https://doi.org/10.31256/Bj3Qo8Y

Vision-based Gesture Tracking for Teleoperating Mobile Manipulators

Tianyu Wang, Yuhui Wan, Christopher Peers, Jingcheng Sun and Chengxu Zhou

***Abstract*—In unconstrained environments, teleoperation exists in consumer-level robots. Motion capture technology has been shown to shrink the skill gap between end user and teleoperation of robots, however, the equipment is getting more expensive and complex such as with motion capture suits. This paper provides a vision-based method with a camera to reduce costs and make teleoperation accessible for consumers. The hand coordinates received from the camera are estimated through Google MediaPipe, a hand posture estimation package. Teleoperation strategies are composed of coordinates and hand gestures. We conduct a simulation study on the Husky robotic platform with a manipulator in PyBullet and demonstrate the control framework through various gestures.**

***Index Terms*—teleoperation, hand gesture recognition, mobile manipulator**

I. Introduction

Although teleoperation has made great progress, the research field for non-professional users is still open to the demand for accurate, fast and cheap solutions. Traditional teleoperation devices such as joystick and keyboard have limitations in the number of degrees of freedom (DOF) they can linearly control simultaneously. In recent years, motion capture technology has been applied to gesture-based robot teleoperation. Compared with full-body gesture estimation using motion capture, hand gesture estimation is a better approach in robot teleoperation as it is more ergonomic for users to show their hands in front of a camera while sitting in a chair. Also, the closer distance to the camera makes detecting much more accurate and resistant to environmental disturbances such as sunlight.

For example, wearable suits which use inertial measurement units (IMU) [1] enable users to operate robots with the movement of their body, making it possible to control more DOFs simultaneously with better accuracy. However, those devices are expensive and are not portable for daily use. A more affordable method is using a camera's visual-based motion capture system.

With the development of artificial intelligence (AI), researchers have proven that real-time human motion capture and posture estimation through a simple camera has gained excellent accuracy and fast computational speed [2], [3]. Therefore, such models may provide an alternative [4] for teleoperation through human body movement. Among them, OpenPose [2] is trained for human body capture for multiple users, with unnecessary computation consumption for single person teleoperation. In teleoperation applications, we only focus on two hands of one single user, and MediaPipe Hand [3] is a more suitable model with a lighter weight and higher computational efficiency.

This paper proposes a hand teleoperation method and demonstrates the control framework through simulation. First, we obtain 21 keypoint coordinates of the hand through MediaPipe Hand and then recognise hand gestures through a three-linear-layer neural network as a classifier. However, there are multiple heuristics-based classifiers, Sung *et al.* [5] concluded that a neural network classifier is more accurate. Then, the teleoperation system is tested on a Husky robot with ViperX 300 (VX300) Robot Arm in PyBullet.

II. System Overview

Our work regards the centre point of the hand as the origin for teleoperation commands, such as the velocity of the Husky robot. Different hand gestures determine different commands to control various modes of the robot. For example, the closed left hand controls the husky, and the right-hand *PointingUp* gesture controls a joint of the VX300 robot arm.

As shown in Fig. 1, our system receives the user's hand picture from an RGB camera and estimates the hand coordinates through MediaPipe Hand [3]. The 2D coordinates are conveyed to the neural network to recognise five different hand gestures, namely *OpenPalm*, *ClosedFist*, *PointingUp*, *FingleGun* and *ThumbUp*, as shown in Table I. The neurons of the three layers are 64, 16, and 16 separately. At last, the hand coordinates and gestures construct teleoperation strategies.

III. Teleoperation Policy

In order to control the mobile manipulator, we implemented a velocity-control method for the Husky mobile base and a position-control method for the robot arm end-effector. We verified the effectiveness of the proposed framework in a physical simulation.

A. Husky Control

When the left hand is recognised as *ClosedFist*, the coordinate of the left-hand centre is recorded. The movement of the Husky is related to the relative position of the current coordinate and the recorded one. Thus, the operator is free to determine the velocity and steering angle by altering the initial position of the hand ($\boldsymbol{H}^0$), in terms of vertical displacement

Authors are with the School of Mechanical Engineering, University of Leeds, UK. c.x.zhou@leeds.ac.uk

This work was supported by the EPSRC [grant numbers EP/R513258/1-2441459, EP/V026801/2] and the AMPI [Innovate UK project number 84646].

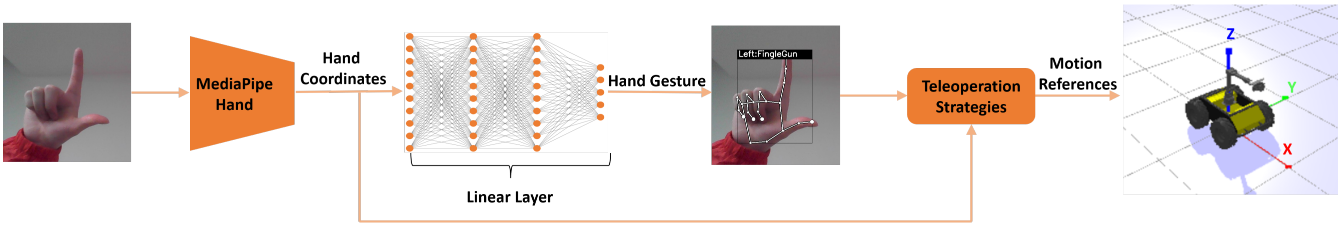


Fig. 1: The structure of the proposed teleoperation system, from visual input to robot motion.

TABLE I: Teleoperation strategies and related commands

Hand	Gesture	Command	Parameter
Left	*ClosedFist*	drive Husky	
Right	*OpenPalm*	open gripper	
Right	*ClosedFist*	close gripper	
Right	*PointingUp*	rotate VX300 around Z axis	θ_1
Right	*FingleGun*	move VX300 in XOZ plane	θ_2, θ_3
Right	*ThumbUp*	rotate VX300 in roll and pitch	θ_4, θ_5

$(H^t_{vertical})$ and horizontal displacement $(H^t_{horizontal})$. The velocity command matrix $(\boldsymbol{V})$, including linear velocity (v) and left-right wheel speed difference (ω), is calculated by the current hand position $(\boldsymbol{H}^t)$ and the gain $(\boldsymbol{K}_v)$,

$$\boldsymbol{V} = \boldsymbol{K}_v(\boldsymbol{H}^t - \boldsymbol{H}^0), \tag{1}$$

where $\boldsymbol{V} = \begin{pmatrix} v \\ \omega \end{pmatrix}$, and $\boldsymbol{H}^t = \begin{pmatrix} H^t_{vertical} \\ H^t_{horizontal} \end{pmatrix}$.

B. Robot Arm Control

As for the 5-DOF VX300 robot arm, we control it with the five different hand gestures as shown in Table I. The 5-DOF from the base to the end-effector are defined from θ_1 to θ_5 in sequence. The X, Y and Z axes are defined in the right part of Fig. 1. The arm motion at time t is represented by

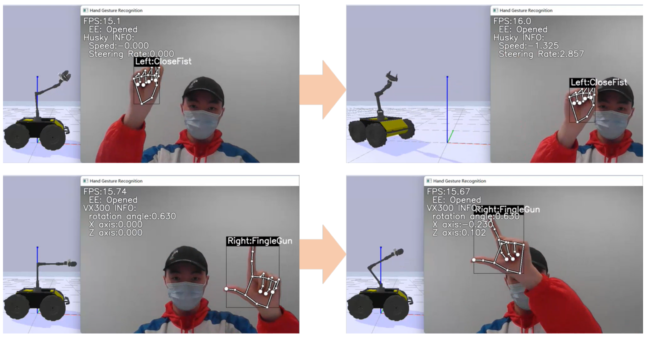
$$\begin{cases} \boldsymbol{P}^t = \boldsymbol{P}^0 + \boldsymbol{K}_p(\boldsymbol{H}^t - \boldsymbol{H}^0) \\ \boldsymbol{q}^t = \boldsymbol{q}^0 + \boldsymbol{K}_\theta(\boldsymbol{H}^t - \boldsymbol{H}^0) \end{cases}, \tag{2}$$

where $\boldsymbol{P} = [P_{horizontal}, P_{vertical}]$ is the end-effector position in XOZ plane and $\boldsymbol{q} = [\theta_1; \theta_4; \theta_5]^T$. Superscripts 0 represent the start time where a gesture is detected. $\boldsymbol{K}_p$ and $\boldsymbol{K}_\theta$ are proportional gains.

The displacement of the end-effector in the XOZ plane is determined by horizontal and vertical displacement of *FingleGun* gesture. Then the corresponding joint angles, θ_2 and θ_3, are calculated through inverse kinematics. The displacements of *PointingUp* and *ThumbUp* control the other three DOFs, namely $\theta_1, \theta_4, \theta_5$. Finally, the gripper is controlled by *ClosedFist* and *OpenPalm* gestures.

IV. Preliminary Tests in Simulation

The simulation (Fig. 2) shows that the user can control the robot effectively to reach various positions. The intuitive idea for *OpenPalm* is to resit the hand position. Hand teleoperation lets users control the robot with simple gestures. Therefore, the user-friendly teleoperation strategies benefit non-professional users outside the lab. Furthermore, our model can achieve approximately 15 FPS in the Windows operating system with Intel Core i7-10870H CPU. At the same time, the computational speed also includes the PyBullet simulation environment, therefore the actual computational speed in the real world without PyBullet simulation will be faster. According to [3], MediaPipe is a light network model which can be deployed on mobile devices such as Android. Thus, it seems possible to develop an application for Android and run the teleoperation model on the user's Android. This will definitely decrease the cost of devices.



```
FPS:15.1
 EE: Opened
Husky INFO:
 Speed:-0.000
 Steering Rate:0.000
```

```
FPS:16.0
 EE: Opened
Husky INFO:
 Speed:-1.325
 Steering Rate:2.857
```

```
FPS:15.74
 EE: Opened
VX300 INFO:
 rotation angle:0.630
 X axis:0.000
 Z axis:0.000
```

```
FPS:15.67
 EE: Opened
VX300 INFO:
 rotation angle:0.630
 X axis:-0.230
 Z axis:0.102
```

Fig. 2: Examples of hand gestures and their related robot movement. (https://youtu.be/y89eR3ZN52w)

V. Conclusion

In summary, our vision-based gesture tracking teleoperation system based on MediaPipe Hand is effective in the simulation environment with a high computational speed. This paper proposes a practical idea to reduce the cost of teleoperation and make it possible for consumers to operate robots daily.

The next step will test the proposed vision-based teleoperation system on a real robot in different operating environments. After that, further work will migrate the system to the Android mobile platform to take advantage of our system's high computational efficiency.

References

[1] C. Peers *et al.*, "Development of a Teleoperative Quadrupedal Manipulator", The 4th UK-RAS Conference, 2021.

[2] Z. Cao *et al.*, "OpenPose: Realtime Multi-Person 2D Pose Estimation Using Part Affinity Fields", IEEE PAMI, 2018.

[3] F. Zhang *et al.*, "MediaPipe Hands: On-device Real-time Hand Tracking", CVPR, 2020.

[4] E. Rolley-Parnell *et al.*, "Bi-Manual Articulated Robot Teleoperation using an External RGB-D Range Sensor", ICARCV, 2018.

[5] G. Sung *et al.*, "On-device Real-time Hand Gesture Recognition", ICCV, 2021.