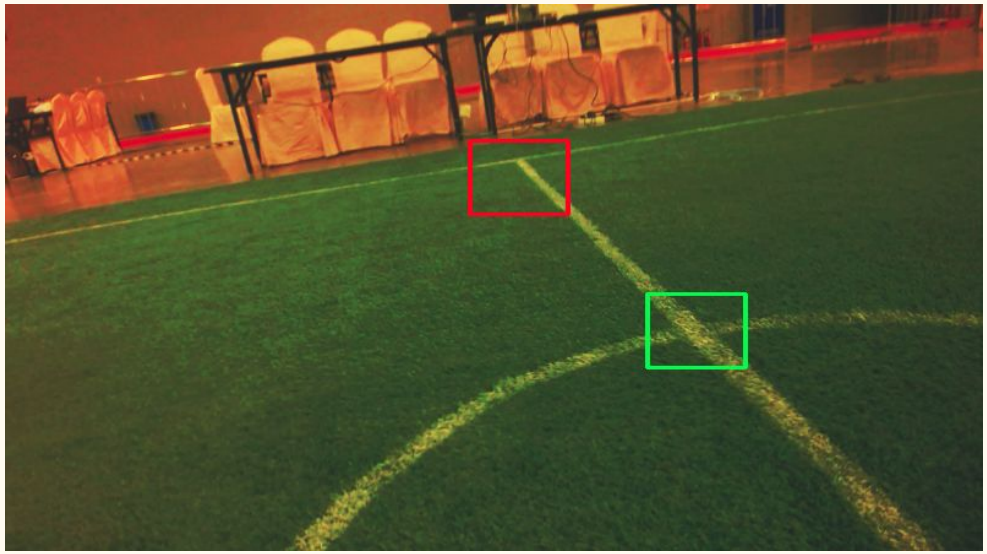
Harris Function

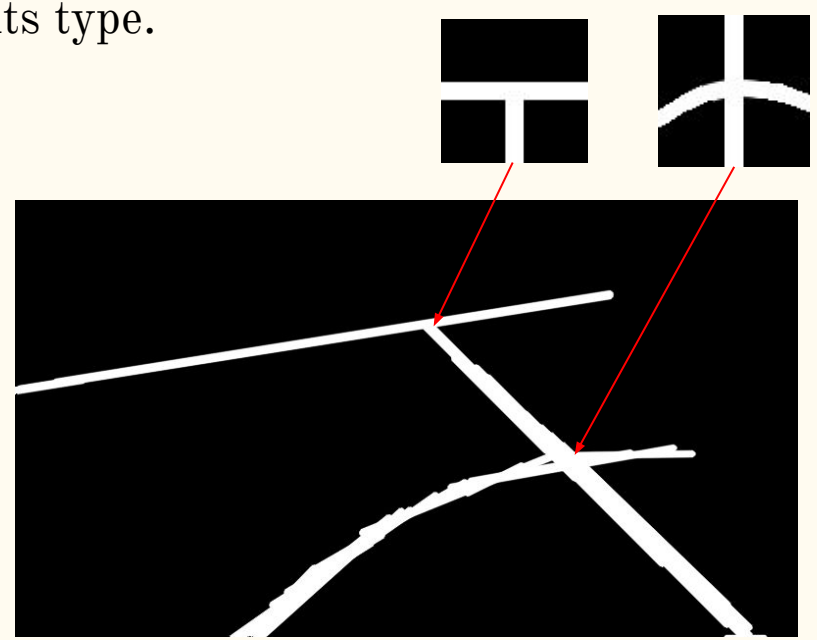
Harris detects feature points. However it requires clean lines otherwise the results are pretty messy.



Pattern Matching Algorithm

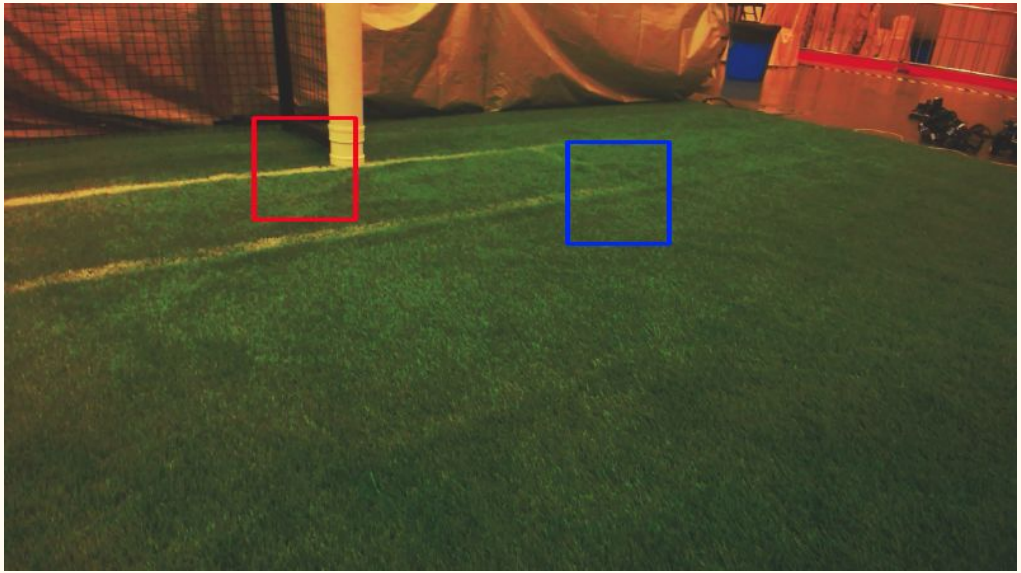
Pattern recognition function tries to find instances of predefined patterns in the image. This shows both where the intersection is and its type.

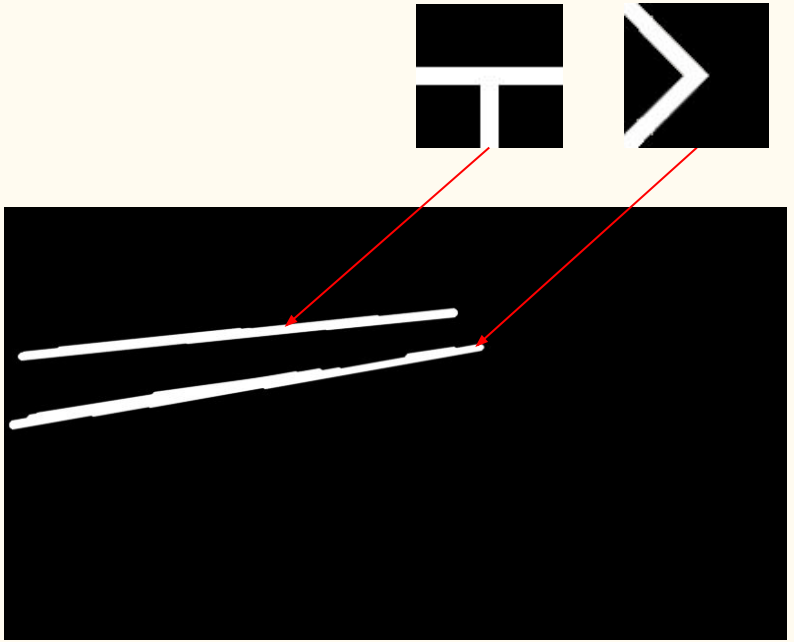




Intersection Detection and Identification

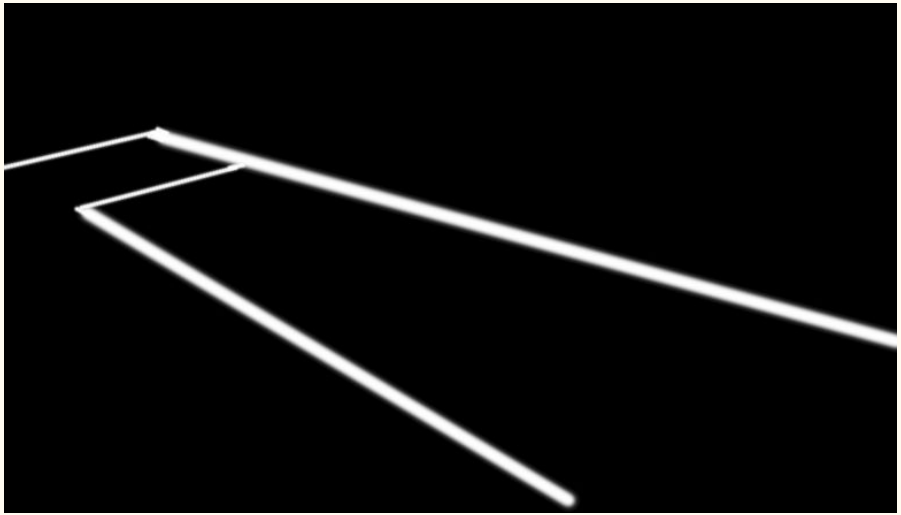
Detection depends on good line detection.

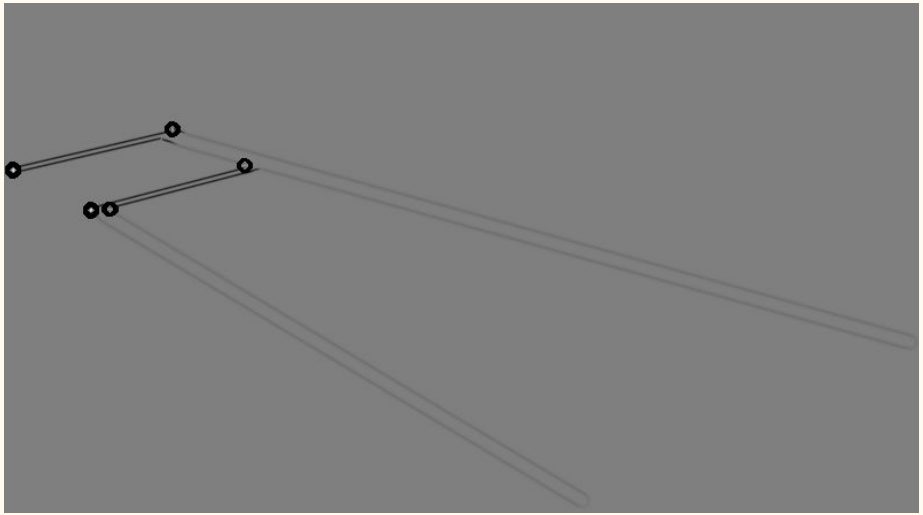




Harris Function

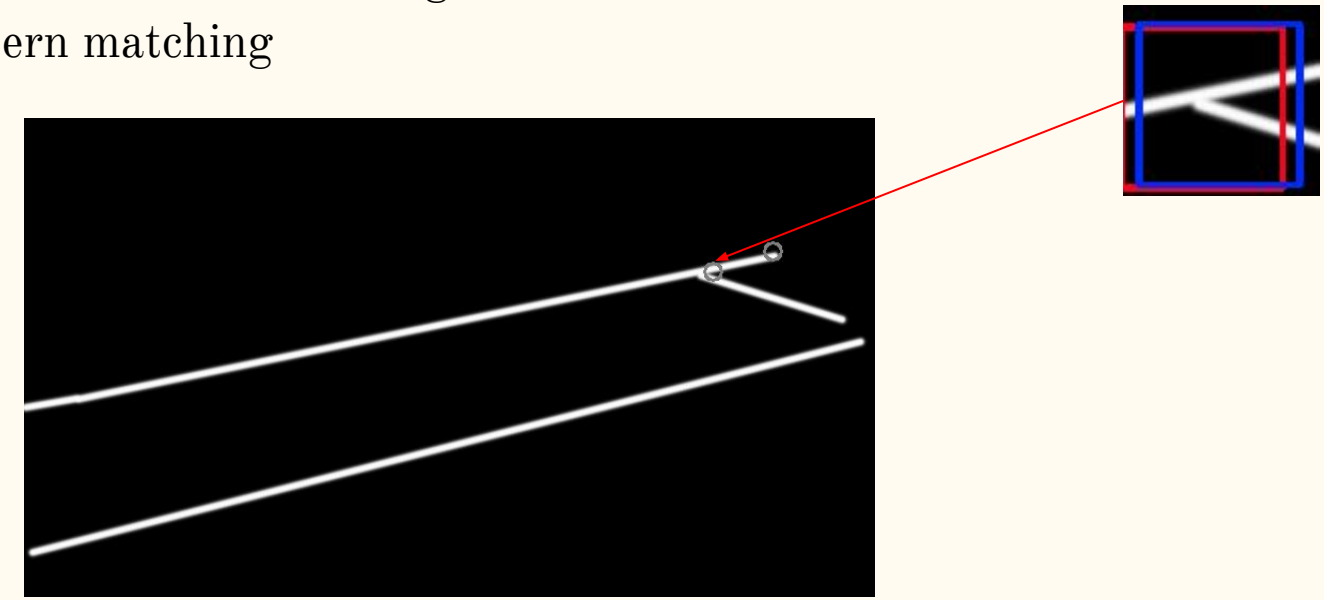
Assuming we get clean pictures of the lines from the previous team, the Harris function works quite well.





Harris Function

The program focuses on these zones for further processing:
-removal of errant line segments
-pattern matching



Conclusion

We did the job of half the other teams in order to make our intersection detection program.

You're welcome Robhan, now go get us a Robocup ummm… cup.