Real-Time Localization Framework for Autonomous Basketball Robots

Naren Medarametla
School of Computer Science Engineering
Vellore Institute of Technology
Chennai, India
naren.medarametla2023@vitstudent.ac.in

Sreejon Mondal
School of Electrical and Electronics Engineering
Vellore Institute of Technology
Chennai, India
sreejon.mondal2023@vitstudent.ac.in

***Abstract*— Localization is a fundamental capability for autonomous robots, enabling them to operate effectively in dynamic environments. In Robocon 2025, accurate and reliable localization is crucial for improving shooting precision, avoiding collisions with other robots, and navigating the competition field efficiently. In this paper, we propose a hybrid localization algorithm that integrates classical techniques with learning-based methods, relying solely on visual data from the court's floor to achieve self-localization on the basketball field.**

***Keywords*—Robot Localization, Autonomous Navigation, Neural Networks, Robocon**

I. Introduction

Section II reviews existing methods and prior research related to this work. **Section III** provides a detailed description of our proposed algorithm, approach, and model architecture. **Section IV** provides the results obtained from our experiments. **Section V** evaluates the accuracy of our approach and discusses potential directions for future work.

II. Related Work

III. Methodology

Our approach is a two-step process that begins with **Preprocessing** the image, followed by passing it to the model for **Inference**.

A. Preprocessing

The input image having dimensions (640 × 480 × 3) is converted from the RGB color space to the HSV color space, then the white regions are masked out using two predefined HSV ranges. The image is downsampled through a radial scan, flattened, and finally passed through the neural network. Figure I shows the preprocessing pipline and Algorithm 1 gives the implementation of the downsampling algorithm.

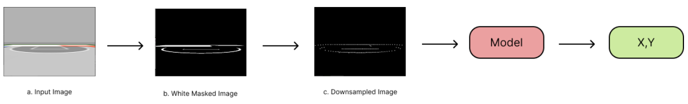

Figure I: preprocessing pipeline

Algorithm 1 Downsampling

```
Input: Image
$H \leftarrow$ Image.height
$W \leftarrow$ Image.width
$R \leftarrow$ Black Image
for angle $\leftarrow$ 0 to 180 step 2 do
    $lastPixel \leftarrow 0$
    $cx \leftarrow$ W / 2
    $cy \leftarrow$ H
    for d $\leftarrow$ 0 to max(H, W) do
        $x \leftarrow cx + d \times \cos(angle)$
        $y \leftarrow cy - d \times \sin(angle)$
        if $0 \leq x <$ W and $0 \leq y <$ H then
            $pixel \leftarrow Image[y][x]$
            if lastPixel = 255 and pixel $\neq$ 255 then
                $R[y][x] \leftarrow 255$
            end if
            $lastPixel \leftarrow pixel$
        end if
    end for
end for
Return R
```

B. Model Architecture

The proposed model is a feedforward neural network consisting of a flattening layer followed by four fully connected layers. The first 200 pixels from the top of the image are removed to reduce the input size, as they do not carry useful information and the pixel values are scaled down to the range [0, 1].

The resulting image with dimensions (640 × 280 × 1) is flattened into a vector of size 1,79,200 and passed through the first linear layer, followed by a ReLU activation function. The subsequent layers have sizes 1024, 256, and 64, each followed by a

ReLU activation. The final layer outputs a 2-dimensional vector representing the predicted x and y positions of the robot.

The rationale for using this relatively simple model lies in the simplicity of the input images and to reducing inference time.

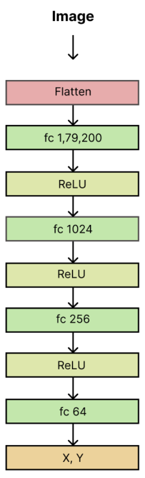

Figure II: architecture diagram

C. *Dataset*

A digital twin of the robot was created and simulated in a replica of the Robocon 2025 arena using the Gazebo [1] simulator with the help of ROS 2 [2].

The robot was driven through the arena capturing images and corresponding x, y coordinates. A TimeSynchronizer was used to synchronize the frame header and the position header. A total of 6283 images were captured and split into training and test datasets with a 0.9 to 0.1 ratio. Care was taken to include every part of the arena for an unbiased dataset.

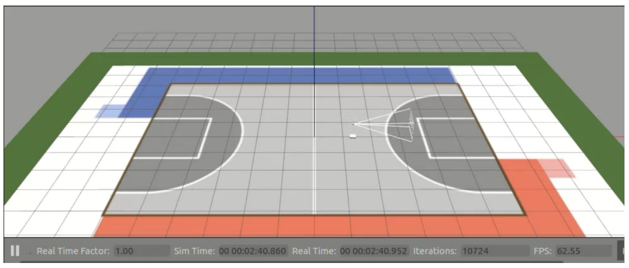

Figure III: gazebo simulation

D. *Training*

The model was trained for 15 epochs using an Adam optimizer [3] with an initial learning rate of 10^{-4} and a Mean Squared Error (MSE) loss function in batches of size 8.

IV. Results

Figure IV depicts the loss at each epoch throughout the training process. 8 independent images at different points of the court were again captured from the simulation and fed into the model, Figure V shows the plot between the ground truth and the prediction made.

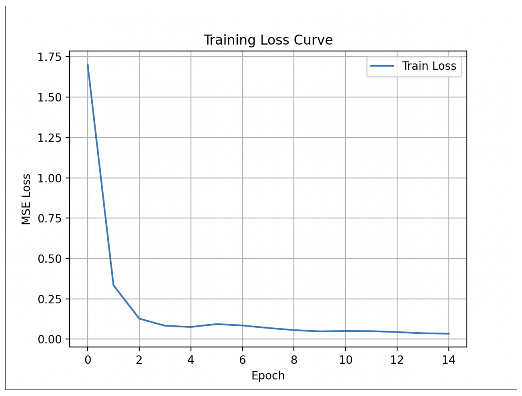

Figure IV: loss curve

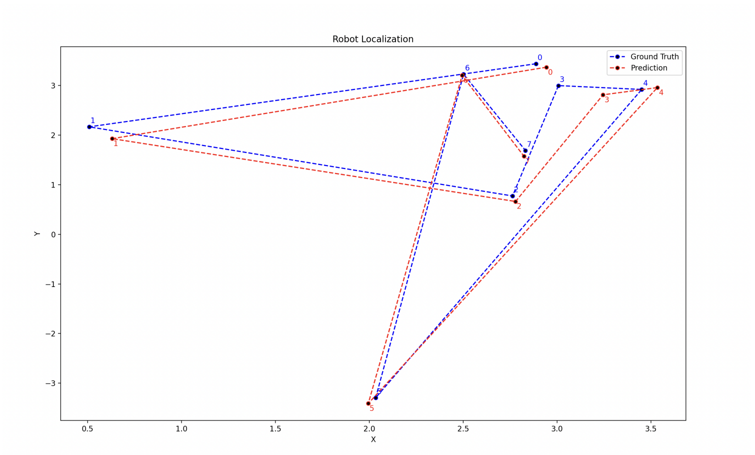

Figure V: truth vs prediction

V. Conclusion

References

[1] N. Koenig and A. Howard, ”Design and use paradigms for Gazebo, an open-source multi-robot simulator,” 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566), Sendai, Japan, 2004, pp. 2149-2154 vol.3, doi: 10.1109/IROS.2004.1389727.

[2] Steve Macenski, Tully Foote, Brian Gerkey, Chris Lalancette, and William Woodall. *Robot Operating System 2: Design, architecture, and uses in the wild.* Science Robotics, vol. 7, no. 66, 2022.

[3] Diederik P. Kingma and Jimmy Ba. *Adam: A method for stochastic optimization.* arXiv preprint arXiv:1412.6980, 2014.