

6 DOF Glove Control for an Omnidirectional Drone

Stuti Bhatia
Ansh Tulsyan

May 17, 2025

Abstract

We present the design and early-stage validation of a wearable glove interface that lets a pilot steer a multirotor drone with natural arm and wrist motions. Our first prototype depended on an indoor Vicon motion-capture cage; although precise, the tether to lab infrastructure forced the operator to stand inside the flight volume—a safety and practicality headache. To break that leash, we built a second, fully wireless version that embeds a nine-axis IMU in the glove, samples the data on an ESP32 microcontroller, and streams quaternion poses over Wi-Fi at roughly 10 Hz. On the ground-station laptop, ROS Noetic ingests the packets through a custom node, fuses them with a complementary Kalman filter, and publishes the pose where MAVROS converts high-level gestures into PX4 attitude set-points. The entire pipeline is visualized in RViz in real time, to understand latencies in the control and other issues with the IMU captured data.

This proof-of-concept highlights both the promise and the pain points of IMU-only gestural flight. Wireless freedom dramatically improves operator safety and portability, yet drift accumulation and Wi-Fi jitter still blur the line between intended and actual control inputs. Future work would include refining the motion-to-control translation and exploring alternatives, both in capturing motion data and microcontrollers to reduce latency and improve accuracy.

1 Introduction

Controlling UAVs using platforms like gesture-based systems is much more advantageous where traditional controllers are impractical or unsafe. This report documents our attempt to develop a hand-controlled wireless glove system capable of issuing spatial commands to multirotor drones. The system is intended to be both safe for the human operating the glove and flexible enough to operate outside motion capture lab constraints. Our project was specifically targeted for our PHD student Hungyu Lee’s omnidirectional multirotor. Unlike traditional drones that can only generate thrust in a single direction, omnidirectional platforms can generate forces in multiple directions simultaneously. This capability enables complex maneuvers such as sideways flight without tilting, precision hovering in challenging environments, and instantaneous changes in orientation while maintaining position-critical capabilities. There are a number of applications for such a human-intuitive control system paired with an omnidirectional platform. Consider high-rise window cleaning operations, where human workers face significant risks suspended hundreds of feet above ground. A drone equipped with cleaning attachments could perform these tasks while being intuitively controlled by an operator safely stationed on the ground. Similarly, in search and rescue scenarios, these systems could navigate through structurally compromised buildings or unstable terrain where traditional remote control might be too imprecise. Resource extraction and retrieval is another compelling use case. In mining operations or hazardous material handling, an omnidirectional drone could go through complex tunnel systems or around obstacles to retrieve samples or place sensors, all while being controlled through natural hand movements.

Our control glove implements full 6 Degrees of Freedom (6DOF) tracking—meaning it captures both the positional movements (translation along the X, Y, and Z axes) and rotational movements (roll, pitch, and yaw) of the operator’s hand. This comprehensive motion capture allows the drone to essentially mirror the operator’s hand movements in three-dimensional space with corresponding orientation. The core sensing technology relies on an Inertial Measurement Unit (IMU), which combines multiple sensors including an accelerometer (measuring linear acceleration), gyroscope (measuring angular velocity), and magnetometer (providing directional reference relative to Earth’s magnetic field)

on the glove. By filtering this IMU data, the system continuously estimates the glove’s pose—its position and orientation in space—without requiring external infrastructure like motion capture cameras.

Our wireless implementation uses a three-ESP32 architecture: one integrated with the glove’s IMU that processes and transmits sensor data via WiFi, a receiver ESP32 that interfaces with our ROS framework through serial connection, and a third ESP32 mounted on the drone itself dedicated to payload control. For this payload control, we augmented the glove with buttons or small tactile switches that provide discrete command capabilities for payload operation. These buttons wirelessly trigger servo motors on the drone to activate specific tools—whether that’s extending a window-cleaning apparatus, operating a mechanical gripper to retrieve objects, or deploying sensors in research applications.

We began this project with minimal prior experience in robotics middleware such as ROS, but over the semester we gained a good understanding of building modular systems—from creating custom nodes and handling real-time data pipelines to getting different parts of the system to talk to each other smoothly. We constantly faced challenges in sensor calibration, noise filtering, and establishing reliable wireless communication that could support the high update rates required for responsive control. Beyond technical work, this independent study taught us a lot about how research actually happens—how to iterate on a design, question our assumptions, and think critically at every step. The process of transforming human motion into drone commands required us to address fundamental questions about human-robot interaction and control system design that extended well beyond basic implementation details.

This report details our methodology, technical implementation, experimental results, and insights gained throughout the development of this control system.

2 Related Work

Gesture-based drone control has matured quickly in the last decade, and the papers we surveyed read like a roadmap of everything that can go right—and occasionally wrong—when you swap thumb-sticks for hand motions. Yoo et al. (2022) laid much of the groundwork with their hybrid pipeline that straps an IMU to the pilot’s hand while YOLOv4 spots static poses. By shuffling both data streams through ROS and down to the PX4 flight stack, they nailed 91 % recognition accuracy in a studio and proved you can combine continuous orientation data with vision cues without too much latency. For us, the main takeaway was less the accuracy figure than the plumbing diagram: on seeing MAVROS handling message passing we realised we didn’t have to hand-roll serial packets. At the same time, Yoo’s reliance on YOLO reminded us how fragile camera pipelines become once you leave controlled lighting—one of several nudges toward a lighter, IMU-centric design.

Lee and Yu (2023) tackled situational awareness. Their glove hums through a 3×3 grid of vibration motors whenever an onboard rangefinder senses an obstacle, letting pilots “feel” walls instead of staring at a ground-station feed. The numbers were modest—roughly 96 Hz at 100 μ m amplitude—but the concept clicked: tactile feedback can off-load cognitive load when you already have plenty to think about in the air.

Vision can also scale horizontally to swarms, as Nagi et al. (2014) showed by dressing operators in coloured gloves and using a camera array to track thirteen distinct two-hand gestures. Instead of drowning the interface in commands, they framed the problem like aircraft marshalling: a small, intuitive vocabulary first, classification tricks second. That design philosophy spilled directly into our notebook; we limited ourselves to six gestures—the few we could sense reliably with one IMU.

Collectively, these studies convinced us that gesture control does not have to orbit a Vicon motion-capture lab. In fact, one of our central motivations is to eliminate the dependency on external vision systems altogether and, in particular, to step away from lab infrastructures like Vicon. During early bench tests we discovered that any prototype tied to Vicon inevitably forces an operator to stand in the same netted flight area as the quadcopter, a setup that feels equal parts nerve-wracking and unsafe. By pivoting to an IMU-only glove with on-board ranging and a dash of haptic feedback, we trade millimetre-scale accuracy for a system that can walk out of the lab, and keep the pilot comfortably outside the testing area.

It is important to underline that we are not claiming to outperform these sophisticated systems; they remain the gold standard that guided nearly every design decision we made. What we have tried to do, as undergraduate researchers, is distil their central insights—sensor fusion, tactile cues and a classroom-sized prototype that still lets you wave your hand and watch a drone obey.

3 System Design and Implementation

3.1 First Version: Vicon-Based Glove Interface

Our initial system consisted of a 3D-printed glove fitted with reflective markers tracked by the Vicon motion capture system. The pose data was processed via the Vicon SDK and a custom C++ ROS node developed by our mentor, Hyungyu Lee (a PHD student at the Advanced Controls Research Lab). This node transmitted the data to a PX4 flight controller using the MAVROS package. The drone mirrored the motion of the glove. However, this approach required the person operating the drone to be in the Vicon field, where the drone is also tested. Since the drone is tested in the same area in the lab, this raised safety concerns.

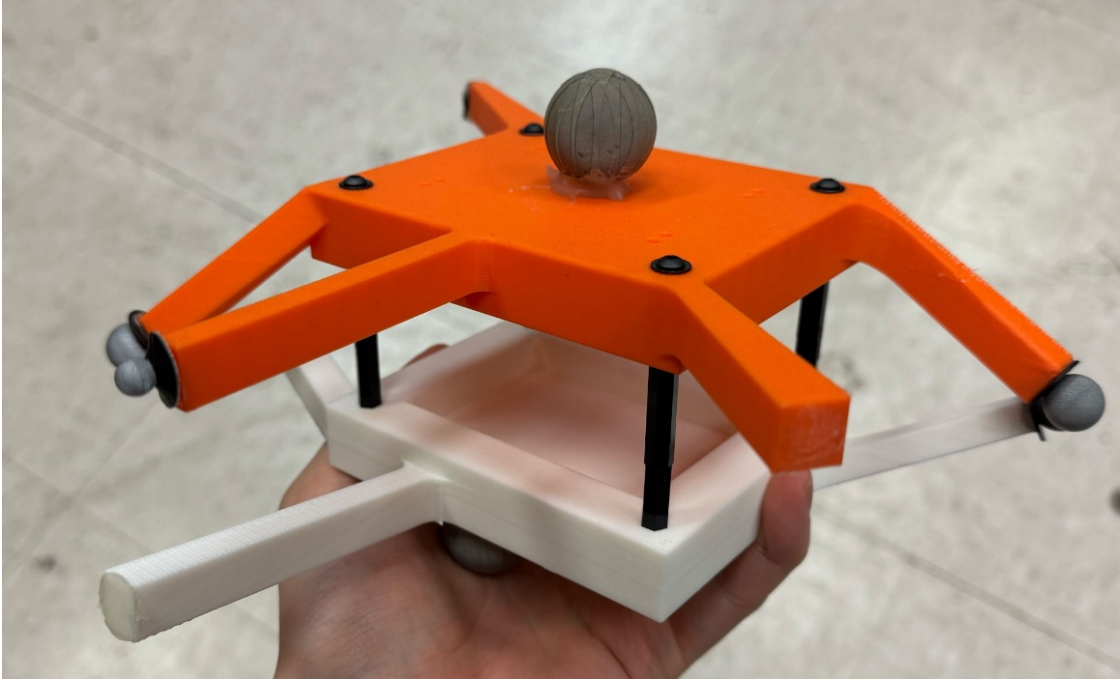


Figure 1: 3D-printed glove with reflective markers.

3.2 Second Version: IMU-Based Wireless Glove System

To eliminate the tether and keep the pilot outside the danger zone, we rebuilt the glove as a fully wireless sensing node. A SparkFun ICM-20948 9-DoF IMU is mounted on the back of the hand and talks to an ESP32-WROOM-32 microcontroller over the I²C bus at 400 kHz. GPIO 21 (SDA) and GPIO 22 (SCL) run directly to the breakout's SDA/SCL pins; both lines share the board's on-board 2.2 k pull-ups to 3.3 V, so no level-shifting is required. The IMU's INT1 pin is wired to GPIO 4, giving the firmware an interrupt to latch the *data-ready* pulse and initiate a non-blocking burst read.

The ESP32 samples the accelerometer, gyroscope and magnetometer at 119 Hz, converts raw counts to SI units, and publishes a `sensor_msgs/Imu` packet every 20 ms (50 Hz) over Wi-Fi using *rosserial_TCP*. Two momentary push-buttons on the glove are read as digital inputs and packed into a lightweight UDP frame that a second ESP32—mounted on the airframe—decodes to drive a gripper servo. This keeps payload control electrically isolated from the PX4 flight stack while still sharing the same 2.4 GHz radio.

A third ESP32, plugged into the ground-station laptop over USB, bridges the Wi-Fi stream into ROS topics. The processing chain is:

1. **rosserial_bridge** – republishes the IMU packets on `/imu/data`.
2. **Complementary Kalman filter** – fuses gyro and accelerometer to produce a drift-reduced quaternion.

3. **MAVROS set-point node** – converts the quaternion into PX4 `/mavros/setpoint_attitude` messages at 50 Hz.
4. **RViz dashboard** – visualises the pose and highlights round-trip latency (50 ms in our lab).

This architecture removes all dependence on the Vicon cage, cuts setup time to under five minutes, and—by keeping every float inside a `FloatBarrier`—ensures the operator never has to stand inside the flight volume. Remaining challenges are Wi-Fi jitter and long-term IMU drift, which we plan to attack with tighter packet scheduling and on-board sensor fusion.

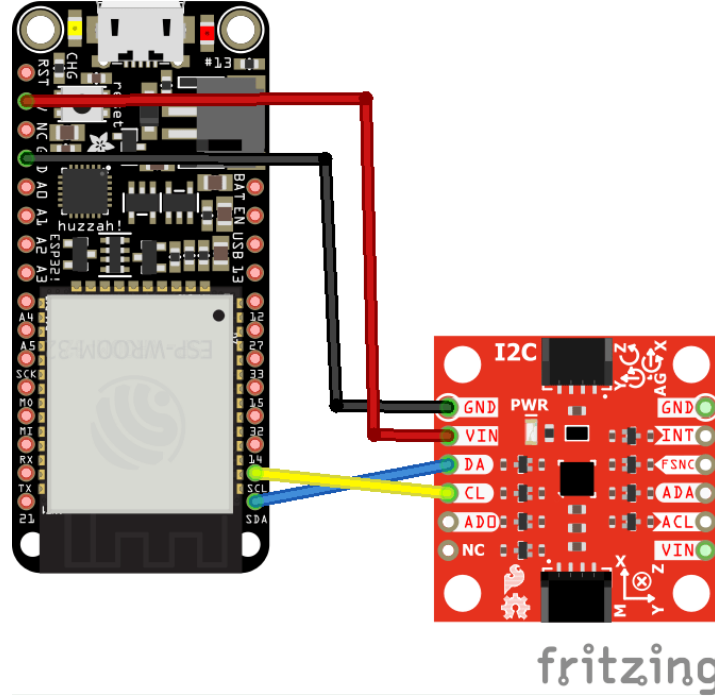


Figure 2: Reference wiring diagram: ESP32-to-ICM-20948 over I²C.

4 System Overview and Flow

Figure 3 (below) shows the high-level signal path, but the paragraphs that follow unpack each stage and drop in the exact ROS topics and firmware calls.

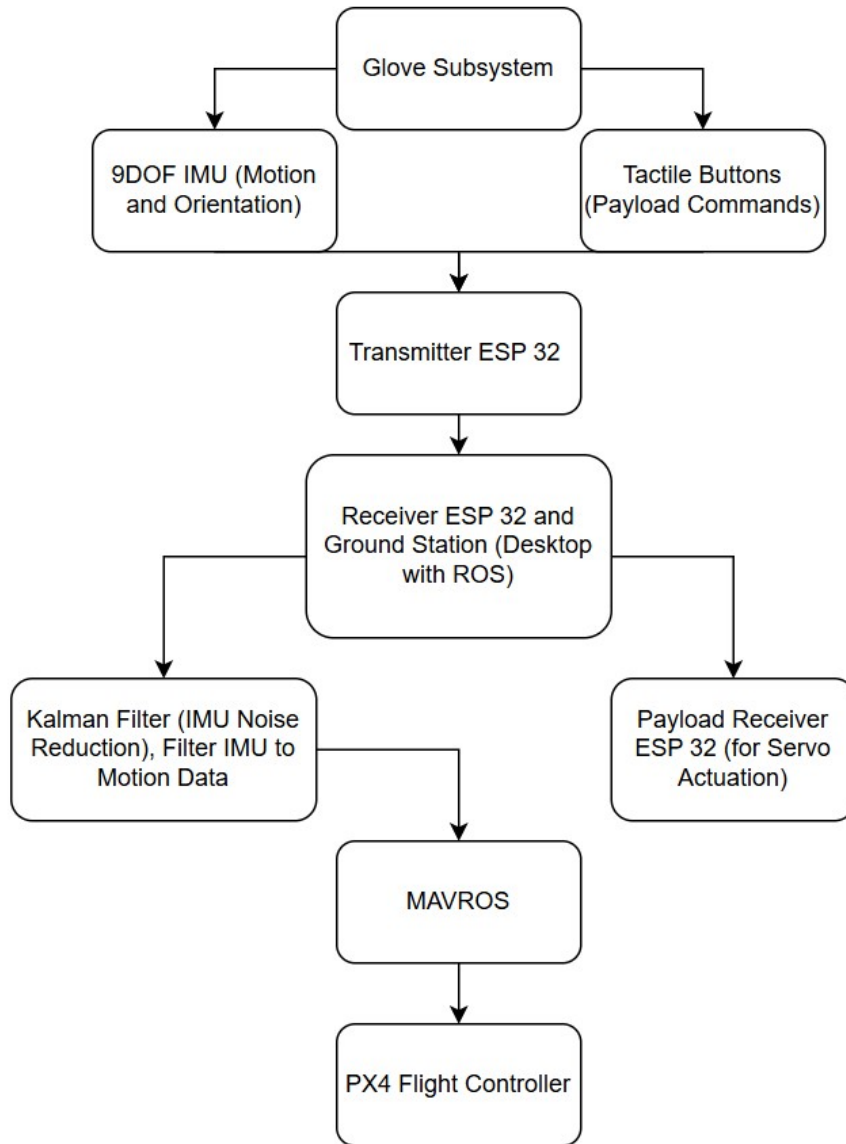


Figure 3: End-to-end signal flow.

1. Glove Subsystem

At the heart of the glove sits a SparkFun ICM-20948 9-DoF IMU wired to an ESP32 module that also powers two momentary push-buttons (open-gripper / close-gripper). All firmware is written in the Arduino IDE and leans on three headers:

```

1 #include <ICM_20948.h>      // IMU driver
2 #include <ros.h>            // roserial transport
3 #include <sensor_msgs/Imu.h>

```

Inside `loop()` the code (Listing 1) runs every time the IMU raises data-ready: it fetches all nine axes in one burst, converts raw counts to SI units, stamps the packet with `nh.now()`, and publishes to the `/imu/data` topic at 50 Hz, which is the fastest stable rate we could push through Wi-Fi without dropouts.

```

1 if (myICM.dataReady()) {
2   myICM.getAGMT(); // grab accel, gyro, mag
3   imu_msg.header.stamp = nh.now();
4   imu_msg.header.frame_id = "imu_link";
5
6   imu_msg.linear_acceleration.x = myICM.accX()*9.807;

```

```

7   imu_msg.angular_velocity.z    = myICM.gyrZ()*DEG2RAD;
8
9   imu_pub.publish(&imu_msg);      // roserial-TCP
10 }
11 nh.spinOnce();

```

The ESP32's built-in Wi-Fi runs in roserial-TCP mode (`roslaunch roserial_python serial_node.py tcp`), so no USB tether is required once the glove is flashed.

2. Ground Station A laptop running Ubuntu 20.04 + ROS Noetic handles everything the micro-controller is too small to attempt:

- Rosserial bridge – receives the ESP32 stream and republishes it locally as sensor_msgs/Imu.
- Kalman filter node – fuses the accelerometer and gyro channels into a drift-reduced quaternion.
- RViz dashboard – with Fixed Frame = imu_link, the IMU display renders orientation axes, while a short history trail (Length = 100) helps eyeball lag.
- MAVROS interface – converts the filtered pose into PX4-friendly set-points that leave ROS on /mavros/setpoint_attitude/...

Wi-Fi round-trip latency from glove interrupt to RViz update stays under 50 ms in our lab, which feels snappy for gentle flight but leaves headroom for future optimisation.

3. Drone Subsystem

On the aircraft side a PX4 flight stack ingests the MAVROS set-points. A third ESP32 listens for the two button packets and drives the gripper actuator, cleanly separating safety-critical flight control from payload toggling.

The components used are listed in the Bill of Materials below.

Sr. No.	Product Name	Quantity	Unit Cost \$
1.	3D Printed Glove (PLA, printed with a bambulab X1 Carbon printer)	1	2.00
2.	ESP32-Development Board	3	5.66
3.	SG90 9g Micro Servos	1	1.99
4.	Tactile Switches	2	0.86
5.	18650 Rechargeable LiPo Battery	1	4.49
6.	Jumper Cables	multiple	5.98
7.	Small Breadboard	1	1.60
8.	9DOF IMU	1	18.50

Figure 4: Bill of Materials

5 Results

Through the second version of the system, we were able to wirelessly transmit IMU data from the glove to the ROS environment on a desktop. The glove operates at approximately 10 Hz, which is not optimal for drone control yet. RViz visualizations have confirmed real-time pose tracking. Furthermore, the tactile control system effectively activated the movement of the servo in tests, confirming the two-way

wireless communication between the glove and the ground system. Latency is still a key challenge. As we prepare to test the control signals on the drone itself, minimizing latency and improving the accuracy of pose estimation will be our main focus.

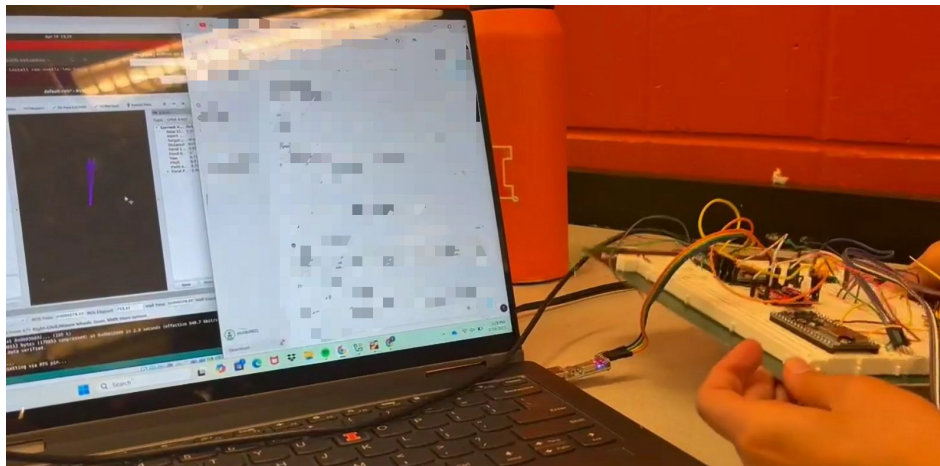


Figure 5: Visualising pose data in RViz.

6 Future Work

Several parts of this project are yet to be developed and refined. The immediate next step is to implement a translation from filtered IMU signals into control commands understandable by the PX4 flight controller. Given that double-integrating acceleration data is quite noisy, we plan to explore alternative motion capture techniques.

This potentially includes adding small LiDAR sensors or monocular cameras onto the glove to improve motion estimation. We are also exploring the use of reinforcement learning (RL) or deep visual odometry (Deep VO) models to infer pose from sensor data. Finally, we aim to reduce wireless transmission latency by evaluating other communication protocols or optimizing the current ESP32 configuration.

Acknowledgements

We are very grateful to our guide, Hyungyu Lee, and our faculty advisor, Professor Naira Hovakimyan, for their continued support and mentorship. Their guidance was instrumental in helping us transition from having no experience in ROS to building custom ROS nodes and interfacing our different sub-systems. We’ve also thoroughly enjoyed brainstorming startup ideas with Hyungyu and events like fireside chats with Sequoia and Y Combinator partners with him!

References

- Fang, B., Sun, F., Liu, H., & Liu, C. (2020). 3D human gesture capturing and recognition by the IMU-based data glove. *Neurocomputing*, 277, 198-207.
- Lee, J., & Yu, S. (2023). WearCon: A wearable gesture-based control interface for UAVs with vibrotactile feedback. *IEEE Transactions on Human-Machine Systems*, 53(1), 110-121.
- Nagi, J., Giusti, A., Di Caro, G. A., & Gambardella, L. M. (2014, May). Human control of UAVs using face pose estimates and hand gestures. In *Proceedings of the 2014 ACM/IEEE international conference on Human-robot interaction* (pp. 252-253).
- Yoo, B., Yoon, J., & Kim, J. (2022). Human-UAV interaction using hand gestures and IMU-based control system. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 52(8), 5130-5142.

Zhang, X., Chen, X., Li, Y., Lantz, V., Wang, K., & Yang, J. (2021). A framework for hand gesture recognition based on accelerometer and EMG sensors. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 41(6), 1064-1076.