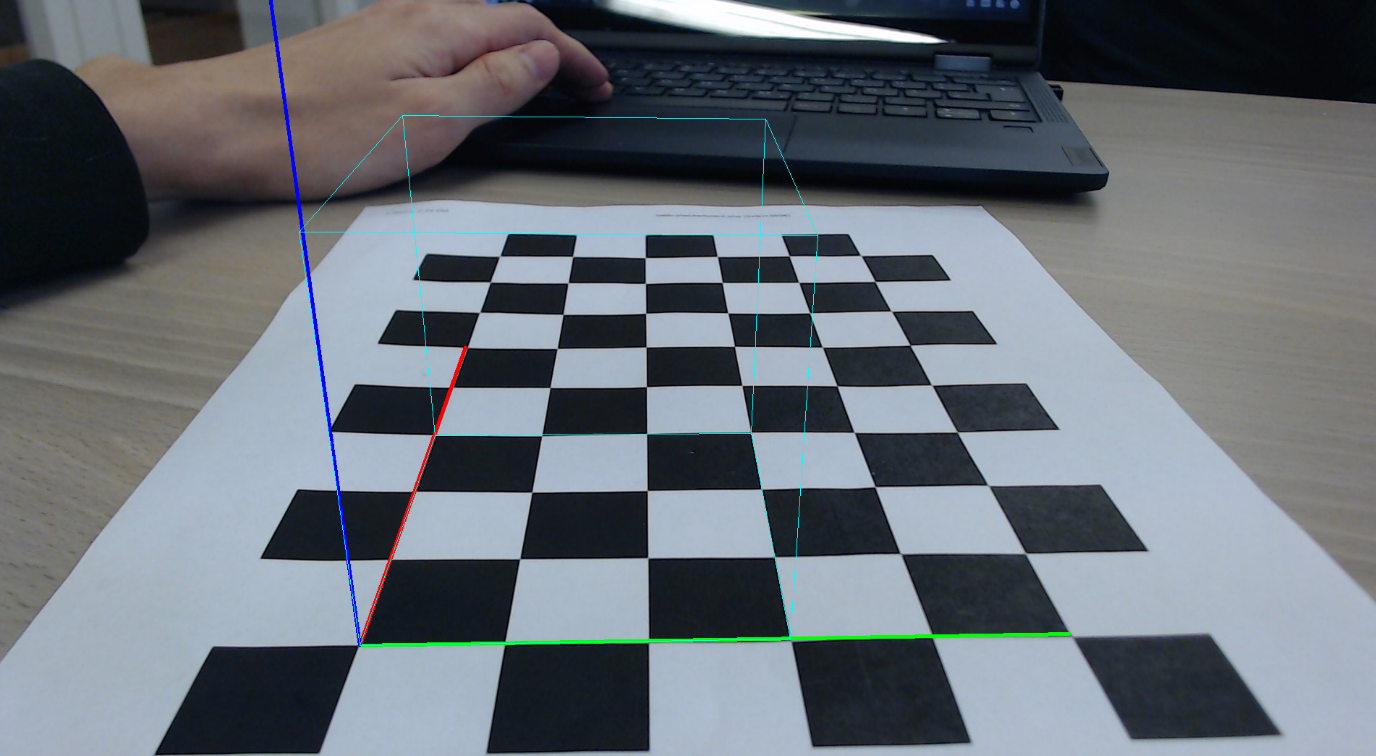
# Run 1

Camera Matrix



# Run 2

Camera Matrix

A picture containing indoor, floor, person, checker

Description automatically generated

# Run 3

Camera Matrix

A picture containing indoor, floor, person, checker

Description automatically generated

# Comparison

The biggest difference can be seen between run 1 and run 2 and 3. The focal length in both **Fx** and **Fy** have a difference of up to 50 units. The optical center **Cx** and **Cy** are also calculated slightly off. The overall quality drops significantly as the amount of input images is reduced. Due to the incorrect distortion calculations the Z coordinate gets more skewed the larger it becomes. In run 2 the left-side corners of the cube begin to shift apart from the center and a mismatch is seen between the XYZ axis and the side of the cube touching it. In Run 3 this effect is a bit more pronounced as the whole top part of the cube is slightly larger than the bottom part.

# Optional Implementations:

**10p** Video tracking -> Video Link:  
**10p** Corner point interpolation:  
To better acquire the exact dimension and transformation of the chessboard on the image we’ve used the function **getPerspectiveTransform()** to map a uniform set of points to the warped image. We achieve that in these steps:

* Select 4 corners in order on the image and get 2D coordinates from mouse selection point.
* Calculate the width and height of the image within the boundary of the 4 selected corners
* Using the rows and cols information and the width and height we can linearly expand an array of 2d coordinates to fit said dimensions uniformly.
* Using **getPerspectiveTransform()** using the 4 corners from the original image and 4 corners from the uniform array we get the transformation required to transform the uniform set to the warped set.
* We perform the transformation on each 2D point using **perspectiveTrasform()**

**10p** Image processing:?