Lab 06 - Focus: Reflection Essay

In this exercise we wanted to verify that the focal length of camera claimed by the manufacturer is the same as the focal length we obtain empirically through the camera calibration procedure.

The value claimed by the manufacturer was (Using EXIF meta-data for the images):

Focal length **f = 3.84 mm**

We took the below 7 images for the purpose of camera calibration. In sequence they are named

Img1.jpg, …, img7.jpg

In these images we found the below correspondences between the 6 feature points and their pixel coordinates

im1 = [[2168,3160],[1888,3152],[1640,3144],[2120,2856],[1848,2856],[1600,2864]]
im2 = [[2288,3064],[2160,3048],[2024,3048],[2320,2840],[2192,2824],[2056,2816]]
im3 = [[2384,3144],[2264,3200],[2136,3280],[2408,3032],[2280,3096],[2152,3168]]
im4 = [[2172,3344],[1976,3312],[1792,3276],[2152,3232],[1944,3196],[1756,3164]]
im5 = [[2268,3636],[2056,3644],[1844,3652],[2296,3544],[2072,3552],[1852,3564]]
im6 = [[2564,1904],[2224,1936],[1904,1972],[2520,1636],[2204,1672],[1900,1712]]
im7 = [[2432,3492],[2256,3496],[2084,3500],[2432,3328],[2260,3332],[2084,3336]]

The sequence used for the 3D real world coordinates was the same as lab05, in order bottom row rightmost, going left and then top row rightmost going left.

ref = [[0, 0, 0], [3, 0, 0], [6, 0, 0], [0, 3, 0], [3, 3, 0], [6, 3, 0]] in cm units

Next we obtained the specifications of the camera from the wikipedia page
https://en.wikipedia.org/wiki/Image_sensor_format#Table_of_sensor_formats_and_sizes
1/2.9" Sony EXMOR IMX322[41]: Sensor Width (mm): 4.98 Sensor Width (mm): 3.74 Aspect Ratio: 4:3.
The resolution obtained from the meta-data was 3000px - 4000px, using this and the above sensor dimensions, we got the dimension of 1 pixel to be 1.25um

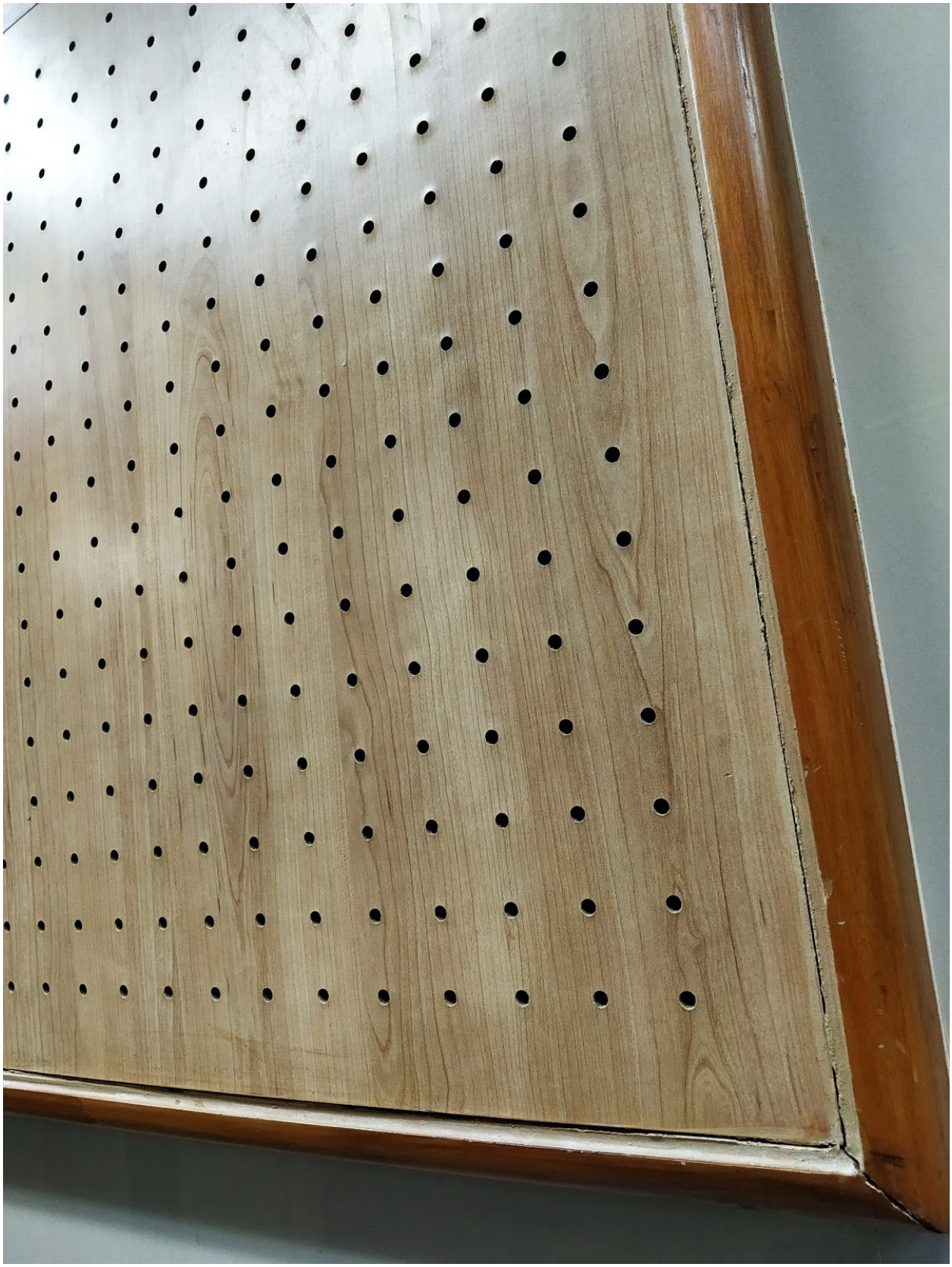
After running the calibration with 3 of the 7 images, img 1, 5 and 6
We got the intrinsic matrix as:
calibration matrix:
[[3.07995068e+03 0.00000000e+00 1.60652509e+03]
[0.00000000e+00 3.02527946e+03 2.13536963e+03]
[0.00000000e+00 0.00000000e+00 1.00000000e+00]]

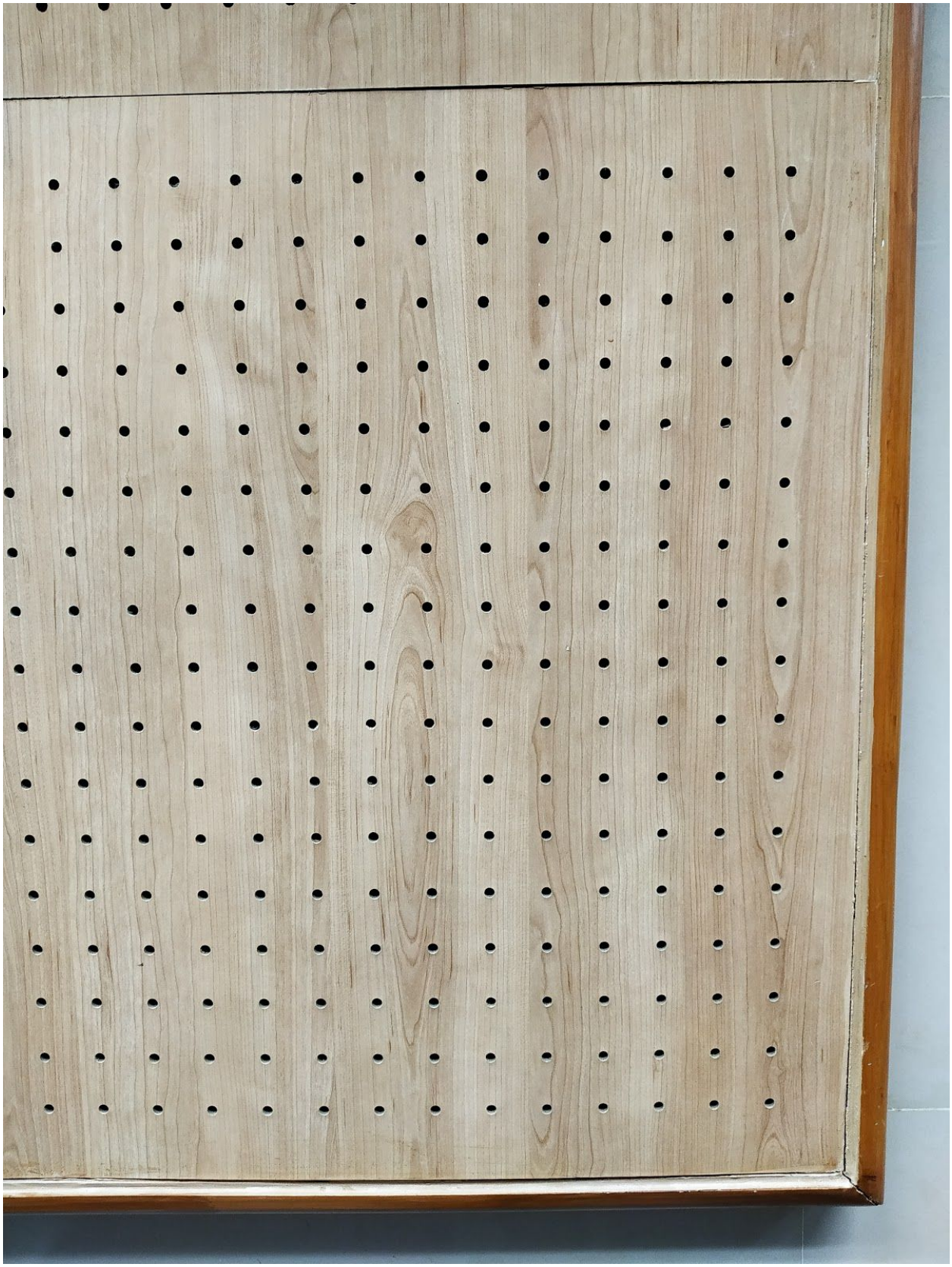
After converting the focal lengths from pixels to mm we got the following values.

fx 3.85mm
fy 3.78mm

Average focal length 3.82mm

Which is quite close to expected value of 3.84mm





Lab 06 - Focus (Outlab): Reflection Essay

In this outlab we were asked to obtain views of a synthetically overlayed book from different angles on a given plane. We were given the intrinsic matrix of the camera with which the images of the plane were taken along with the four images of the plane in question. These input images were taken from different viewpoints.

To obtain each projection, we did the following -

1. We first calculated the rvecs and tvecs associated with the image using the solvePnP method and 6 reference points along with their corresponding points in the image. These were used to obtain the camera matrix for the image.
2. Using the length, width and breadth we get the 3D world coordinates of the virtual book to be augmented on the plane
3. Then the camera matrix is used to project them on the 2D image hence giving us one view of the book.
4. The drawlines() function then draws red lines for the skeleton of the book
5. The textures are then added by first computing the Homography matrix and then performing a Perspective Transform using the openCV API warpPerspective() as in lab03. It is crucial that we draw textures only for “Visible-Surfaces”
6. Finally the textures are merged with the input images by using simple masks to get the final results

The following points were chosen for correspondence in the given images to find the **rvecs and tvecs**:
Img 1 : [[1359, 2090], [1467, 2083], [1569, 2078], [1361, 1950], [1470, 1948], [1573, 1945]]
Img 2 : [[1056, 2784], [1216, 2788], [1364, 2780], [1052, 2632], [1208, 2636], [1364, 2636]]
Img 3 : [[962, 2035], [1215, 2024], [1474, 2013], [990, 1870], [1237, 1859], [1468, 1848]]
Img 4 : [[1321, 2745], [1404, 2785], [1493, 2829], [1320, 2638], [1398, 2676], [1486, 2715]]

The corner points of the Texture images are also kept ready for the Homographies
# texture_front = np.float32([[0, 0], [99, 0], [99, 99], [0, 99]]) - Blue Default Cover
texture_front = np.float32([[0, 0], [401, 0], [401, 599], [0, 599]]) (HP-1 Cover)
# texture_side = np.float32([[19, 0], [19, 99], [0, 99], [0, 0]]) (Default Green Side)
texture_side = np.float32([[307, 0], [307, 3338], [0, 3338], [0, 0]]) (HP-1 Side)
texture_pages = np.float32([[0, 2463], [0, 0], [567, 0], [567, 2463]]) (Papers in a book)

The Reference Sequence: [[0, 0, 0], [3, 0, 0], [6, 0, 0], [0, 3, 0], [3, 3, 0], [6, 3, 0]]
Is used for calibration. The above lengths being in cm (3cm gaps between dots assumed)

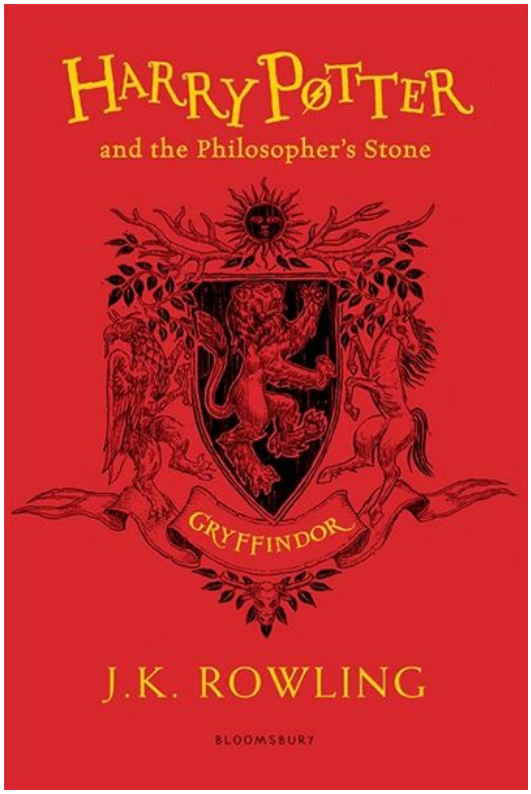
The command used to obtain the following results was-
python3 code/augment_book.py data/augment_book -w 3 -l 15 -b 10

The output images in each case are attached below:

In all of the cases, the books are oriented as per the instructions in the PDF such that we can rotate mid-air and look at the cover correctly (While playing Quidditch in Hogwarts perhaps!). NOTE: Results are not displayed by the script but saved to disk directly since their large size makes them difficult to handle with python viewer.

As mentioned, textures are added only for visible surfaces. In all the views of the augmented book. We can surely say that the front cover will be visible and the back cover will not be visible, however anywhere between 0 - 2 additional faces may be visible depending on the view. Automating the process of painting only the visible faces is hard since we do not know the number of visible faces.
In case the number of visible faces is known, then depending on the lengths between the projected vertices, we can identify which edges are closer to us and thus unoccluded.
This can be done since the edges closer to us will appear longer than the edges far away from us when their true world lengths are the same (length and breadth).
Then we can color only the faces associated with these longer edges to get the desired result.
However if both faces of a certain type are occluded (as was the case in this exercise for views 2 and 3) then we cannot use this method and have to resort to either manually doing it or somehow using the camera calibration params rvecs and tvecs to get the result.

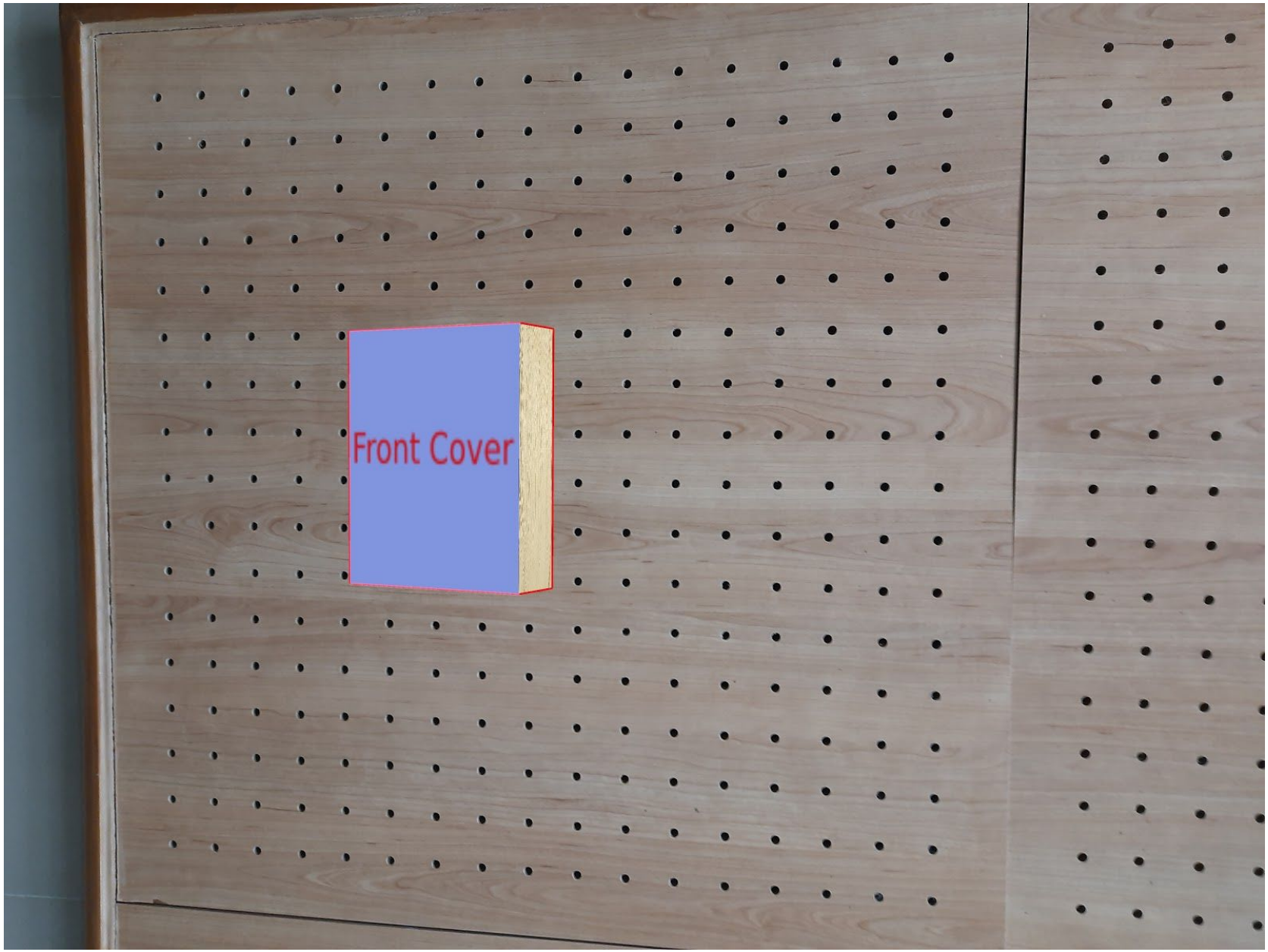
We have added our own textures to make the book look appealing and realistic, these are shown below!



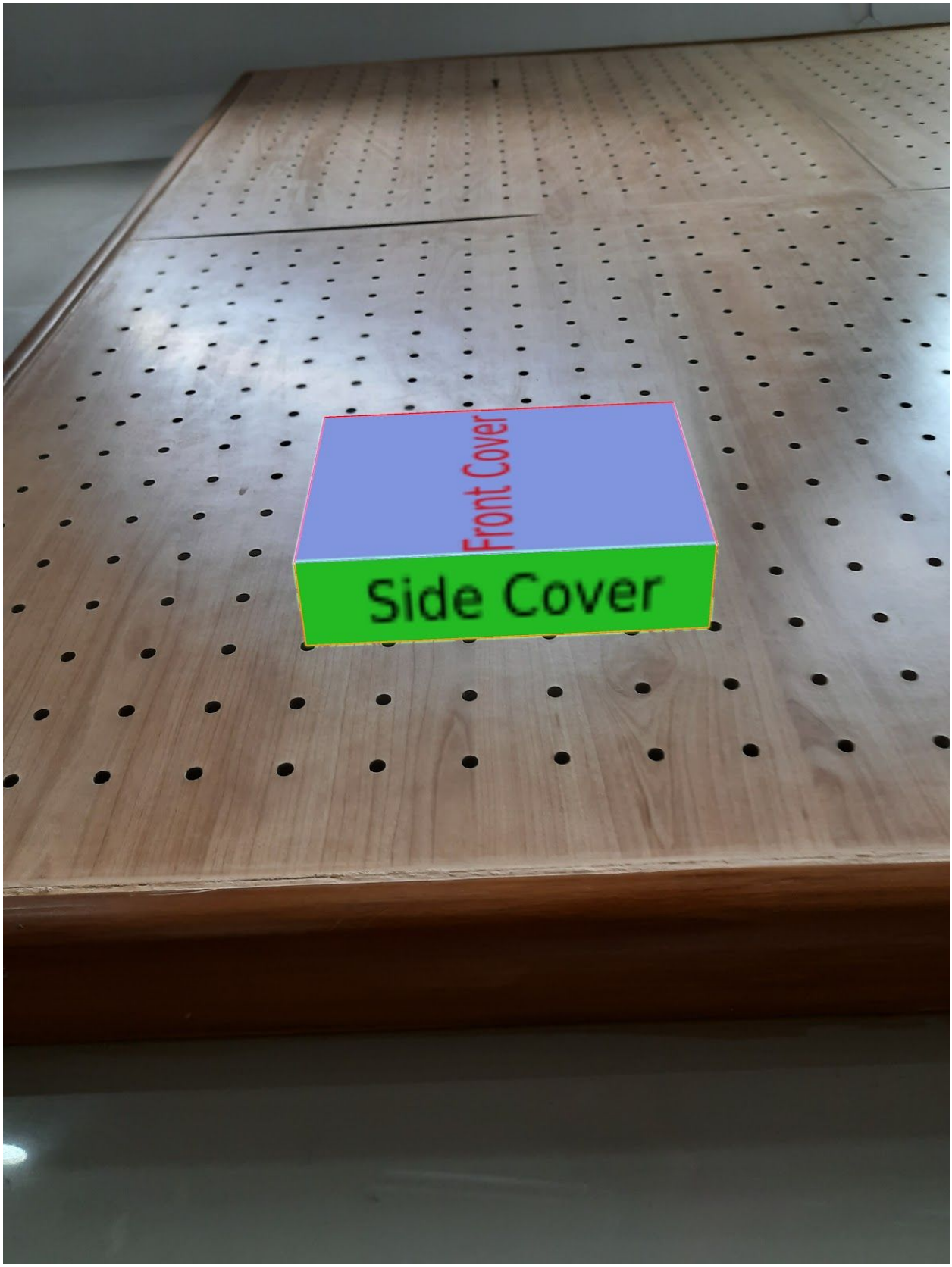
Harry Potter
and the
philosopher's stone
J.K. ROWLING


Front Cover

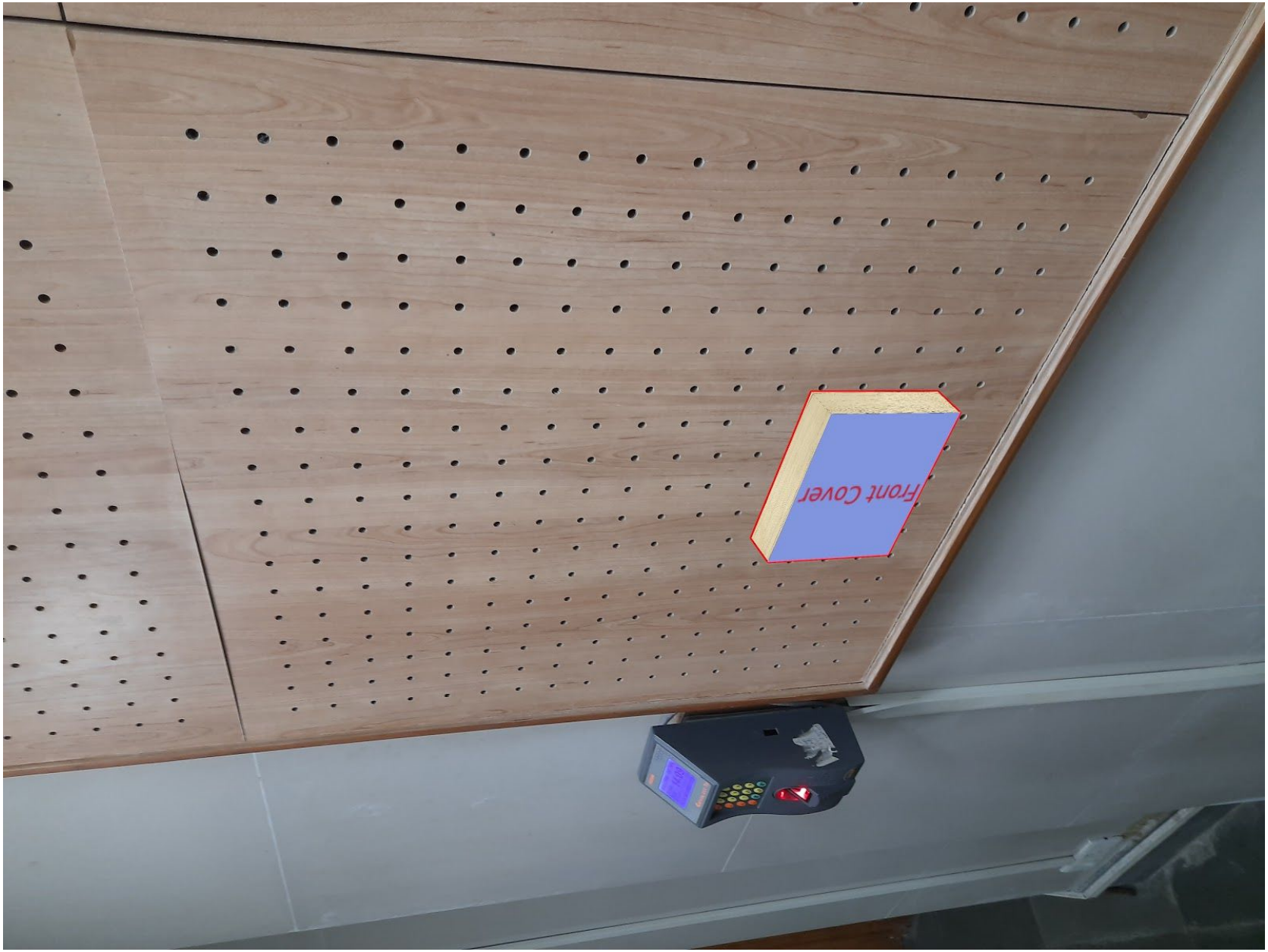

HARRY POTTER
and the Philosopher's Stone
GRYFFINDOR
J.K. ROWLING

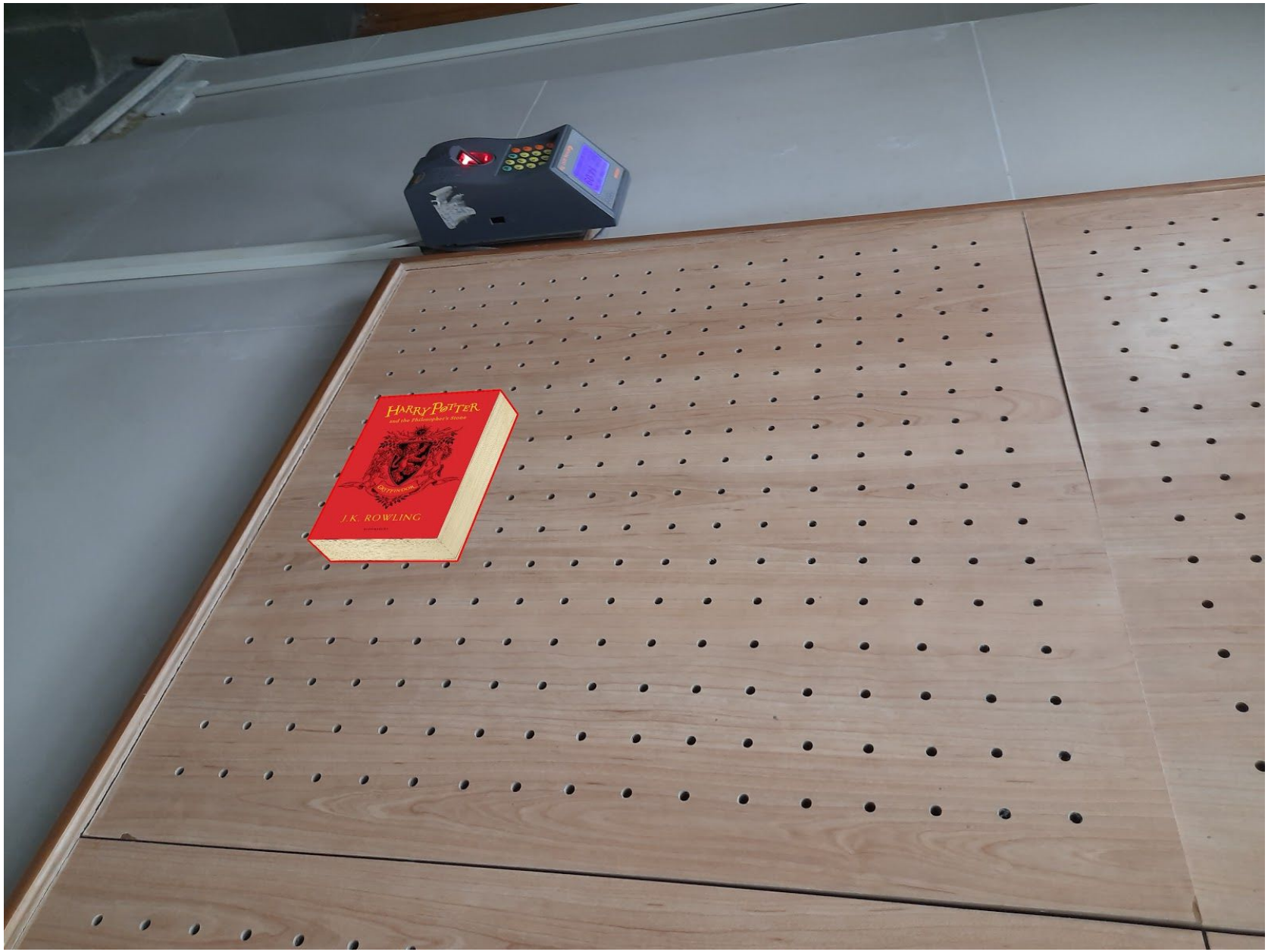

Front Cover


HARRY POTTER
and the Philosopher's Stone
GRYFFINDOR
J.K. ROWLING


Front Cover
Side Cover


HARRY POTTER
and the Philosopher's Stone
J.K. ROWLING
HARRY POTTER
and the
Philosopher's Stone
J.K. ROWLING


Front Cover


HARRY POTTER
and the Philosopher's Stone
J.K. ROWLING