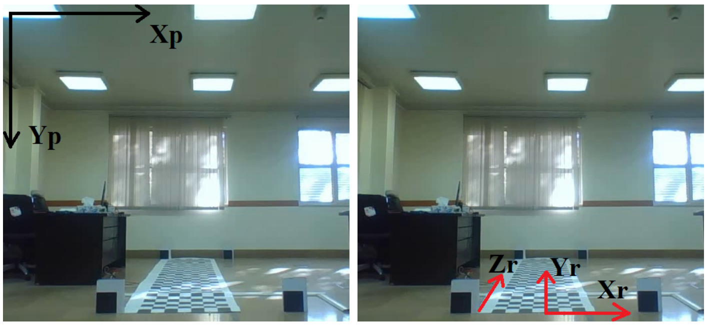

Fig. 1: The axes assumed in pixel (left) and real-world coordinates (right).

1 Proposed Camera Calibration

The axes of pixel coordinates in the images and real-world coordinates are shown in Figure 1. A proposed neural network has been trained using video data captured by a single camera in space served as for conversion from two-dimensional pixel coordinates to real-world coordinates. An illustrative example will be provided in the following to demonstrate the process of generating a dataset for training the neural network-based calibration model.

The calibration methodology involves considering the visual scene on the floor as a rectangular region, with the front side of the rectangle aligned parallel to the camera lens. Additionally, we make the assumption that the dimensions of the visual scene are already known. The visual scene is covered with checkerboard-shaped plots where the squares on them have certain dimensions, for example 10cm × 10cm (Figure 1). The camera serves as the reference point for real-world coordinates, situated 320 cm away from the initial line of sight in the visual scene (front line), with a height of 72 cm from the ground, and 52 cm from the left standing square. The leftmost bottom corner of the standing square that located at the forefront on the left-hand side is of real-world coordinates $x_r = -52$, $y_r = -72$, and $z_r = 320$. It is important to highlight that the checkerboard planes are utilized solely for ease of reference, and any preferred technique can be employed to partition the room's floor and calculate distances.

1.1 Dataset for calibration

Per each cross section, we place the 15cm × 15cm standing square on one of the squares on the checkerboard planes at that cross section and keep it there for 3 seconds, then move to the next square in longitudinal line until we reach out the end of the scene (Figure 2). Since the length, width, and height of the visual scene are known, the three-dimensional position of all vertices of the standing square relative to the reference point along the sweep of the scene will be known. To get the two-dimensional position (pixel coordinates in the images) of each point, it is enough to identify the four corners of the 15cm × 15cm square in the image. Due to the fixed number of pixels at a certain distance from the camera, by calculating the distance between two points, we can obtain the amount of changes in height and width and obtain the pixel coordinates for all points in that distance. Considering that the width of the room is 3 meters and the width of the square is 15 centimeters, as a result, we have 20 specific points in the horizontal direction. If we calculate the points up to a height of 225 cm (equivalent to 15 points) for each specific distance, then we will have 300 points for each cross section. On the other hand, considering that the length of the visual scene is 4 meters, we can place the square in 40 different points, so we will get a total of 12,000 points, which will have both their two-dimensional coordinates and their three-dimensional equivalent.

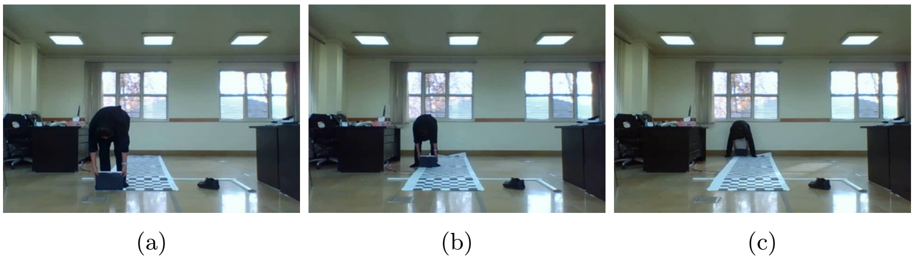
(a) (b) (c)

Fig. 2: Sweeping the floor of the visual scene via a standing square. Three frames have been shown from the recorded video.

1.2 Calibration model structure

The obtained dataset is employed for the training of the neural network shown in Figure 3. The network takes as input the two-dimensional coordinates of the vertices of the upright squares in the images, along with their corresponding distance to the camera (z_r). The output of the network consists of the two-dimensional coordinates of the vertices in the real-world space (x_r, y_r).

The model consists of three hidden layers with 300, 32, and 16 neurons respectively, while the output layer comprises two neurons. The Multi-Layer Perceptron (MLP) network was chosen for this task because of its ability to learn complex, nonlinear relationships in the data. As shown in experimental results, this model successfully mapped inputs in pixel coordinates to outputs in real-world coordinates with high accuracy, avoiding overfitting.

1.3 Assessment of Calibration Model

The NN structure proposed in Figure 3 has been trained using data gathered and explained in section 1.1. Mean squared error (MSE) as the objective function is minimized via Adam optimizer in order to determine the mapping between pixel coordinates and real-world coordinates. The training process employs 70% of the data, while the remaining portion is reserved for evaluation. Training is conducted over 700 epochs with a batch size of 32. The learning curve is depicted in Figure 4 to demonstrate the performance of the calibration across the training phase in terms of MSE.

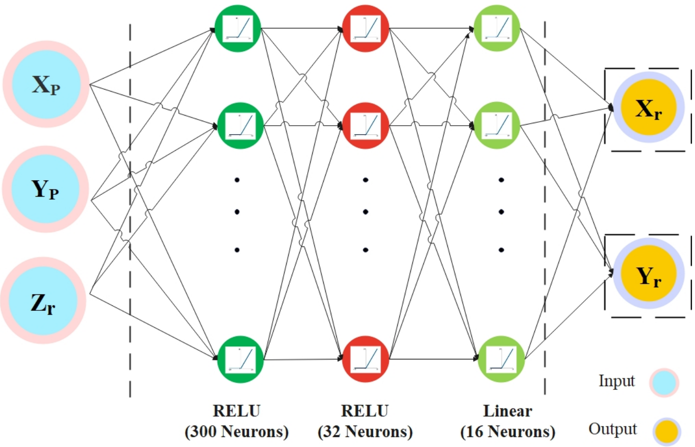

Fig. 3: A four-layer NN for mapping the pixel coordinate to the real-world coordinate. z_r is also fed as a side information along with (x_p, y_p) to the NN. Estimation of z_r is explained in section 2.2.4 of the paper.

The final MSE reaches out 2.73 for the training set, while the MSE for the evaluation set is 1.50, verifying the potential of the designed network to learn the underlying mapping between pixel and real-world coordinates.

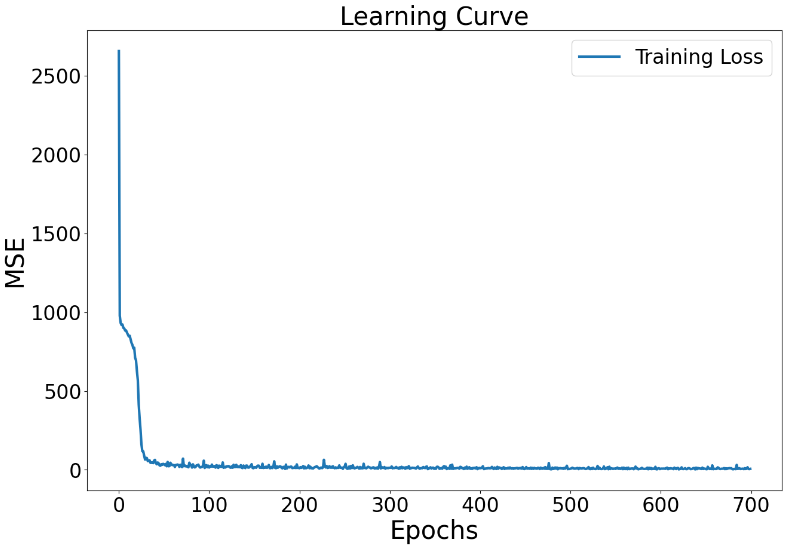

Fig. 4: The learning curve of calibration neural network as shown in Figure 3.

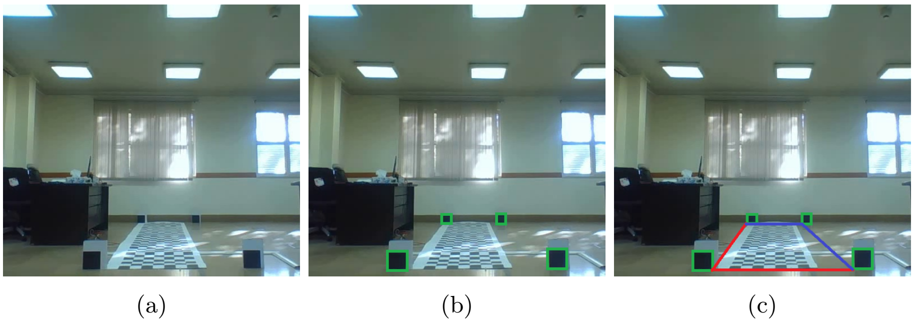

Fig. 5: Identification of the visual scene, (a) Squares in the four corners of the room, (b) Identification of black standing squares, (c) Border lines of the visual scene's floor are specified through connecting proper vertices of the four black standing squares.

2 Visual scene determination

In order to identify and track speakers located within a specific area of a room, it is crucial to delineate particular areas of the floor space that are relevant to the intended visual scene, enabling the recognition of speakers and the derivation of their two-dimensional coordinates. The designated area is regarded as rectangular, and to establish its boundaries, we require four black squares of precise and equal measurements. To accomplish this, we utilized squares measuring 15cm × 15cm and positioned these black upright squares in the four corners of the space as depicted in Figure 5(a). The next step involves identifying all four standing squares that define the boundaries of the visual scene. In order to detect the black squares, an averaging filter is initially applied to the entire image to blur the color space. Subsequently, the smoothed image is converted into a gray-scale image using the cv2.COLOR_BGR2 command. Finally, the squares are identified by utilizing the boundingRect function (Figure 5(b)).
For horizontal lines, simply connect the lower right corner and the lower left corner of the front squares. Similarly connect the same corners for the squares at the end of the scene. In order that all four squares are thoroughly visible in the image, the squares positioned at the end of the scene are placed inward, while those at the front are placed outward. For two depth lines, on the left side of the scene, the bottom right corner of the front square is connected to the bottom left corner of the square at the end, and on the right side, the bottom left corner of the front square is connected to the bottom right corner of the square at the end. As a result, the four border lines of the room's floor in the visual scene are defined as shown in Figure 5(c).