Court Detection

Using Deep Hough Transform Line Priors

Basem Shaker[a]

[a]*Department of Industrial Engineering, University of Trento, Italy.*

July, 2022

Abstract

Tennis court detection is beneficial in highlight extraction and score tracking. Hough Transform is commonly used for line detection. The objective of this project is to employ a Deep Hough Transform Line Priors Pre-trained LCNN [1] to detect court lines. Filtering the lines post detection is necessary to identify the true bounding box of the court and to calculate the homography matrix. The homography matrix can be utilized to warp the perspective and deployed in various other applications such as score tracking or player movement behaviour during the match.

1. Introduction

Deep Hough Transform Line Priors is a technique that combines deep learning methods and knowledge based line segment detection. Hough transform provides the prior knowledge about the global line parameterization, while the convolutional layers learn the local line features [1]. Figure 1 shows the result of applying the Deep Hough transform CNN on a still picture of a tennis court.

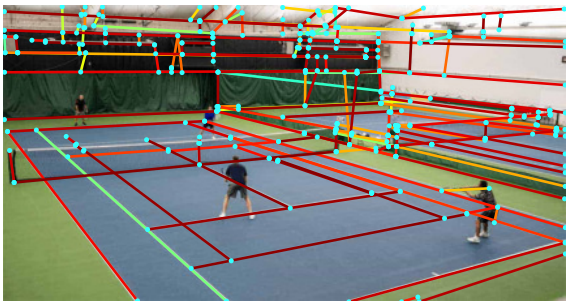


Figure 1: Applied Deep Hough transform line priors CNN on a tennis court

Detected Hough lines can be used to find the vanishing points. The homography matrix is then calculated and could be used in further tracking of player and ball relative locations, or project messages on exact locations such as advertisements.

1.1. Related Work

To tackle court detection, [2] presented a generic algorithm that can be adapted to any sport in which the court consists of a sufficient number of straight lines. The camera calibration has eight degrees of freedom and requires at least two horizontal and two vertical lines for proper calibration. They used a trivial classification for horizontal and vertical lines that used the line angle threshold. All combinations of each two horizontal and vertical lines were first checked by a test that rejected any homography matrix that displayed a Non-isotropic scaling. Later they evaluated the models by projecting the court onto the source and verifying that the model accurately covers the white pixels.

In [3], they detected the court lines using image binarization, then applied Hough transform to find straight lines that are candidate for the court line. The outermost four lines were selected using relative position and length of each straight line.

Edge detection was carried out on the binary image in [4]. They used Sobel operator to detect edges in horizontal and vertical directions, and later used Hough transform to find the lines. They relied on the fact that the court will always be in the center to filter out any outliers. The line locations and angles reflect characters that are specific to tennis courts. A Support Vector Machine (SVM) was trained on court and non-court shots to identify if there is a court currently in view.

1.2. Problem Description

Multiple Hough transform lines lie outside of our area of interest. These lines need to be filtered out to identify the true court borders. Multiple methods will be utilized and are explained in section 2 below.

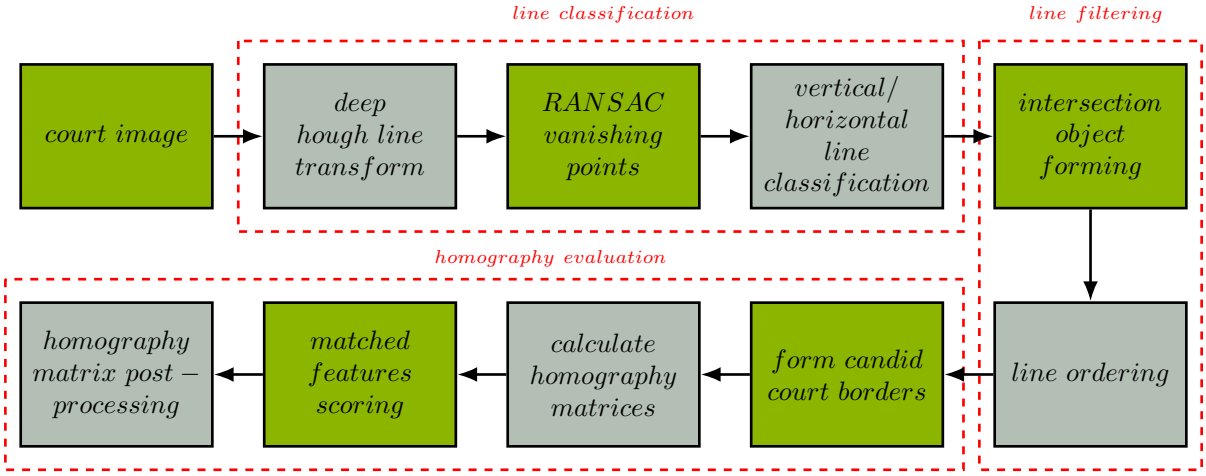


Figure 2: Process Pipeline

2. Method & Theory

This section explains theory behind the procedure of the algorithms. Figure 2 shows a high level view of the process pipeline. The following paragraphs are a brief description of the pipeline, which can be combined into three main operations, line classification, line filtering, and homography evaluation.

The process starts with the court image as an input to the Deep Hough Transform Line Priors Pre-trained LCNN. The output is an array of candid Hough lines along with their score. The lines are utilized to form a hypothesis about the location of the 3-point perspective vanishing points through a RANSAC algorithm. The vanishing points are later used to classify the lines into their respective orientation, which falls under two horizontal and one vertical set.

The vertical lines are discarded and the intersection points between the two horizontal sets are calculated. The lines with no intersection are ignored. Lines that share intersection points are all merged into objects. The object with the most amount of lines is assumed to be the Region Of Interest (ROI).

Inside the ROI, the horizontal lines are ordered within their orientation classification from left to right. To reduce the required number of calculations, the ordered lines are filtered, only leaving a percentage of the outermost lines. The two horizontal sets are then coupled to form a four line candid sets of court borders.

The four intersection points of each court border candid set is calculated to generate a homography matrix using OpenCV. Each homography matrix is applied to warp the image perspective, and the result is scored based on number of matched features with a tennis court reference. The homography matrix with the most matched features is chosen and post processing is performed to apply the proper scaling and guarantee the image is within frame after perspective warping.

2.1. Deep Hough-Transform Line Priors

Current state-of-the-art line detection methods rely on deep learning models powered by massive datasets. The amount of needed training data can be significantly reduced by adding global geometric line prior knowledge to deep networks.

The algorithm relies on a $\mathcal{HT} - \mathcal{IHT}$ block that consists of a Hough Transform ($\mathcal{HT}$) and an Inverse Hough Transform ($\mathcal{IHT}$). The block combines local learned images features and the global lines priors. Given an image line in polar coordinates, with an offset of ρ and angle θ, a point along this line can be calculated by using equation 1. [1]

$$(x(i), y(i)) = (\rho \cos\theta - i \sin\theta, \rho \sin\theta + i \cos\theta) \quad (1)$$

Traditionally, Hough transform relies on binarizing the featuremaps. However, with deep hough transform, the featuremap activations (F) are accumulated instead for each certain line bin.

$$\mathcal{HT}(\rho, \theta) = \sum_i F_{\rho,\theta}(x(i), y(i)) \quad (2)$$

The gradient of the $\mathcal{HT}$ layer is only a mapping from Hough bins to pixel coordinates in the

featuremap (F). The pixel coordinate can be calculated as the average of all bins where the pixel has voted as shown in equation 3. [1]

$$\mathcal{IHT}(x,y) = \frac{1}{N_\theta}\sum_{\theta}\mathcal{HT}(x\cos\theta + y\sin\theta, \theta) \quad (3)$$

2.2. Vanishing Points & RANSAC

A vanishing point is at which receding parallel lines viewed from a certain perspective appear to converge. The number of vanishing points defines the type of perspective. Figure 3 shows a 3-point perspective which is considered in this project. A RANSAC algorithm can be used to estimate the location of those vanishing points. [5]

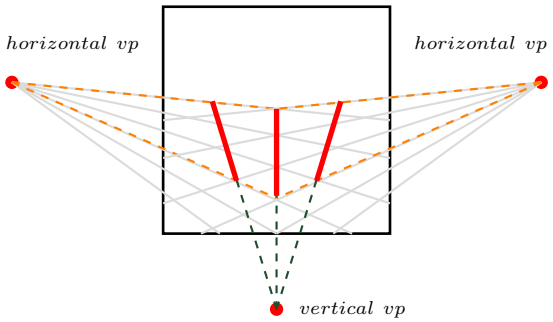


Figure 3: Vanishing point illustration

Random Sampling Consciences (RANSAC) is an iterative method which generates candidate solutions by using the minimum number of observations to estimate model parameters.

RANSAC can find the prevailing set of line directions. The algorithm is run three times, removing the inliers found in the previous run. Two lines are chosen randomly to generate a vanishing point hypothesis, which is their intersection point. The remaining lines vote based on the angle they produce when joined with the vanishing point. If it is below a certain threshold, the line is considered an inlier. The pseudocode is shown in Algorithm 1.

Algorithm 1 RANSAC

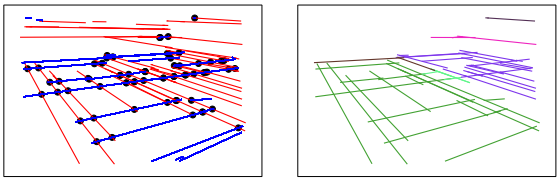
Input: Hough lines, iterations, angle threshold
Output: vanishing point hypothesis, inliers

```
procedure ESTIMATEVPS(H_lines)
    for i= 0 : iterations do
        choose L1, L2 ∈_R H_lines
        vanishing point hypothesis (vp) = L1 . L2
        form lines (LS3) bt. vp and H_lines
        magnitude(M)=norm(LS3)*norm(H_lines)
        θ = cos^-1(LS3 * H_lines)
        inliers = θ < angle threshold
    end for
end procedure
```

2.3. Line Filtering

The outputs from the RANSAC algorithm are classified into two sets of horizontal inliers and one set of vertical inliers. The vertical set is discarded. The intersection points between the two horizontal sets can now be calculated.

The lines with no intersection are removed. The lines that share intersection points are all merged into objects [an object has the same color in Figure 4(b)]. The object with the most amount of lines is assumed to be the ROI.



(a) before forming (b) after forming

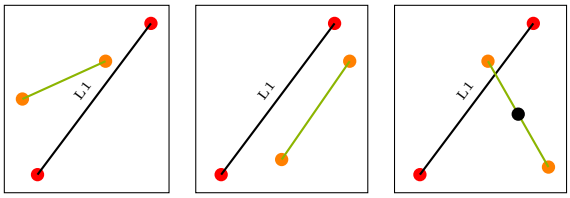
Figure 4: Object forming based on line intersections

To further filter the lines within the ROI, the lines need to be ordered. For this to be achieved, the equation of line for each inlier was calculated and used to extend all lines to the image borders.

A custom sorting class that compares each line to the rest was created. If the points of the current line both lie to the left of the compared line, it returns true, if both are on the right, it returns false. If one point is on the left and the other is on the right, the compared line midpoint is chosen instead to compare if it is on the left or right as shown in Figure 5.

Equation 4 determines which side of the line L1 $= (x_1, y_1), (x_2, y_2)$ a point P $= (x, y)$ falls. If $d > 0$, then it lies on the left side, if $d < 0$ it is on the right side, and if $d = 0$, it lies on the line.

$$d = (x - x_1)(y_2 - y_1) - (y - y_1)(x_2 - x_1) \quad (4)$$



(a) on left (b) on right (c) mid-point

Figure 5: Line ordering classification

The ordered inliers could be naively filtered for the minimum and maximum for both horizontal

and vertical lines to get the borders of the court. However, This of course would encounter major problems if the outside lines are not the true borders of the court. Instead, we would pick the top and bottom 20 % of each set for further analysis as shown in Figure 6.

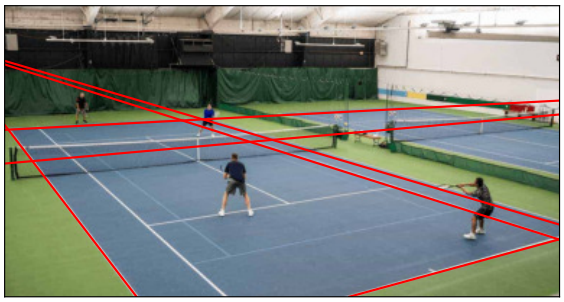


Figure 6: Filtered lines furthest out

Each pair of horizontal lines are combined with a pair of vertical lines to form a court border hypothesis. The list of border hypotheses are used to calculate multiple homography matrices that are going to be scored.

2.4. Homography Evaluation

A Homography matrix is a transformation that relates points in one image to corresponding points in another. It is a 3x3 matrix with 8 degrees of freedom as it is scale invariant [Equation 5]. Homography transformation can be useful in augmented reality, perspective correction, and image stitching.

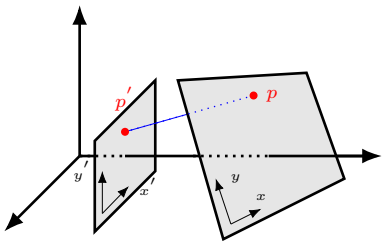


Figure 7: Homography matrix transformation. Adopted from [6]

$$s\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = H\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix}\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad (5)$$

The homography matrix maps points in a plane to points in the image, as there is no depth [Figure 7]. It can be estimated from four world points and their corresponding image points. A tennis court reference corners would be image points and world points will be the four points of intersection in each border hypothesis found in the previous step. [6]

To compute the homography matrix (H) with eight points (four pairs) $(p_1, p_1^{'})$, $(p_2, p_2^{'})$, $(p_3, p_3^{'})$, $(p_4, p_4^{'})$, a matrix system can be written that contains 2x9 matrices in the following form:

$$p_i = \begin{bmatrix} -x_i & -y_i & -1 & 0 & 0 & 0 & x_i x_i^{'} & y_i x_i^{'} & x_i^{'} \\ 0 & 0 & 0 & -x_i & -y_i & -1 & x_i y_i^{'} & y_i y_i^{'} & y_i^{'} \end{bmatrix}$$

The p_i matrices can be vertically stacked into a matrix (P) to compute $PH = 0$. To avoid getting an H matrix solution of all zeros, a constraint $|H| = 1$ is also added. Singular value decomposition (SVD) method can decompose the P matrix into the three matrices U, Σ and V^T, where H is the last singular column vector in the V^T matrix after reshaping it into a 3x3 matrix. [7]

$$P = U\Sigma V^T \quad (6)$$

The points on the plane $(p^{'})$ and the points on the court reference image (p) must follow the same ordering. For convention, clock-wise ordering from top left point was chosen. Algorithm 2 shows the ordering function used to achieve the clock-wise ordering of the points.

Algorithm 2 corner points ordering

Input: 4 un-ordered points
Output: 4 clock-wise ordered points

```
function pointOrder(pts)
    sort pts on x-axis
    lpts = two most left pts
    rpts = two most right pts
    sort lpts on y-axis
    tlpt [top left pt] = lpts[>y]
    dist = euclidean distance(tlpt, rpts)
    brpt [bottom right pt] = rpts[>dist]
    return [tlpt, trpt, brpt, blpt]
end function
```

Once the homography matrices have been calculated for all candid court borders, they are scored to find out the most accurate one. The scoring relies on the number of matched features between the reference court and the image after applying the warping dictated by each homography matrix.

To perform the scoring, a threshold is applied to the original image and any pixel values > 0 is pulled up to 1. The reference court image is then warped using the homography matrix and multiplied by the original image threshold. The summation of all the resultant values is the number of the matched features. The scoring also penalizes for the locations on the warped court that didn't

match any features on the original image threshold. The scoring calculations are shown in equation 7.

$$
\begin{aligned}
correct &= \sum P_{\{i,j\}} * P'_{\{i,j\}} \\
incorrect &= \sum (P_{\{i,j\}} * (P'_{\{i,j\}} - 1)) \\
score &= correct - (0.5 * incorrect)
\end{aligned}
\tag{7}
$$

Where $P_{\{i,j\}}$ are the values of pixels in the original image and $P'_{\{i,j\}}$ are the pixel values of the court reference. Figure 8 shows an example result of one of the homography matrices, where green shows the correctly matched features, and red shows the features that didn't match. Finally, the homography matrix with the highest score is chosen and used to warp the original image.

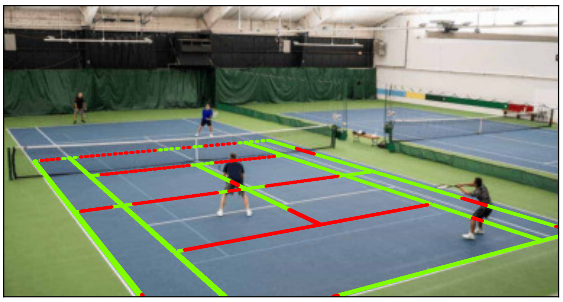
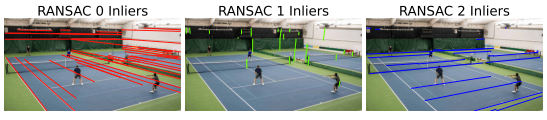


Figure 8: Scoring example of a Homography matrix

3. Results

This section will showcase some selected outcomes of the process explained in section 2 on three different images. The order of the section follows the same as the pipeline.



(a) Image 1

(b) Image 2

(c) Image 3

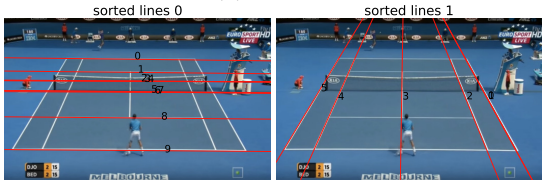
Figure 9: RANSAC vanishing point detection and line classification for 3 different images

Figure 9 shows the vanishing point detection using the RANSAC algorithm. It was successful in correctly classifying the lines into vertical and horizontal lines. From that point, it was simple to filter out for horizontal lines only.

The results of extending the lines to the image borders and ordering them are in Figure 10. This simplifies the filtering even further as we can now discard most of the lines closer to the middle.

(a) Image 1

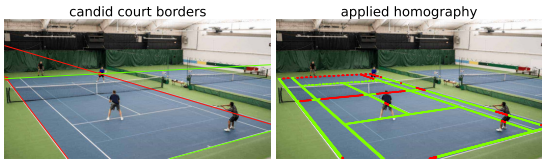
(b) Image 2



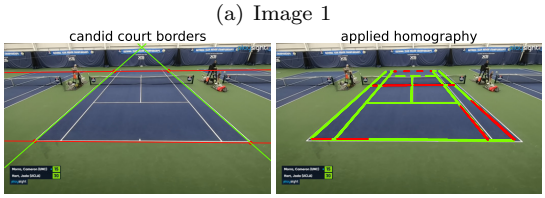
(c) Image 3

Figure 10: Ordered Line Results

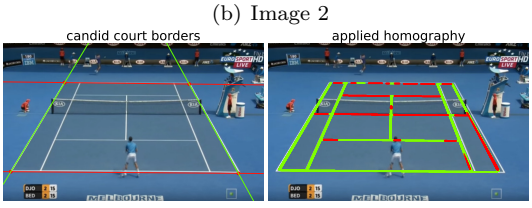
In Figure 11, candid court borders are tested. Once homography matrices are calculated, the warped court reference is scored on matched features.



(a) Image 1

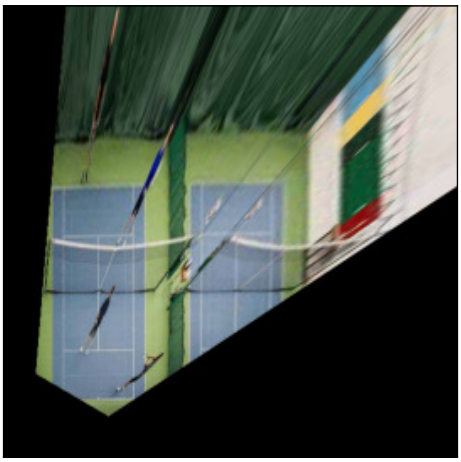


(b) Image 2



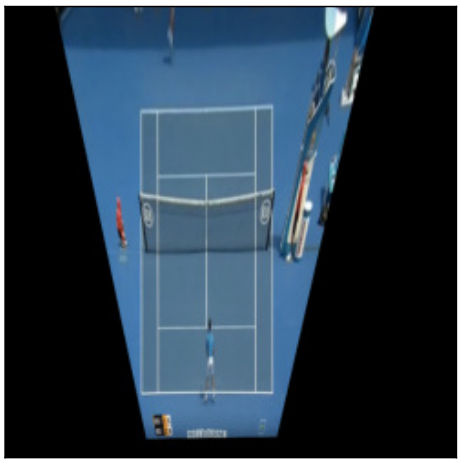
(c) Image 3

Figure 11: Homography matrix scoring process



(a) Image 1

(b) Image 2



(c) Image 3

Figure 12: Original image warping result

The court borders with the highest score is chosen. Figure 12 shows the warped original image for each tested court image after applying the chosen homography matrix.

4. Discussion & Conclusion

The objective of this project, which was to apply Deep Hough Transform Line Priors model has been achieved. The output of the algorithm has many useful applications in augmented reality, perspective correction, or relative motion tracking. The results from this algorithm, although promising, has some limitations such as:

- Only limited to tennis courts.
- ROI finding relies on being the object with the most features.
- Part of the court must at least be in frame.
- Taking the top and bottom 20% of the lines can fail if many lines have been identified outside of the true court borders.
- Scoring has no minimum threshold, which could lead to false positives
- The true borders of the court must exist in the candid list.

To overcome some of these limitations, some suggested future work can include:

- Implement different court reference images in the scoring phase. Multiple templates can be tried out to find the best performer. However, this may increase the calculation time if no filtering is applied.
- Implement a threshold on how many features should match before it is accepted as a potential solution. The threshold could be a percentage.
- During the scoring phase, the court reference image can be divided into sections and tested separately. The overall score could be the summation of all the parts. This could reduce the effect of not having the true border in the candid list. As this would be computationally expensive, we could trigger this only if the minimum threshold has not been met.
- Given that the output from the Deep Hough Transform Line Priors model have been quite accurate, using its Hough lines output instead of the original threshold-ed image for homography matrix scoring can yield much better results. From experiments, the threshold image often had high noise and could easily ruin the scoring.

References

[1] Y. Lin, S. L. Pintea, and J. C. van Gemert, ≪Deep hough-transform line priors≫, (2020).

[2] D. Farin, S. Krabbe, P. With, and W. Effelsberg, ≪Robust camera calibration for sport videos using court models≫, in , Vol. 5307 (Jan. 2004), pp. 80–91, 10.1117/12.526813.

[3] T. Shimizu, R. Hachiuma, H. Saito, T. Yoshikawa, and C. Lee, ≪Prediction of future shot direction using pose and position of tennis player≫, in (Oct. 2019), pp. 59–66, ISBN: 978-1-4503-6911-4, 10.1145/3347318.3355523.

[4] Y.-Z. Zhang, Q. Dong, J.-Y. Wang, and Y.-W. Dai, ≪Court view shots detection based on hough transform and svm≫, in 2011 international workshop on multi-platform/multi-sensor remote sensing and mapping (2011), pp. 1–4, 10.1109/M2RSM.2011.5697366.

[5] D. Drazil, *What type of perspective should you use?*, June 2021.

[6] *Basic concepts of the homography.*

[7] R. Hinno, *Simple svd algorithms*, Jan. 2021.