Gesture-Based Control of a Quadruped Robot using Neural Networks

Group 1, Construction Robotics, WS 2023/24

Harsh Prabhakar Sharma (561451),
Ayush Vasantbhai Patel (564382),
Ishan Sandeep Mohite (561464)

***Abstract*—The issues at construction sites involve potentially hazardous or inaccessible locations, slow tool distribution, and the necessity for effective surveillance. These problems can compromise worker safety and result in low efficiency at work. This paper presents a novel approach to using the Petoi Bittle Robot dog for assistance and security at construction sites that helps to improve site safety and overall efficiency. Controlling the actions of the robot is achieved by utilizing hand gestures detected by the camera. The image feed received from the robot camera is passed through a neural network. The hand gestures are detected using a pre-trained neural network and open-source MediaPipe framework for hand landmark detection. The actions corresponding to the different gestures are defined beforehand. The gesture with the highest confidence based on the trained labeled data is selected. The actions are executed using the action commands from the Robot Operating System (ROS) framework.**

***Index Terms*—Hand gestures, Computer Vision, assistance and security, robot actions**

I. Introduction

The field of robotics has found its way into many different sectors in recent years. Different robots are used for a variety of automation and assistance tasks. Particularly, mobile robotics is an emerging field due to their mobility, flexibility, and wide range of tasks they can perform. Quadruped robots that mimic dog-like movement provide a wide range of motion that can be used to perform various movements on different terrains.

In modern times, neural networks are fundamental to artificial intelligence. Computer vision tasks are particularly well-suited for Convolutional Neural Networks (CNNs). [1] and [8] explain the architecture of CNN and its usage in image detection and recognition. The key objective of this paper is to use neural networks to recognize hand gestures from the real-time camera feed of a robot. Through the study of neural network architectures and training techniques, the objective is to improve gesture-based control and human-robot interaction. When the neural network techniques are combined with a quadruped robot, it helps to solve important issues in the construction industry.

Gesture-detection-capable robots help to overcome challenges such as safety risks, inefficiencies, limited accessibility, and human errors which occur if tasks are carried out manually by the workers. The robot helps in surveillance duties, navigating through difficult terrain, and delivering tools to precise locations, which improves overall site safety and operational efficiency. [6] presents a way to execute actions using the quadruped robot based on the hand gestures detected by the camera.

The method defined in the current paper provides a way to detect gestures using neural networks. A neural network architecture consisting of a hand detection net and a CNN first detects the hand landmarks using the MediaPipe framework. Then a feed-forward neural network is implemented which classifies the detected gesture from the set of gestures based on the labeled training data.

However, when performing tasks involving high computations, there always remains a question of computational efficiency and limitations. As the memory available on the robots to perform real-time recognition and command execution is limited, the neural network should be efficient whilst performing the actions with acceptable delay.

This paper dives deeper into the background and previous work, methodology used in the framework, results, and encountered challenges in further sections.

II. Background

A. MediaPipe Hand Detection

The MediaPipe Hand Landmark detection is an open-source framework by Google to detect the landmarks of the hands from an image. [7] & [2] show the application of the MediaPipe library to create hand gesture recognition systems. MediaPipe can be used to identify 21 hand landmark points and apply visual effects to them.

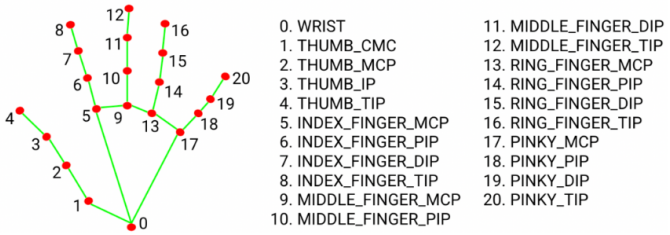


Fig. 1: Hand landmarks (MediaPipe)

MediaPipe library uses a machine learning (ML) model to operate on image data, which can either be a static image

or a continuous stream as in this project. The output is the handedness of several detected hands as well as hand landmarks in both image and world coordinates.

The precise localization of 21 key points over the hand is achieved inside the detected hand region. Figure 1 shows the 21 key points defined as hand landmarks.

Each key point has coordinates composed of X, Y, and Z. The Z coordinate gives the depth of the landmark, and the X and Y coordinates are first converted to relative coordinates concerning key point zero. These coordinates are then flattened to a one-dimensional array and normalized to a maximum value (absolute value). The normalized coordinates obtained in the form of an array are used for further applications involving the analysis or interpretation of hand landmarks.

III. Framework for Gesture-Based Control of Quadruped Robots

This section is divided into five parts. First, hand gesture detection, mIDOG setup, and robot control are explained. After, the data acquisition and interfacing of modules are presented.

A. Hand Gesture Recognition using Neural Networks

Neural networks are widely used for tasks related to image recognition. In particular, a Convolutional Neural Network (CNN) is a deep learning algorithm used for image processing and analysis. [4] & [5] show the implementation of CNNs for creating a hand gesture recognition system. Compared to other deep learning algorithms, CNN requires very little pre-processing data. CNN works efficiently as its layers have fewer parameters compared to the fully connected layers of a traditional neural network.

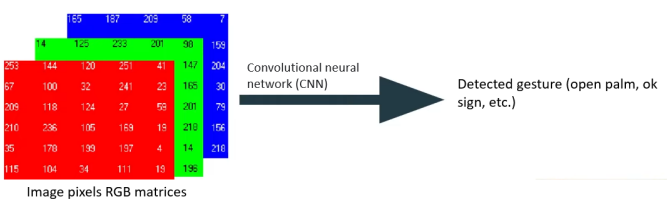


Fig. 2: Classifying hand gestures directly using CNN

Training a network to predict hand gestures from the image using CNN can be achieved using the following approaches:

- Extracting the RGB color matrices from the image and applying a CNN to classify the hand gestures by extracting features directly from the RGB matrices (as shown in Figure 2)
- Extracting the RGB color matrices from the image and applying a CNN to get hand landmarks. Then, training and applying a feed-forward neural net to detect hand gestures from these landmarks

Training the network to identify images using the latter approach is significantly easier and more efficient than training the network just on pixels. Different images have different camera resolutions, variable hand shapes and sizes, and contrasting backgrounds and lighting. Due to these factors, training the network just on pixels requires much more training data.

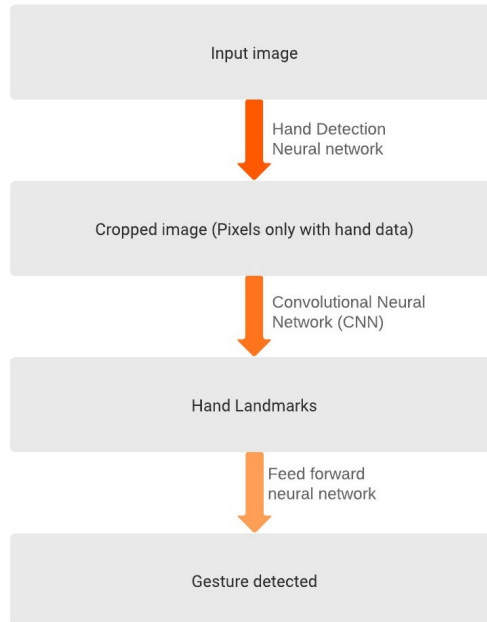


Fig. 3: System workflow

Instead of passing the whole image to the CNN, the image is cropped using a hand-detection neural network to get the pixels that contain hand data. This cropped image representing pixel RGB values is then passed through a CNN to get the hand landmarks. Implementation of both the mentioned networks is done using the MediaPipe library. Then a small feed-forward net is trained to identify the gestures from the landmarks. Figure 3 shows the workflow of the neural networks and the data passed through the architecture.

The neural network from [3] is used as a base model for this project. The network is then further modified as per the required functionality and more gestures and training data are added. The training data required to train the network by this approach is way less and the model is more efficient.

For this project, only single hand gestures are used to control the robot. Hence, the maximum number of detected hands at a time is set to one.

B. mIDOG setup

The hardware used in the mIDOG setup is a Petoi Bittle Robot dog equipped with a Raspberry Pi 3A+ that acts as a micro-controller, nine servo motors to perform various motions, and sensors like a Raspberry Pi camera and Inertial Measurement Unit (IMU). The robot consists of a Li-ion battery which has a battery life of around 70 minutes. Figure 4 shows the robot with the sensors in a standing pose.

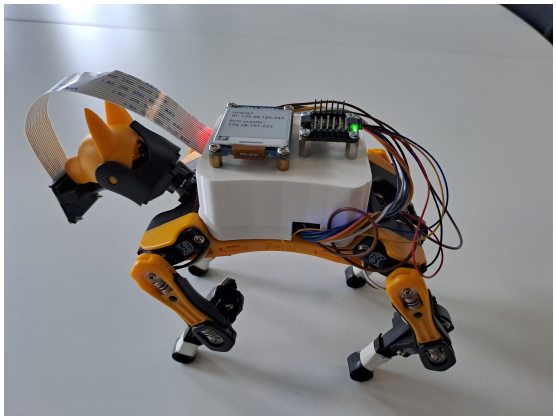


Fig. 4: mIDOG Robot

The software environments Gazebo and RViz in conjunction with the ROS framework are used to simulate the behavior of the robot before testing the robot in the real world. Due to the limited computation capability and memory of Raspberry Pi, the robot is connected to a remote server using ROS where the computations are carried out. This allows for more computational memory and complex tasks like image segmentation and neural network processing.

C. Robot Control

The mIDOG is a quadruped robot with eight servos in its legs which allows a variety of actions and movements. The robot controller has some predefined functions that can be directly used to execute commands and gaits using the servos and the sensors. Executing the control functions or commands helps to achieve basic and viable actions based on hand gestures.

Hand gestures	Executed actions
open palm	stop
close palm	walk forward
Ok sign	greeting
rock and roll action	stretch
index finger	crawl left
pinky finger	crawl right
two fingers	sit
3 fingers	walk backward

TABLE I: Hand gestures and corresponding actions of the robot

The actions executed by the robot for the corresponding hand gestures are as mentioned in Table I. The number of gestures that the robot can react to can be increased and corresponding actions can be assigned. However, as the number of labels or gestures increases, the network requires more data to correctly classify the labels. Otherwise, the accuracy of the network decreases and it predicts incorrect gestures.

D. Data Acquisition

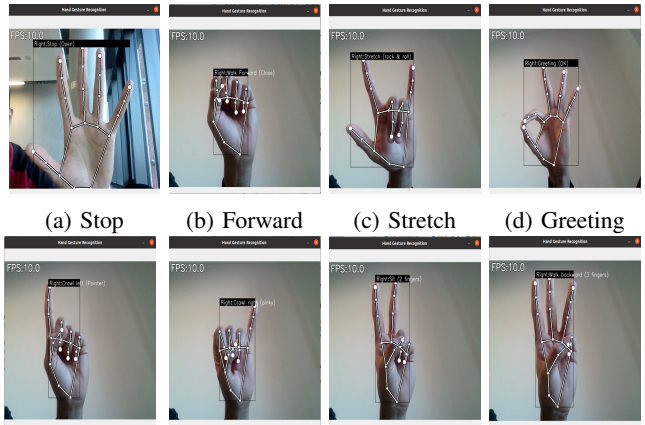
The normalized coordinates of the hand landmarks are stored as a CSV file in a training dataset enabling simple manipulation and access. This file serves as the important input for the machine learning model, improving its ability to learn and anticipate scenarios based on the variety of hand gestures that were recorded throughout the training phase. A portion of the CSV file that is created for new hand gestures can be seen in Figure 5.

```
6,0.0,0.0,-0.15853658536585366,-0.10975609756097561,-0.24390243902439024,-0.32926829268292684
6,0.0,0.0,-0.16265060240963855,-0.0963855421686747,-0.25301204819277107,-0.3253012048192771,-
6,0.0,0.0,-0.15217391304347827,-0.09782608695652174,-0.22282608695652173,-0.32065217391304346
6,0.0,0.0,-0.14871794871794872,-0.11282051282051282,-0.2153846153846154,-0.3128205128205128,-
6,0.0,0.0,-0.13636363636363635,-0.11616161616161616,-0.20202020202020202,-0.30808080808080807
6,0.0,0.0,-0.14646464646464646,-0.11616161616161616,-0.20707070707070707,-0.30303030303030304
6,0.0,0.0,-0.155,-0.105,-0.22,-0.3,-0.145,-0.445,-0.045,-0.54,-0.185,-0.55,-0.29,-0.75,-0.34,
6,0.0,0.0,-0.15639810426540285,-0.0947867298578199,-0.22748815165876776,-0.27488151658767773,
7,0.0,0.0,-0.09289617486338798,-0.09289617486338798,-0.1092896174863388,-0.2459016393442623,0
7,0.0,0.0,-0.08695652173913043,-0.09239130434782608,-0.10326086956521739,-0.2554347826086957,
7,0.0,0.0,-0.0855614973262032,-0.0855614973262032,-0.10695187165775401,-0.25133689839572193,0
7,0.0,0.0,-0.09183673469387756,-0.07653061224489796,-0.11224489795918367,-0.23469387755102042
7,0.0,0.0,-0.0861244019138756,-0.07655502392344497,-0.1100478468899521 6,-0.22009569377990432,
7,0.0,0.0,-0.08597285067873303,-0.07239819004524888,-0.10407239819004525,-0.22171945701357465
7,0.0,0.0,-0.09913793103448276,-0.06896551724137931,-0.13362068965517 24,-0.21551724137931033,
7,0.0,0.0,-0.10121457489878542,-0.06072874493927125,-0.13765182186234817,-0.19838056680161945
```

Fig. 5: Training dataset stored as a .csv file

Another file in the repository [3] contains various hand gesture labels with their respective index. The first column in the key points file denotes the gesture labels, while the 2nd and subsequent columns show the coordinates of key points.

By selecting 'K' on the keyboard, the code allows the entry of training data and switches to logging or recording mode. When a user clicks on an index number that corresponds to a certain hand gesture label, a snapshot is taken and the hand landmarks are saved in the training dataset under the corresponding gesture label. This process allows for effective model learning.



(a) Stop (b) Forward (c) Stretch (d) Greeting

(e) Crawl left (f) Crawl right (g) Sit (h) Backward

Fig. 6: Gesture detection using trained model

For this project, eight labels of hand gestures are created as mentioned in Table I. The training data set for all eight labels has been created considering different backgrounds, lighting, and other factors. The model has been successfully trained with the new training data set. Figure 6 shows the outcome obtained after training the model with the new data set.

E. Interfacing Modules

The neural networks, OpenCV library, and robot control commands are combined into a system and executed using the ROS framework.

The robot camera publishes the live image data with a bit of delay which is subscribed to by a subscriber node. This data is then processed and OpenCV methods are used to get a bounding rectangle, landmark list, info text, drawing landmarks, and other required functionality. Then, the input image is cropped to relevant information using a hand-detection neural network. The cropped image is passed through the CNN and hand landmarks are obtained. The hand landmark data obtained is then fed to the feed-forward network for gesture recognition.

The identified gesture with maximum confidence is published onto a topic. The robot controller node then subscribes to this topic and gets the identified gesture. The robot controller is programmed with a delay to prevent rapid and unwanted changes in actions. Hence, the robot controller is programmed with a 3-second delay to read a new message from the gesture topic after the first message is read. The Raspberry Pi is then commanded to execute actions like walking or crawling corresponding to the gesture.

IV. Results

This section presents the outcomes of the proposed system for Gesture-based control of a Quadruped Robot using Neural Networks. The primary goal of the project was to successfully train a neural network model to recognize distinct hand gestures and translate them into control commands for a quadruped robot through the Robot Operating System (ROS).

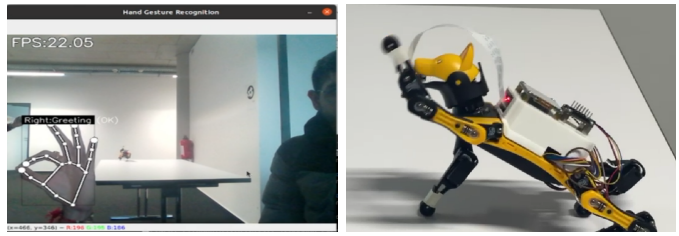
The neural network model has been trained using an extensive number of hand gestures by a unique dataset. Throughout testing, the model performed well and showed an acceptable level of accuracy in identifying different gestures. The training process involved fine-tuning the neural network to capture subtle variations in hand movements, ensuring a reliable gesture recognition system.

Initially, hand gesture data was acquired using a Raspberry Pi camera. However, a lag in reading the data from the Pi camera resulted in processing delays that affected the robot's real-time control. To address this issue, the camera on a remote device was used during the testing phase, to enhance the speed of data processing and reduce the lag.

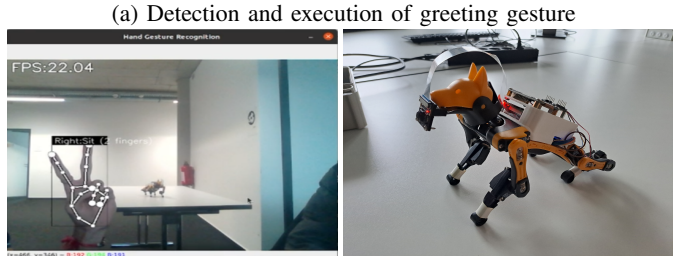
After the successful recognition of hand gestures, the next step was to seamlessly integrate the neural network model with ROS to generate control commands for the quadruped robot. The robot's movements and actions are controlled by the ROS nodes, which are designed to subscribe to the gestures recognized by the neural network and execute a control command to the Raspberry Pi.

The gesture-based control system was implemented successfully, allowing for real-time interaction with the quadruped robot. Various hand gestures stipulated expected control by triggering distinct commands. The robot proved the effectiveness of the combined neural network and ROS-based control framework by reacting promptly to the recognized gestures.

During the experimentation phase, the quadruped robot successfully executed tasks based on recognized gestures confirming the effectiveness of the proposed hand gesture detection neural network model and the ROS-based control system. Figure 7 shows the gesture recognition such as 'OK' and 'Two Fingers' signs, and the corresponding robot actions, such as greeting when the 'OK' gesture is detected and sitting when the 'Two Fingers' gesture is detected.



(a) Detection and execution of greeting gesture



(b) Detection and execution of siting gesture

Fig. 7: Experimental results

V. Conclusion

This research focused on developing a Neural Network-based Gesture Control system for a Quadruped Robot. A Neural Network architecture was used to recognize and react to hand gestures effectively through the application of the MediaPipe framework. The difficulty with the robot's lagging camera feed was overcome by using the camera on the remote system for implementation. A dataset was created during the training phase on which the model was trained. The robot was able to identify and respond to particular hand motions with predetermined behaviors.

The gesture-based control was successfully put into place, enabling real-time communication between the robot and the user. Distinct orders were given by different hand gestures, providing accurate and seamless control. The robot shows the effectiveness of the combined neural network and ROS-based control framework by reacting quickly to the recognized gestures. Even though the research has successfully achieved gesture-controlled operations, real-time applications still need to handle hardware limitations. To improve performance, future iterations might concentrate on enhancing the robot's onboard camera or through approaches mentioned in section VI.

VI. Future Scope

The presented system provides a groundwork for potential future scopes in assistance and security at construction sites to improve output and overall safety.

One possible direction to further work is to implement algorithms that allow the robot to navigate autonomously within the construction site, creating maps and avoiding obstacles

using the sensor data from IMU and camera. This aspect assists with monitoring and optimizing the effectiveness of repetitive duties.

The robot can be equipped with additional sensors for detecting various environmental parameters or data from construction sites. A system can be developed for the robot to log and report data related to construction site activities. The information that the robot provides could be used for project management, safety audits, and performance analytics.

Further, improving the accuracy and efficiency of the neural network architecture and reducing the overall delay in the system from detection to implementation could be interesting domains to work on.

This project's future scope presents exciting opportunities for robotics advancements on construction sites. The specified improvements not only tackle existing issues but also clear the path for a robotic platform that is more intelligent, autonomous, and resilient. This will greatly improve safety, efficiency, and innovation in the construction sector.

Acknowledgement

The authors of this paper are extremely grateful to the IDAC at TUHH for providing the 3D model and technology needed to carry out their work.

References

[1] Rahul Chauhan, Kamal Kumar Ghanshala, and R.C Joshi. "Convolutional Neural Network (CNN) for Image Detection and Recognition". In: *2018 First International Conference on Secure Cyber Computing and Communication (ICSCCC)*. 2018, pp. 278–282. DOI: 10.1109/ICSCCC.2018.8703316.

[2] Moh Harris, Ali Suryaperdana Agoes, et al. "Applying hand gesture recognition for user guide application using MediaPipe". In: *2nd International Seminar of Science and Applied Technology (ISSAT 2021)*. Atlantis Press. 2021, pp. 101–108.

[3] Kazuhito Takahashi. *GitHub: Hand Gesture Recognition*. URL: https://github.com/kinivi/hand-gesture-recognition-mediapipe.

[4] Gongfa Li et al. "Hand gesture recognition based on convolution neural network". In: *Cluster Computing* 22 (2019), pp. 2719–2729.

[5] Hsien-I Lin, Ming-Hsiang Hsu, and Wei-Kai Chen. "Human hand gesture recognition using a convolution neural network". In: *2014 IEEE International Conference on Automation Science and Engineering (CASE)*. IEEE. 2014, pp. 1038–1043.

[6] Elena Nazarova et al. "HyperPalm: DNN-based hand gesture recognition interface for intelligent communication with quadruped robot in 3D space". In: *2022 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE. 2022, pp. 2043–2048.

[7] Vaibhav Mudgal. *Real-Time Gesture Recognition Using GOOGLE'S MediaPipe Hands*. URL: https://mudgalvaibhav.medium.com/real-time-gesture-recognition-using-googles-mediapipe-hands-add-your-own-gestures-tutorial-1-dd7f14169c19.

[8] Jianxin Wu. "Introduction to convolutional neural networks". In: *National Key Lab for Novel Software Technology. Nanjing University. China* 5.23 (2017), p. 495.