1er Congrès Annuel
Marseille 7-9 Juin 2023

Teleoperated aerial manipulator and its avatar. Communication, system's interconnection, and virtual world.

Germán Ramírez[1], Thibaut Raharijaona[2], Gerardo Flores[3]

[1] Center for Research in Optics, México
carlosrp@cio.mx
[2] Laboratoire Conception, Fabrication, Commande (LCFC, Metz), France
thibaut.raharijaona@univ-lorraine.fr
[3] Center for Research in Optics, México
gflores@cio.mx

Mots-clés : *Teleoperation, Virtual Reality, Aerial manipulator.*

1 Introduction

Drones are increasingly being used as tools in a wide variety of applications, such as surveillance, and precision agriculture, and in many cases, drones have the potential to reach inaccessible or hazardous locations where human access is limited or risky [1, 2], which is why a drone teleoperation is a valuable tool because it enables operators to remotely control drones over a network connection, ensuring safe and precise navigation. For this reason, several efforts have been conducted for robot teleoperation [3]. However, nowadays technology is not entirely developed to achieve complex tasks autonomously. One of the major problems for teleoperated systems, especially for mobile robots, is the human pilot's visual limited range [4, 5] That is why we propose a human-in-the-loop system combining the use of drones with virtual reality and mapping to accomplish these kinds of tasks. This project consists of the development of a teleoperated drone that can be visualized in Unity with real-time virtual reality and that has a haptic system and a robotic arm as shown in Figure (1).

2 System

2.1 Virtual reality environment

The purpose of designing this environment in Unity is to create a 3D remote teleoperation ground station. It allows virtual reality devices to visualize the environment and get a robot's position feedback in the real world. The aerial manipulator model in Unity contains several apps. These apps work individually and interact to create and send the robot's states depending on the input information. The Unity model's main reference is placed on the UAV body followed by each link of the robotic arm. Each revolute joint represents every degree of freedom that moves individually. The model's main reference contains one app to get the vehicle's position and another app to communicate Unity and Gazebo. Depending on the position input, the

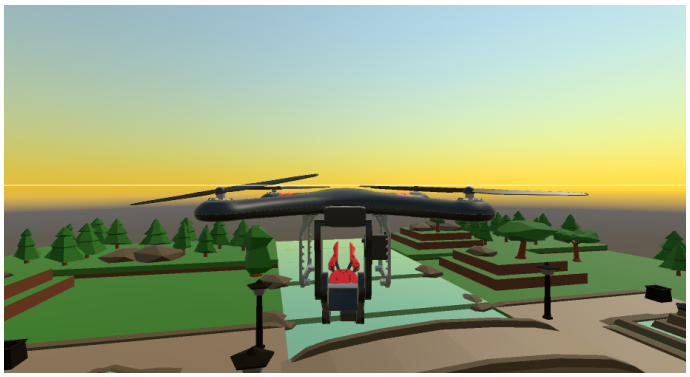

FIG. 1 – Teleoperated UAV in Unity.

first code generates the necessary force to be applied on each rotor to get the desired attitude (ϕ,θ,ψ). The desired attitude leads to the desired position values from Gazebo. This virtual environment in Unity3D is a recreation of the real world where the aerial manipulator is moving.

2.2 Control and software in the loop

Gazebo is a 3D open-source simulator that provides sensors, actuators control, cameras, simulation tools, and realistic dynamics of each model. The purpose of making this second virtual model is to work on a SITL simulation. Gazebo allows testing and simulating the aerial manipulator before being tested in the real world. We program the aerial manipulator's Gazebo model with the PX4 firmware being one of the most used autopilots by the robotics community. PX4 firmware contains packages that integrate the gazebo to perform SITL and facilitate the vehicle's design and control implementation in this case we have used [7]. To communicate with some external software/hardware, a communication protocol to process the messages is needed. In this case, MAVROS (mavlink/ROS) was the best option to extract or send information through the PX4-Gazebo via Python/C++ scripts.

2.3 Communication

MAVROS is a ROS node that allows communication through mavlink protocols containing several topics. Each of them contains specific information about sensors and actuators from the aerial robot. As was previously mentioned, the required topics are the LocalPosition (subscriber) and MountControl (publisher). To get data from LocalPosition a python script was created using rospy and allows linking python with ROS. Then, the geometry message is defined to establish communication with the aerial robot and be published in the servo message. To allow Unity to get the message a ROSBridge protocol is used. This protocol, which is a WebSocket, subscribes to servo messages to get the positions messages via the Internet. To get and read Unity's message, a script is created to communicate with Ubuntu using an IP address. The receiving data is obtained from the servo message using JSON-formatted scripts. Once Unity gets the positions it moves the virtual environment's vehicle to get to the same position as the message. Another python script is created to publish 3 variables for each joint of the robotic arm. Gazebo reads those 3 variables and moves the robotic arm to the desired position. Unity subscribes to a message called data to get the robotic arm information and move the manipulator in the virtual environment. A complete diagram of the communication structure is shown in as show Figure (2).

2.4 Experiments

In this experiment, the aerial robot is teleoperated through the HTC Vive. The task to perform is to take an object and transport it to a given point chosen by the operator. This is

performed in the Gazebo environment while running the control algorithm. The avatar copies the task and the environment. The results demonstrate the control performance and robustness to a mass variation under forces and moments generated by the arm and exerted on the drone. The mass of the object is 160 grams.

A video of the experiments is available in the following link
https ://www.youtube.com/watch ?v=Ipo-vKNvP8k

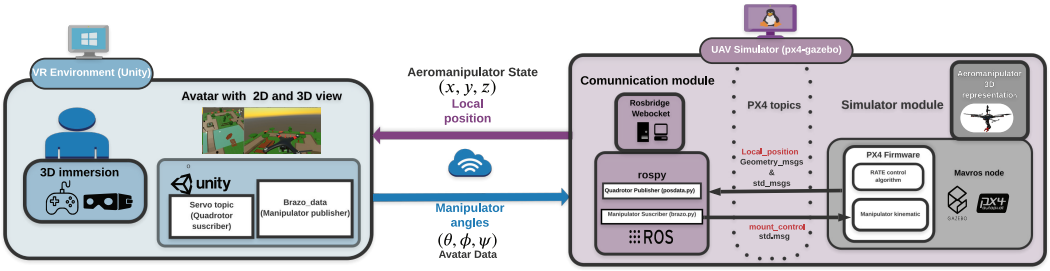

FIG. 2 – Diagram of the system architecture.

3 Conclusions

A teleoperated aerial manipulator was simulated in a virtual environment in Unity 3D and Gazebo. The simulation contains the vehicle dynamics and the kinematics of the manipulator and the control program in the PX4 firmware in Gazebo. Also, it is teleoperated remotely by commands transmitted via ROSBridge protocol (WebSockets). This allowed the VR application to visualize in real-time the states of the aerial manipulator. The time response can be improved using high-performance computer equipment besides considering a higher speed internet and a different communication protocol.

Références

[1] Watkins S, Burry J, Mohamed A, Marino M, Prudden S, Fisher A, Kloet N, Jakobi T and Clothier R (2020). Ten questions concerning the use of drones in urban environments. *Journal Pre-proof.*

[2] J. Escareno, G. Flores, M. Rakotondrabe, H. Romero, R. Lozano, and E. Rubio (2014). Task-based control of a multirotor miniature aerial vehicle having an onboard manipulator. *International Conference on Unmanned Aircraft Systems (ICUAS).*

[3] D. Zhang, G. Yang, and R. P. Khurshid (2020). Haptic teleoperation of UAVs through control barrier functions. *IEEE Transactions on Haptics.*

[4] J. Thomason, P. Ratsamee, J. Orlosky, K. Kiyokawa, T. Mashita, Y. Uranishi, and H. Takemura (2019). A comparison of adaptive view techniques for exploratory 3d drone teleoperation. *ACM Trans. Interact..*

[5] G. Flores, S. Zhou, R. Lozano, and P. Castillo (2013). A vision and GPS-based real-time trajectory planning for MAV in unknown urban environments. *International Conference on Unmanned Aircraft Systems (ICUAS).*

[6] R. Verdín, G. Ramírez, C. Rivera and G. Flores (2021). Teleoperated aerial manipulator and its avatar. Communication, system's interconnection, and virtual world. *International Conference on Unmanned Aircraft Systems (ICUAS).*

[7] G. Flores and R. Lozano (2013). Lyapunov-based controller using singular perturbation theory : An application on a mini-UAV. *American Control Conference.*