Notes IPCV Virtual Advertising project

**Camera calibration**: the process of estimating and adjusting the parameters of the camera or the system so that the captured images and videos accurately reflect the real-world dimensions, geometry, or other properties . Camera calibration is used to determine the internal characteristics of the camera (also called **intrinsic parameters**) and sometimes its position relative to the scene (**extrinsic parameters**).

* To convert **2D** images (advertisment) into **3D** real-world coordinates (football field)

Why line detection?

**Projection of 3D Lines onto a 2D Image**: In computer vision, lines in the 3D world are projected onto the 2D image plane through the camera. This projection is governed by the camera’s **intrinsic** and **extrinsic** parameters. Detecting lines in the image allows you to reverse this process, extracting the **relationship between 3D world lines and their 2D projections**. This is crucial for camera calibration.

**Vanishing Points**: Parallel lines in the real world (e.g., sidelines, centerlines) converge at **vanishing points** in the 2D image. By detecting these vanishing points, you can infer the **orientation of the camera** relative to the ground.

Homography:

**Detection of Field Lines**: Suppose you are broadcasting a soccer match and want to overlay an advertisement next to the goal. You would:

* Detect the corners of the goal and the nearest field lines in the image.
* Establish corresponding points in the real-world coordinate system (e.g., the physical locations of the goal and lines on the field).
* Calculate the homography matrix using these corresponding points.

**Projecting the Advertisement**: When placing the ad, you can take the coordinates of the corners of the advertisement and transform them using the homography matrix HHH. This transformation ensures that when the advertisement is overlaid on the video feed, it accurately aligns with the real-world locations on the field.

**Application in Virtual Advertising**

In the context of virtual advertising, the relationship between homography and camera calibration can be outlined in the following steps:

1. **Calibrate the Camera**:
   * First, perform camera calibration to obtain the intrinsic and extrinsic parameters. This calibration can be done using a calibration pattern (like a chessboard) or by using known objects in the scene.
2. **Estimate Homography**:
   * Once the camera is calibrated, use the intrinsic parameters to help estimate the homography matrix that relates the 2D image coordinates of the advertisement to the 3D coordinates on the field.
3. **Project the Advertisement**:
   * With the homography matrix available, you can then accurately project the advertisement onto the field in the video feed. This ensures that the ad is correctly aligned with the physical dimensions and perspective of the field.
4. **Adapt to Camera Movement**:
   * If the camera moves, you may need to re-estimate the extrinsic parameters and adjust the homography accordingly. A robust calibration process allows you to continuously adapt the virtual advertisement placement to maintain its accuracy throughout the broadcast.

APPROACH

* Perform line detection
* Divide into horizontal and vertical lines and find intersections
* (Manually) select pairs of corresponding intersections of the model (soccer field) and detected field lines (step before)
* With that we can create the homography

**Line detection**

* Field mask: we know that the field is green so we can first do a field mask so that the stadium and boardings are removed, before doing the line detection.
  + Histogram of the image, we get the green value and select a range around this value. Pixels within this range keep them, the rest in black.
  + Maybe we detect pixels in the stands or somewhere we don’t want (also players): filter by selecting the largest component or maybe morphological operations (erosion,dilation)
  + Then:
    - Hough transform line detection algorithm to obtain the mathematical representation of the lines
    - Ransac

Hough transform for line detection

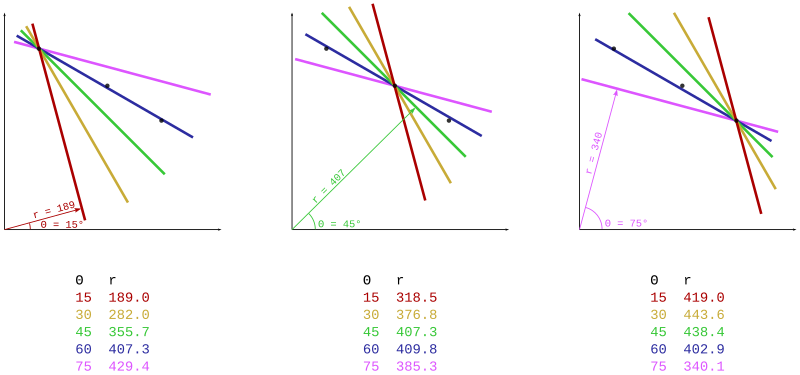
* **Edge Detection**: Detect edges in the image using techniques like Canny edge detection.
* **Transform Points**: For each edge point, compute the (ρ,θ) representation of the lines passing through that point.

ρ=xcos(θ)+ysin(θ)

where:

* ρ is the perpendicular distance from the origin to the line.
* θ is the angle of the line with respect to the x-axis.

For every edge point, we can compute multiple ρ for different θ values ranging from 0 to 180 degrees. This means each edge point in the image contributes to multiple potential lines in the Hough space.



* **Vote in Hough Space**: Populate the Hough accumulator by voting for every edge point and its corresponding (ρ,θ) values, increment the accumulator cell corresponding to those values.
* **Identify Peaks**: Analyze the accumulator for peaks that indicate likely line parameters.
* **Extract Lines**: Convert peak values back to line equations in the original image.

**Ransac**

Overview of RANSAC

RANSAC works by iteratively selecting random subsets of data points, fitting a model to these points, and then evaluating how well this model explains the remaining data points. Here’s how the RANSAC algorithm works for line detection in detail:

For each iteration of the RANSAC algorithm:

1. **Random Sampling**:
   * Randomly select a minimal subset of points required to fit the model. For line detection, you only need two points to define a line.
2. **Model Fitting**:
   * Fit a line model to the selected points.
3. **Inlier Detection**:
   * For each point in the dataset, calculate its distance to the fitted line model.
   * Count how many points fall within the defined distance threshold TTT. These points are considered inliers.
4. **Model Evaluation**:
   * If the number of inliers for the current model is greater than the previously found best model, update the best model with the current line parameters and the corresponding inliers.

**5. Termination**

* After completing the specified number of iterations, the algorithm will have identified the best-fitting line based on the maximum number of inliers.

**6. Output**

* The output of the RANSAC algorithm is the parameters of the best line model (slope and intercept) and the set of inliers that fit this line.

