Cairo University  
Faculty of Engineering  
Computer Engineering department

**Image processing project**

**Tennis Motion Tracker**

Team #13

Team members:

Rania Gamal

Mai Mohamed

Mayar Ahmed

Maryam Mohamed

The proposed system tracks the motion of tennis players, and visualizes this motion to be able to spot the pattern of movement of players and see if they were taking offensive or defensive positions during the points analyzed.

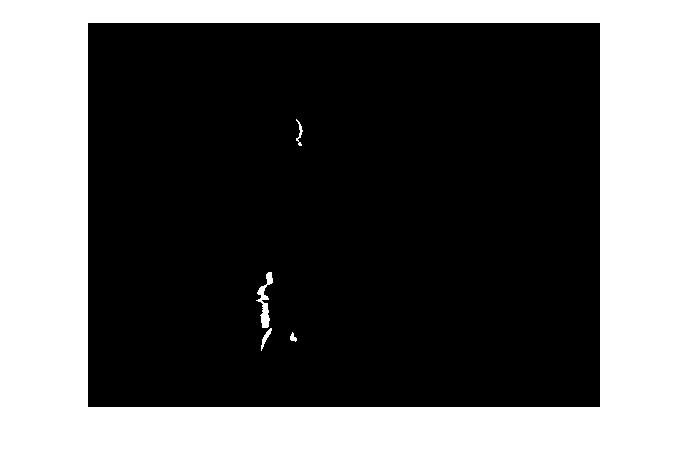
The main goal of the system was the motion tracking of the players throughout the point. This could be achieved with different approaches such as optical flow, Kalman filtering, particle filters, background subtraction and frame differencing. Given the fact that the camera is almost steady during the points analyzed, and that there are only two objects to focus on during tracking there was no need to use complex statistical models to track the players. Our tracking was based on Frame differencing and blob analysis.

The algorithm steps are the following:

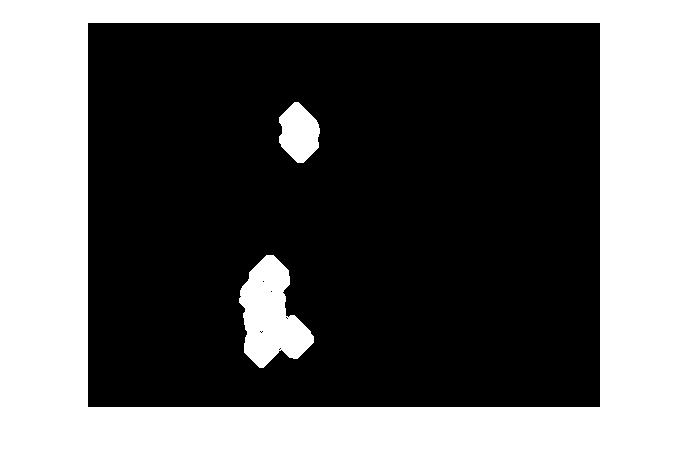
1. Frame differencing
2. Noise removal
3. Morphological operations
4. Blob Analysis
5. Detecting players
6. Visualize moving pattern
7. Frame Differencing: Since the camera is steady, subtracting each video frame from the previous will give us the movement in the image, however there will be noise coming from movement of undesired small objects in the scene (example: small movement in the crowd)



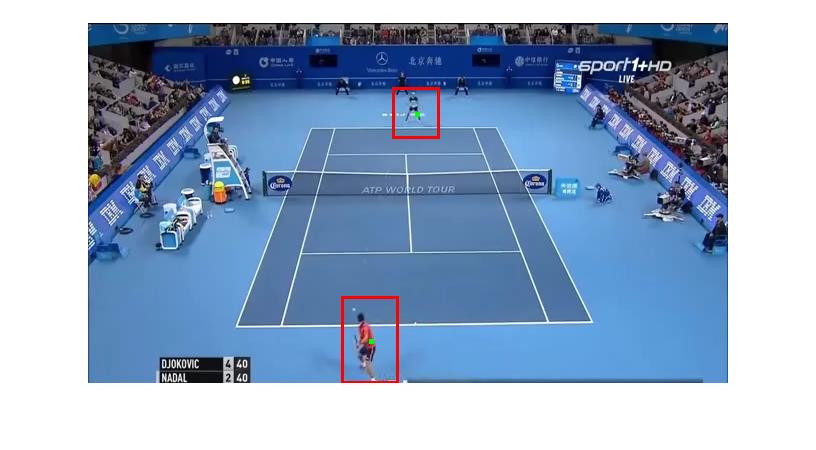
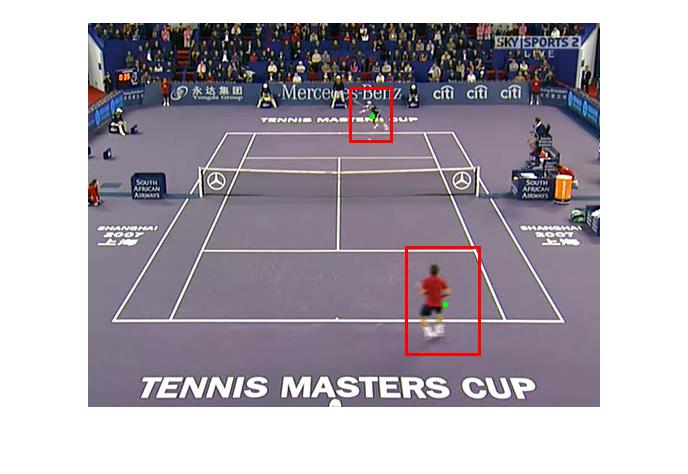
1. Noise Removal: After frame differencing, we apply a threshold on the resulting difference and convert it to a binary image, furthermore objects with small number of pixels after thresholding are eliminated. This will leave us with the biggest moving objects in the scene.



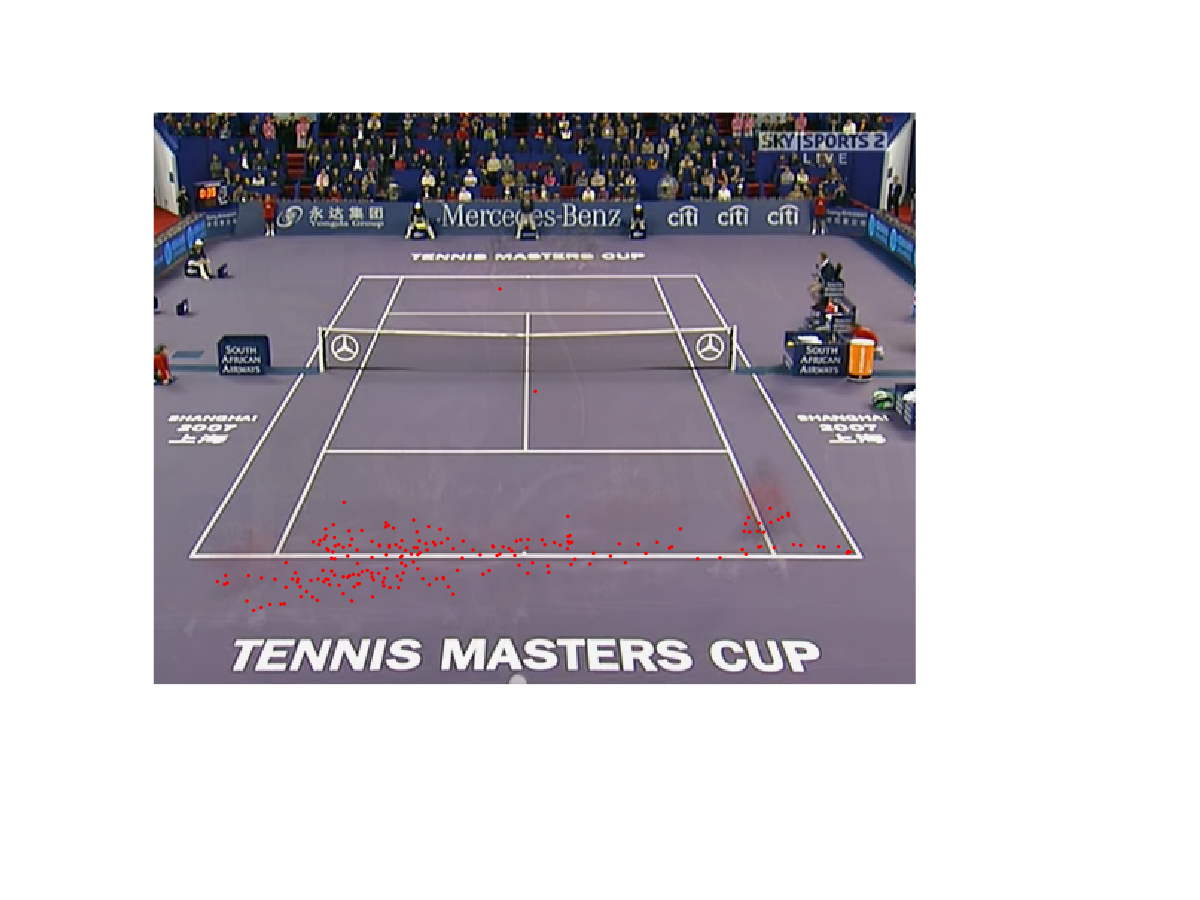
1. Morphological operations: The resulting objects are not very clear and the pixels of the same object are disconnected, so we apply morphological operations to fill the gaps and make the detected players more clear. After multiple trials of different morphological operations, dilation followed by thickening and connecting components was the best option to make the detection clear.



1. Blob Analysis: blobs are areas with higher intensities than the rest of the intensities in the image. After morphological operations, the detected blobs are analyzed using MATLAB Blob Analysis object. It computes the statistics of the given blobs across the image such as minimum area, connectivity. After that it returns the centroids of the blobs (the moving objects in our case) and their bounding boxes.
2. Detecting Players: After getting the centroids and the bounding boxes of the blobs, these boxes are inserted in the frame and displayed.



1. Visualizing moving pattern: At the end of the point, all detected centroids from player movements are saved. The court is extracted from the scene by taking the average of multiple frames, after that the movement of the player in front of the camera is plotted on the court showing his overall pattern throughout the game.



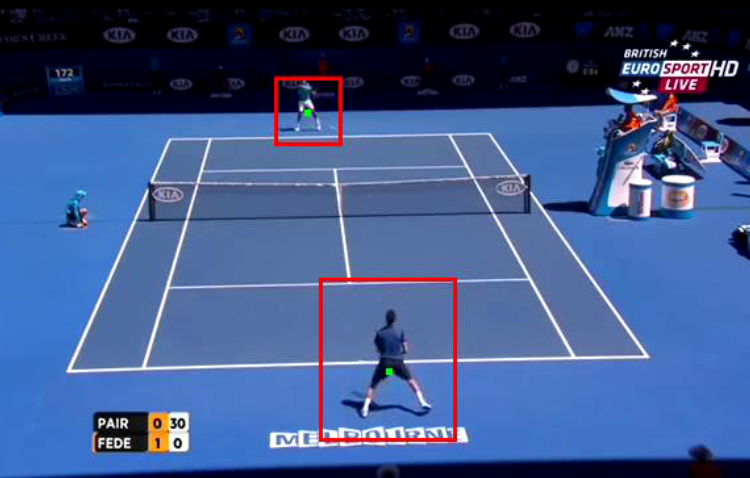
The previous will work in the ideal case where only the players are moving, the camera is steady the entire game, the player far from the camera is clear enough to be detected. Here are some things that were taken into consideration to improve the accuracy detection:

* Small movements from the sides of the court will be detected as it’s large enough not to be dismissed as noise. To exclude as many of these blobs as possible, a constraint was put on the minimum area of the detected blob. The player blobs are usually the largest in the scene, so if the detected blob doesn’t exceed this minimum area it probably means that it was not of a player.
* Noise removal isn’t perfect. If there’s still some noise, then after running morphological operations they will be enlarged and might be counted as a blob in the frame. To increase the accuracy, the number of connected components (number of blobs) in the frame is computed, if it exceeds a certain range, then we skip this frame as it this small noise won’t probably last for the next frame.
* If the camera moves then frame differencing will result in a very large number of blobs as the entire scene is shifted not just the players, so we skip some frames in the video until the camera stabilizes again (until number of blobs is within the accepted range).

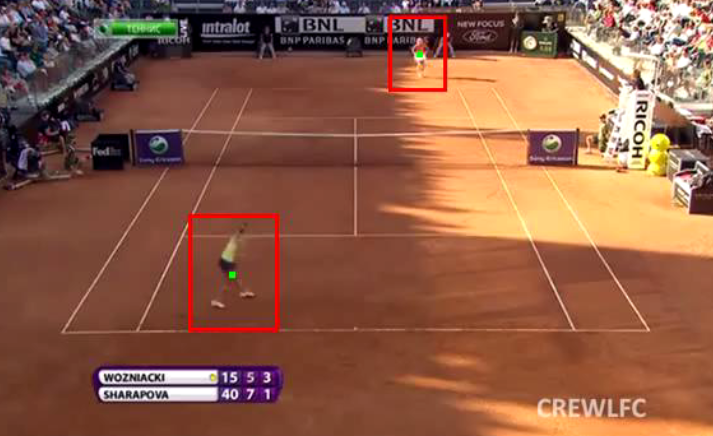
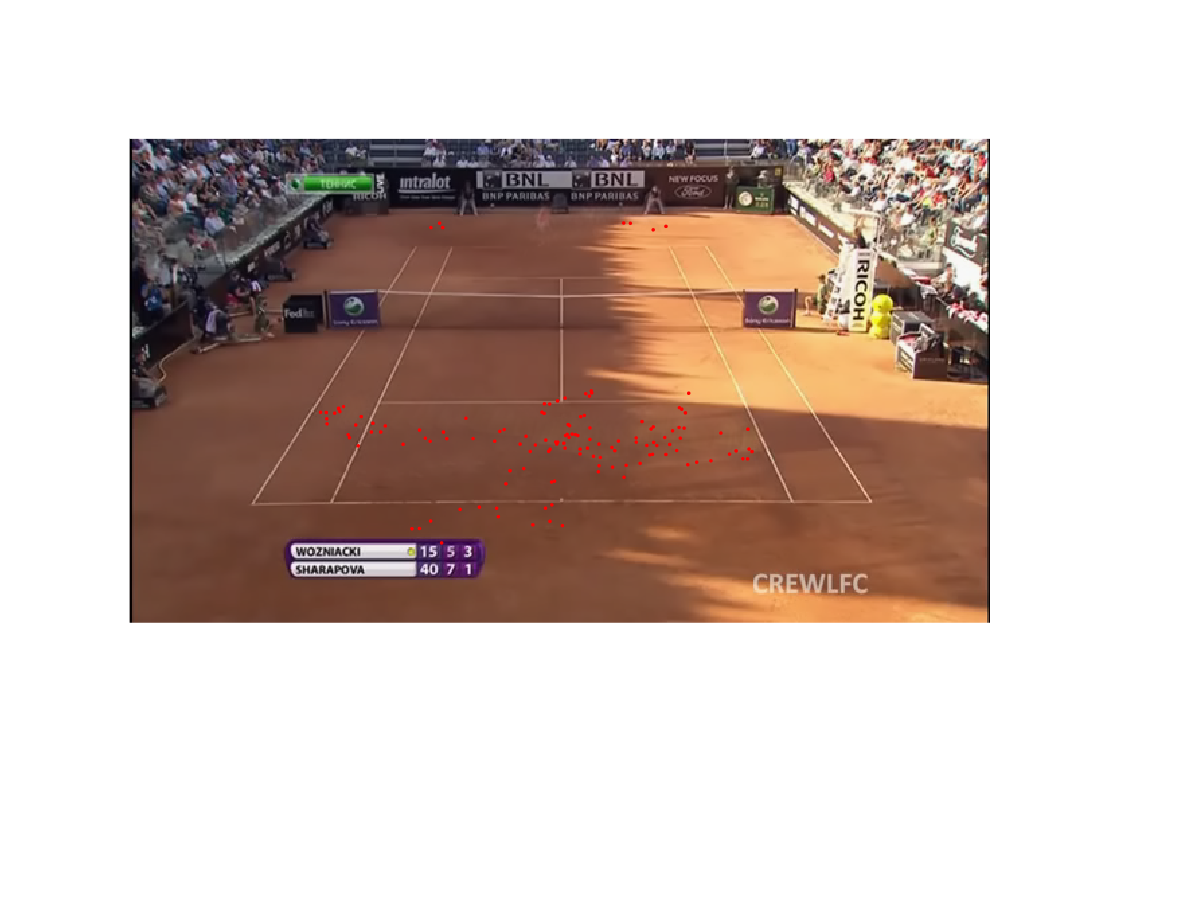
Results:

Running the algorithm on different videos with different camera position and light intensities









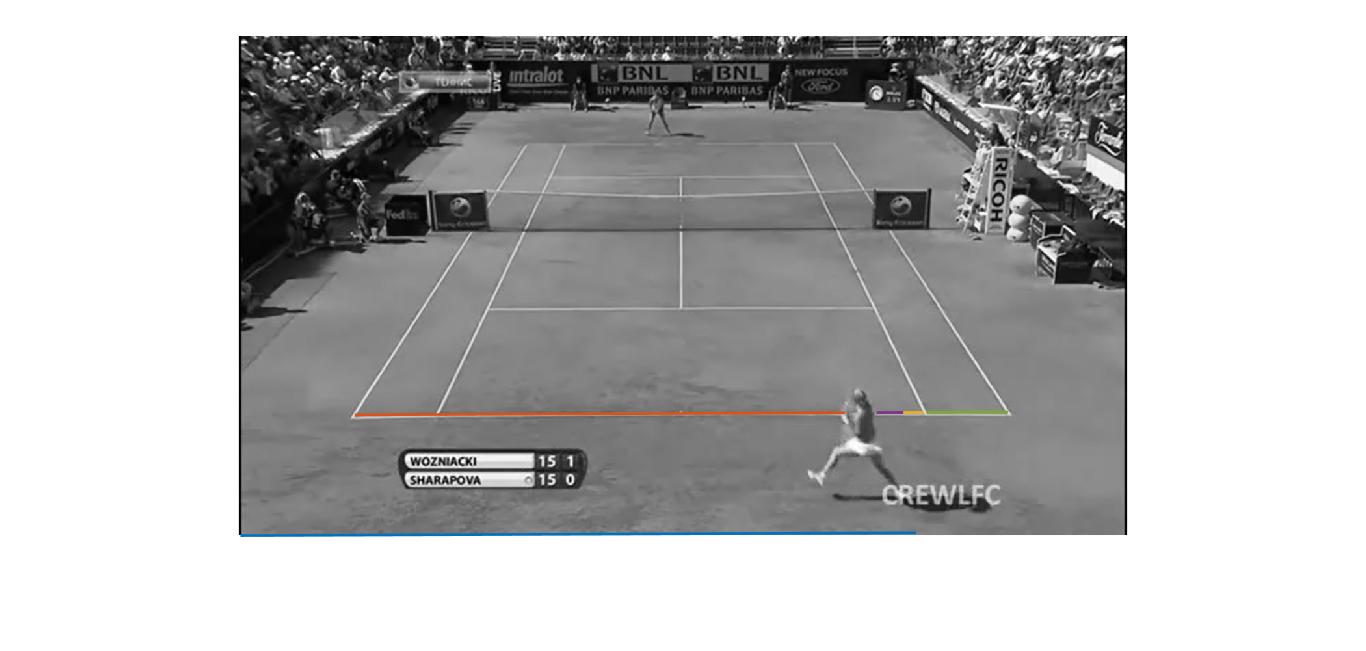
* Some comments on the accuracy of the detection:
* The player in front of the camera is detected with a high accuracy.
* The player far from the camera will have a small blob and might be dismissed if the camera is too far because of the constraints we’ve put on the area. But in most of the cases they’re detected correctly and if it’s dismissed then it will only be for a very small time before it’s detected again.
* The constraints put on the area and the number of detected blobs are not guaranteed to work 100% of the time, because of the different properties of different courts and scenes, but it does improve the accuracy and remove a lot of unwanted objects.
* The shadow of the player moves as well each frame, it will only affect the width of the bounding box of the player as the shadow is considered as part of the movement. but the position of the player remains the same.
* If a linesman moves behind the player and the player isn’t moving, their blob will be large as well, they might be mistaken as the player.

For Further analysis, we tried to extract the exact positions of the lines in the first half of the court and calculate the percentage of the points in each region (behind the baseline, between the baseline and the service line)

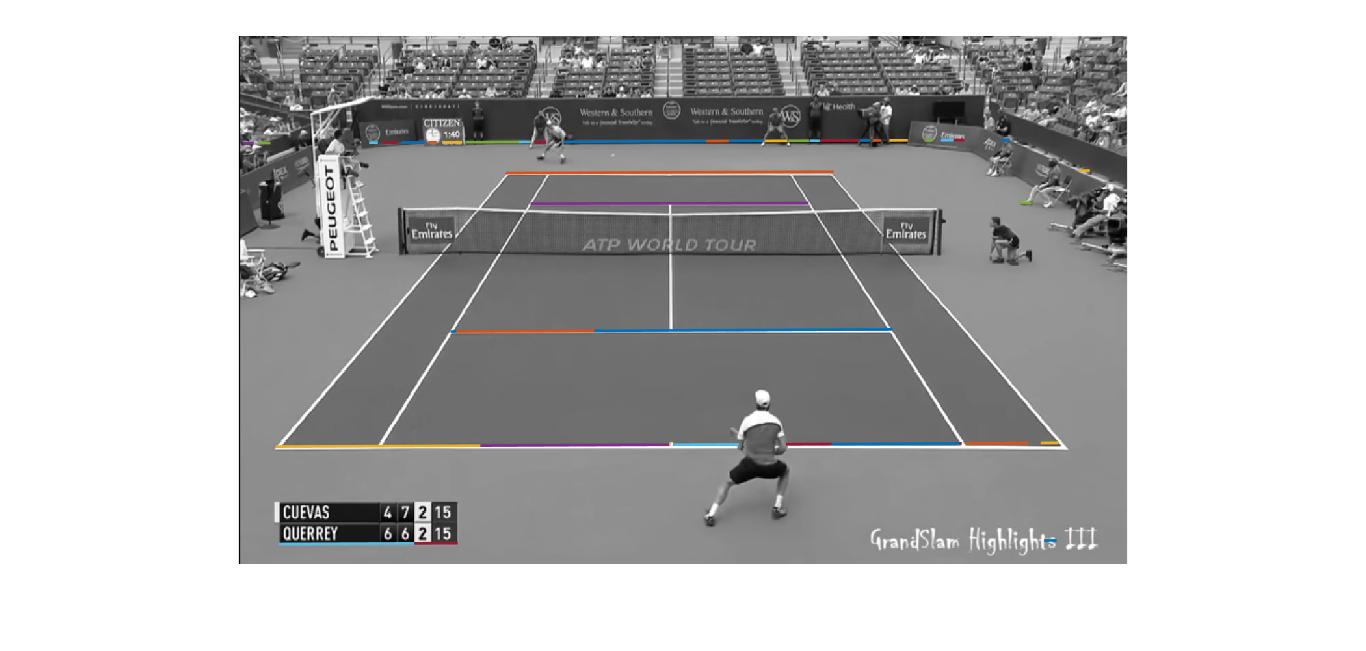
The problem was to detect the court lines:

First solution was to use hough lines but the problem with the results was

1. It doesn’t detect one line as a single line but as multiple smaller lines
2. A lot of noise and unwanted lines were detected
3. In some images where the light is different or the resolution is less, it didn’t detect any line or only one line

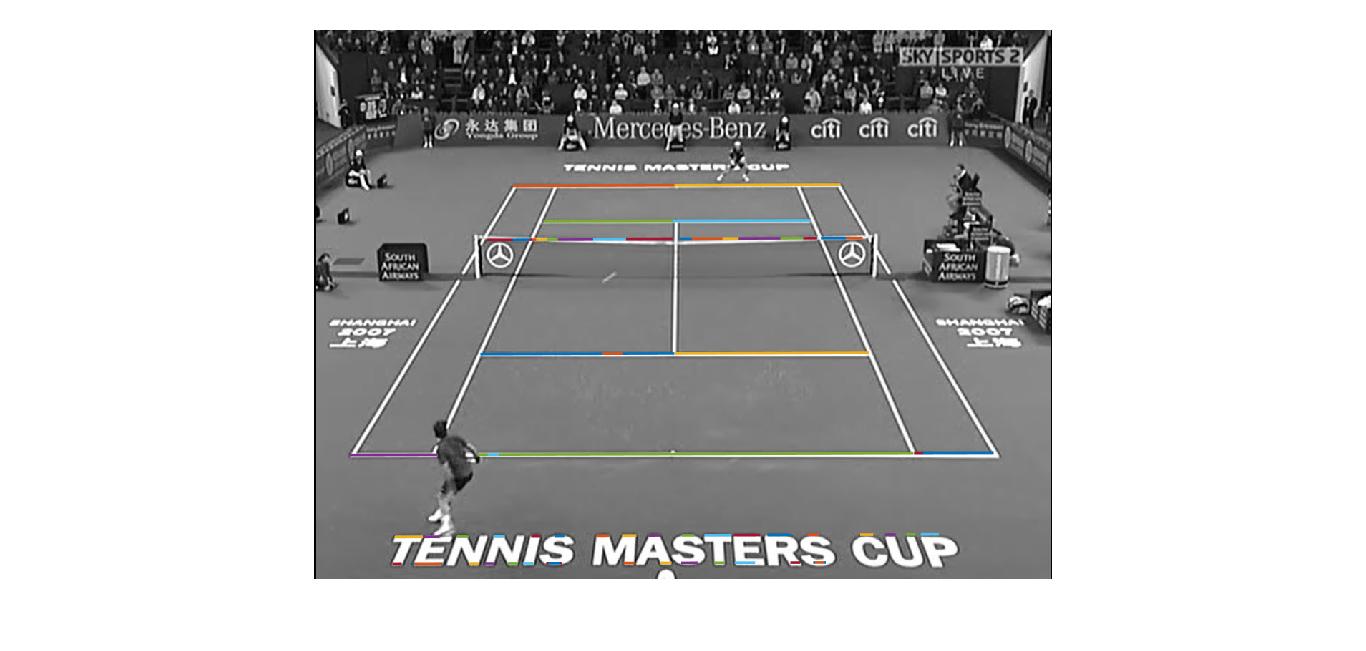






In order to remove the noise, we applied a mask on the image before hough lines using color thresholding tool from MATLAB which creates a mask function based on some modification we make on the image

Doing that resulted in less noise but also made more problems:

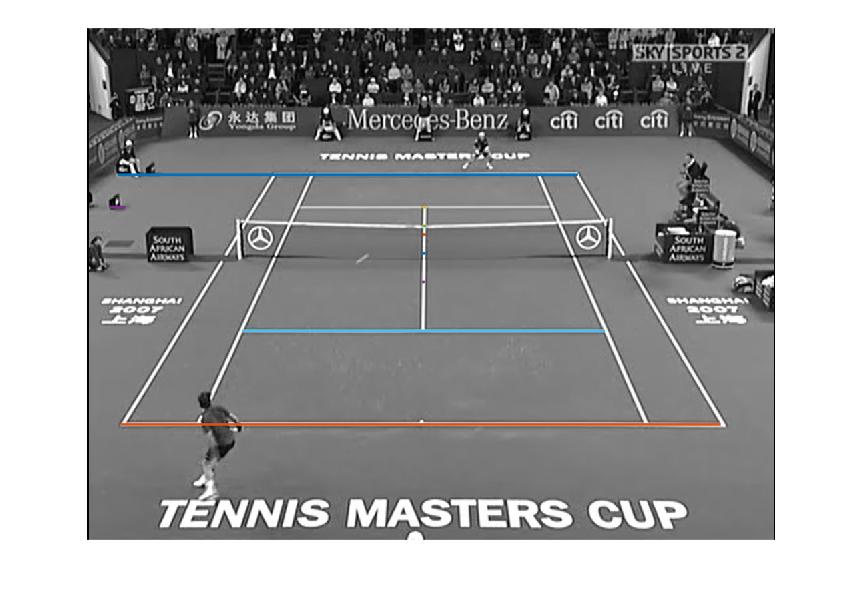
1. The court lines weren’t always detected
2. The noise made by text on the ground were enhanced so they were detected as lines
3. The mask didn’t work for all the images



In order to make the lines from hough lines into one single line we made a function that makes a single long line from separate lines if they have a slight difference in y coordinate

The problem with this was:

1. The noise lines made from detecting the text and other noise on the left and right of the field became a single long line as they had the same y coordinate
2. The threshold that was set didn’t work for all images with different resolution
3. The result wasn’t predictable



Using Canny before hough lines only made it worse as a lot of lines weren’t detected and canny detected a lot of noise in the crowd behind the court and the people.

Some results after getting position of service line and baseline and calculating percentage of points in each region.

