DOI 10.34185/1562-9945-5-161-2025-03
UDC 004.358

S.V. Sidletskyi, A.A. Demchyshyn, O.S. Kaleniuk, A.G. Donets

CYBER-GLOVE-BASED HUMAN-COMPUTER INTERACTION SYSTEM WITH FINGER FLEXION SENSING

Abstract. The potential for advancing augmented reality technologies is closely tied to the improvement of wearable input-output devices. The work presents a hardware-software system for human-computer interaction in the form of a cyber-glove. The system includes a manipulator developed using a wearable microcontroller board equipped with orientation and finger bending sensors, microcontroller firmware, a server application that reads and processes data using the Madgwick filter, and a client application that visualizes command execution. The use of the filter for processing sensor data enables stable determination of the manipulator's spatial orientation, while finger bending degree provides gesture-based commands. The proposed architecture and application programming interface allow the manipulator to be used for remote control tasks.

Keywords: cyber glove, human-computer interaction, virtual reality, Madgwick filter, augmented reality.

Problem Statement. The increasing precision of data provided by microelectromechanical systems (MEMS) [1], combined with advances in bidirectional human-computer communication [2], has facilitated the emergence of new forms of human perception and experience – namely, virtual and augmented reality [3]. At the same time, the development of communication channels has enabled the use of manipulator devices for real-time control of remote robotic systems.

Today, the potential for advancing augmented reality technologies is closely tied to the improvement of wearable input-output devices. Image latency during rapid manipulator movement and unintuitive control schemes remain key challenges that hinder synchronization between the perceived and expected visual output, ultimately degrading immersion in the mixed reality continuum [4, 5]. For instance, a computer mouse, as an input device, captures motion only within a plane, while virtual reality (VR) system controllers incorporate buttons or triggers that are unnatural to human interaction. Such devices not only limit the ability to fully leverage intuitive hand and finger gestures but also necessitate additional layers of abstraction during the design of graphical user interfaces.

Analysis of recent research and publications. Among VR controllers, devices that use cameras (e.g., Oculus Rift [6]) or beacons (Lighthouse systems, such as HTC Vive [7]) for spatial tracking are in common use. The operation of such controllers requires fixed base stations equipped with cameras or optical systems. The latter have an advantage because these stations do not require a direct line of communication with the personal computer and do not limit the number of sensors operating simultaneously. The control elements of these controllers mostly include buttons, triggers, analog sticks, and touchpads.

Regarding cyber gloves, existing devices do not require the installation of external tracking stations, which significantly simplifies the setup of such manipulators at the workspace. Cyber gloves can be connected either via cable or via wireless protocol. In this case, hand orientation in space is determined by a built-in inertial measurement unit (IMU), while additional sensors track finger movements, as implemented in CaptoGlove [8]. There are also more advanced models that provide haptic feedback through vibration motors or even include exoskeletons to restrict finger movement, such as CyberTouch and CyberGrasp from CyberGlove Systems LLC. Overall, cyber glove-based solutions are significantly more expensive than VR controllers, but their applications extend beyond VR systems. Manufacturers claim these devices are used for animation, modeling, biomechanics, robotics, and medical fields [9].

To determine the orientation of an object in space, the readings from inertial measurement sensors are processed using a filtering algorithm. The traditional approach employs the Kalman filter, which has a rather complex implementation and requires a large number of computations per update cycle, making its use infeasible on portable low-performance systems. Alternative filters based on quaternions have also been developed specifically to reduce computational load on portable systems [10]. For example, the Mahony filter [11] first estimates orientation from gyroscope readings and then corrects it using magnetometer data.

The Sebastian Madgwick filter is one of the newest methods for calculating spatial orientation based on accelerometer, gyroscope, and magnetometer readings (the magnetometer is optional) [12]. Applications based on this algorithm process data significantly faster than comparable methods, which is why it is often used in software development for quadcopters, as they need to respond quickly to changes in rotation angle while minimizing energy consumption. The variant of the filter that uses a magnetometer is more popular because it compensates for magnetic distortions and gyroscope bias. The filter represents the rotation angle using quaternions. In addition to sensor noise, one of the main challenges in calculating orientation based on gyroscope and magnetometer data is zero drift and magnetic distortion. The Madgwick filter compensates for these anomalies.

Research objective. To develop and validate a cyber-glove-based system for the input of orientation commands and state control of an object based on integrated finger bend sensing.

Main research content. *Sensor Data Fusion Filter.* The Madgwick algorithm was implemented in JavaScript as a server-side module according to the following approach.

The core of the Madgwick filter is the determination of orientation based on angular velocity and vector observations. Since the gyroscope measures angular velocity, the values obtained relative to the three axes can be used to define the vector (1), whose derivative describes the rate of change of the object's orientation relative to the Earth (2).

$$s_{\omega} = \lfloor 0 \quad \omega_x \quad \omega_y \quad \omega_z \rfloor . \tag{1}$$

$$ {}^{S}_{E}\dot{q}=\frac{1}{2}{}^{S}_{E}\otimes{}^{S}\omega. \tag{2}$$

As a result of integrating the quaternion derivative ${}^{S}_{E}\dot{q}_{\omega,t}$, the orientation of the Earth relative to the object at time *t* (3) is obtained. Provided the initial conditions are known, the orientation values can be determined as (4).

$$ {}^{S}_{E}\dot{q}_{\omega,t}=\frac{1}{2}{}^{S}_{E}\widehat{q}_{est,t-1}\otimes{}^{S}\omega_{t}, \tag{3}$$

where ${}^{S}\omega_{t}$ – angular velocity measured at time *t*;

${}^{S}_{E}\widehat{q}_{est,t-1}$ – previous orientation estimate.

$$\begin{gathered} {}^{S}_{E}q_{\omega,t}=\frac{1}{2}{}^{S}_{E}\hat{q}_{est,t-1}+{}^{S}_{E}\dot{q}_{\omega,t}\Delta t, \\ {}^{S}_{E}q_{\omega,t}=\frac{1}{2}{}^{S}_{E}\widehat{q}_{est,t-1}+{}^{S}_{E}\dot{q}_{\omega,t}\Delta t. \end{gathered} \tag{4}$$

When determining orientation based on vector observations, the filter utilizes quaternion representation of orientation, as it helps to find a unique orientation value, unlike Euler angle–based solutions, which can yield infinitely many solutions. The sensor's orientation ${}^{S}_{E}\widehat{q}$ is considered as the value that aligns a predefined field direction in the Earth's coordinate system ${}^{E}\widehat{d}$. An optimization problem is formulated to represent ${}^{S}_{E}\widehat{q}$ as the solution of equation (5), where (6) defines the objective function.

$$\min_{{}^{S}_{E}\widehat{q}\in\Re^{4}} f({}^{S}_{E}\widehat{q},{}^{E}\widehat{d},{}^{S}\widehat{s}). \tag{5}$$

$$f({}^{S}_{E}\widehat{q},{}^{E}\widehat{d},{}^{S}\widehat{s})={}^{S}_{E}\widehat{q}^{*}\otimes{}^{E}\widehat{d}\otimes{}^{S}_{E}\widehat{q}-{}^{S}\widehat{s}, \tag{6}$$

where ${}^{S}\widehat{s}$ – is the measured field direction in the sensor's coordinate system.

Gradient descent method is selected as the optimization algorithm. Based on the initial value ${}^{S}_{E}\widehat{q}_{0}$ and the step size μ, the orientation value can be computed over n iterations of (7).

$$ {}^{S}_{E}q_{k+1}={}^{S}_{E}\widehat{q}_{k}-\mu\frac{\nabla f({}^{S}_{E}\widehat{q}_{k},{}^{E}\widehat{d},{}^{S}\widehat{s})}{\left\|\nabla f({}^{S}_{E}\widehat{q}_{k},{}^{E}\widehat{d},{}^{S}\widehat{s})\right\|}, k=0,1,2...n. \tag{7}$$

Gradient descent algorithm ensured high performance at low sampling rates. As a rough estimate, each filter update involved 277 minimization iterations (or 109 if magnetometer data was unavailable), which was significantly less than what was typically required by an extended Kalman filter.

Hardware Implementation. The hardware development of the interaction system involved the MPU-9250 motion-processing unit (MPU) under the control of a SAM D21 microcontroller featuring a single core with a 32-bit Armv6-M architecture. The MPU integrated three spatial sensors – an accelerometer, a gyroscope, and a magnetometer – providing sufficient data to determine the object's spatial orientation with defined accuracy [13]. The board interface included analog-to-digital converter inputs, which were used to connect finger bend sensors based on resistive elements. The glove (Fig.1) was connected to the computer via a USB 2.0 cable. Compared to wireless connections, this solution minimized signal latency, supplied power to a wearable device, and was justified in conditions where radio-electronic countermeasures are active. The cable ran under the shirt sleeve and did not restrict movement; the limited range imposed by the cable did not pose difficulties, considering the tasks and configuration of the hardware-software system.

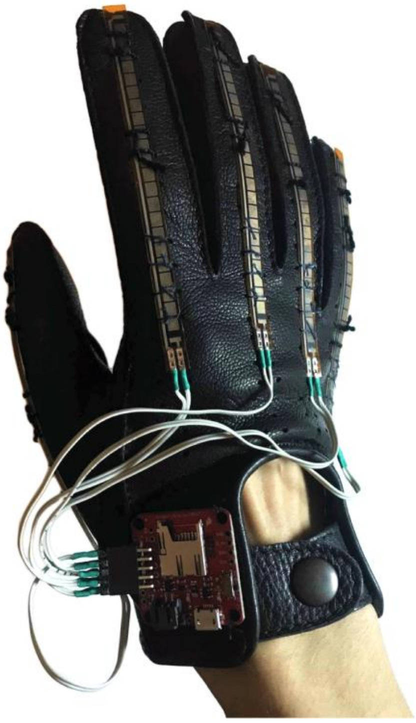

Figure 1 – A low-cost cyber-glove with finger bend monitoring

Software Implementation. The client–server architecture of the software system, illustrated in Fig. 2, enables the glove to be used both on a single PC as well as on remote computers.

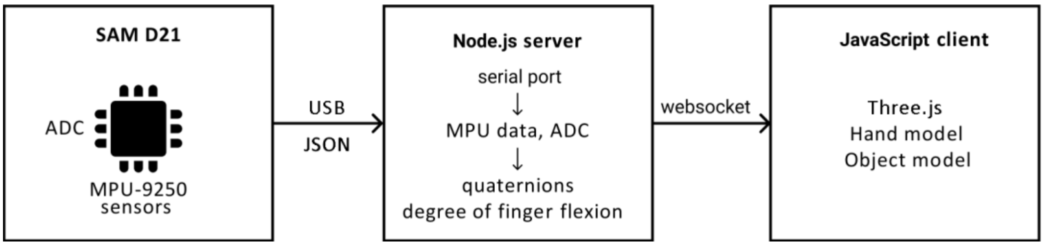

Figure 2 – Data flow diagram in human-computer interaction system

Wearable device firmware implementation. The firmware for the SAM D21 microcontroller was developed in C++. The main functionality of the firmware routines is to read data from the sensors and transmit it via the USB hardware port. Data is transferred in JSON format – a text-based data interchange format derived from JavaScript, and supported by many other programming languages. Using JSON allows for an intuitively readable data structure and enables data validation, as a parsing module will return an error if the string is corrupted.

Server implementation. Within the system, the personal computer acted as the master device where all computations were performed, including the Madgwick filter. Unlike the typical approach, where filtering is done directly on the microcontroller, data processing was entirely moved to the server side that ran on the computer.

Initialization of the cyber-glove began with starting the calibration procedure on the server, implemented in Node.js. During this process, the boundary values for each resistive sensor were determined – corresponding to the fully extended and fully bent positions of the finger. The obtained data were then used for normalization, which was necessary for accurately calculating the degree of finger flexion.

Hand modeling. The hand model was created in Blender software using a rig for skeletal animation (Fig. 3). This approach enabled easy adjustments to the appearance and animation of model elements by modifying the settings of corresponding skeleton components.

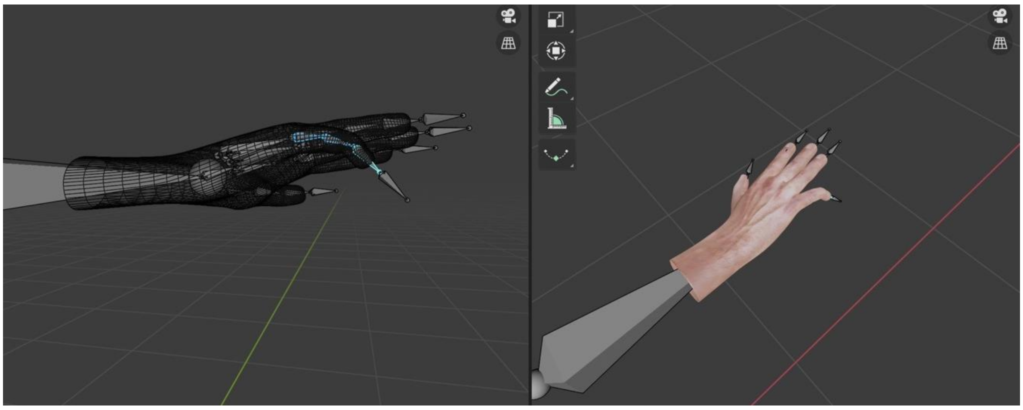

Figure 3 – 3D model of a hand with bone configuration

Client implementation. The client application was implemented using the JavaScript programming language and the Three.js library. The 3D model formats selected for use were FBX and GLTF, which enabled access to the built-in animation module functionality of Three.js.

The Three.js animation module allows for the manipulation of the model's bones (provided that a rig has been configured and bound to the polygonal mesh); sets of vertex positions for automatic interpolation between them; material properties, including color and transparency; visibility; position and rotation.

Skeletal animation, which employs rigging (the process of creating a skeleton structure), enables manipulation of a large number of a model's components using a relatively small set of control elements – namely, the skeleton's bones. Due to the hierarchical structure of the bones, the offset of each dependent bone is influenced not only by its own transformation, but also by the transformations of all its parent bones and joints.

Each Three.js object possesses two transformation matrices: a local matrix, which contains the position, rotation, and scale values relative to its parent, and a global matrix, which represents the object's absolute position in the scene.

To modify the wrist's rotation, a transformation matrix was used, while the visualization of finger bending was achieved through appropriate skeleton configuration of the 3D model (Fig. 4).

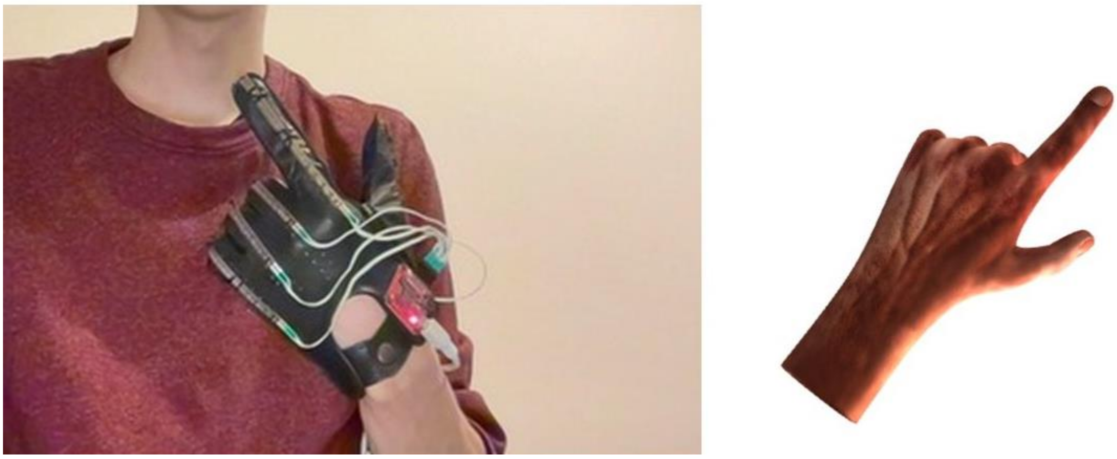

Figure 4 – Visualization of the cyber glove’s orientation along with hand virtual model (avatar)

The client implementation featured several visualization modes for the controller. In one of the modes, the user could solve a Rubik’s Cube using hand gestures. A quick inward twitch of a finger triggered the rotation of the corresponding face.

Conclusions. The work presented a hardware-software command input system for human-computer interaction, which included a physical manipulator, firmware for the manipulator’s microcontroller, a server application that read and processed data using a sensor fusion filter, and a client-side web application that visualized command execution. The use of a sensor fusion filter provided a stable determination of the manipulator’s spatial orientation and finger flexion, with an average delay of 36 ms.

The developed application programming interface made it possible to use the manipulator for remote control tasks with visualization on any device with an Internet connection and WebGL support – such as a smartphone.

REFERENCES

1. Abrash M. What VR could, should, and almost certainly will be within two years. URL: http://media.steampowered.com/apps/abrashblog/Abrash%20Dev%20Days%202014.pdf (date of access: 11.06.2025).

2. Senturia S. D. Microsystem Design. Springer New York, NY, 2000. 689 p.

3. Mine M.R. Virtual Environment Interaction Techniques. TR95-018. University of NC United States, 1995. 18p.

4. Sherman W.R., Craig A.B. Understanding Virtual Reality: Interface, Application, and Design. Morgan Kaufmann Publishers, 2018. 582 p.

5. LaValle S.M. Virtual reality. Cambridge University Press, 2023. 426 p.

6. Davis B., Bryla K., Benton P.A. Oculus Rift in Action. Manning, 2015. 440 p.

7. Buckley S. This Is How Valve's Amazing Lighthouse Tracking Technology Works. URL: https://gizmodo.com/this-is-how-valve-s-amazing-lighthouse-tracking-technol-1705356768 (date of access: 11.06.2025).

8. CaptoGlove. Wearable Gaming, VR, AR, PC & Mobile Controller. CaptoGlove LLC. URL: https://www.kickstarter.com/projects/146300353/

captoglove (date of access: 11.06.2025).

9. Sturman D. A survey of glove-based input. Zeltzer, 1994. 30 p.

10. Cocoli K., Badia L. A Comparative Analysis of Sensor Fusion Algorithms for Miniature IMU Measurements. Dept. of Information Engineering (DEI), University of Padova, Italy Conference: 24th International Seminar on Intelligent Technology and Its Applications (ISITIA): Indonesia, 2023. 6p.

11. Nitin J. S. Mathematical Model of an IMU. URL: https://nitinjsanket.github.io/tutorials/attitudeest/mahony.html (date of access: 11.06.2025).

12. Madgwick S. An efficient orientation filter for inertial and inertial/ magnetic sensor arrays. URL: https://www.samba.org/tridge/UAV/madgwick_internal_report.pdf (date of access: 11.06.2025).

13. 9DoF Razor IMU M0 Hookup Guide. URL: https://learn.sparkfun.com/tutorials/9dof-razor-imu-m0-hookup-guide (date of access: 11.06.2025).

Received 11.08.2025.
Accepted 13.08.2025.

Комплекс людино-машинної взаємодії на основі кібер-рукавички з реєстрацією згину пальців

Потенціал розвитку технологій доповненої реальності тісно пов'язаний із вдосконаленням носимих пристроїв введення-виведення. Метою дослідження є розробка та валідація програмно-апаратного комплексу на основі кібер-рукавички для введення команд орієнтації та керування станом об'єкта на основі інтегрованого вимірювання згину пальців. Комплекс включає маніпулятор, розроблений на основі носимої плати мікроконтролера, оснащеної датчиками орієнтації та згину пальців, прошивку мікроконтролера, серверний застосунок для зчитування та обробки даних з використанням фільтра Маджвіка, а також клієнтський застосунок для візуалізації виконання команд. Використання фільтра для обробки даних датчиків забезпечує стабільне визначення просторової орієнтації маніпулятора, а ступінь згину пальців — подачу команд на основі жестів. Запропонована архітектура та інтерфейс програмування застосунків дозволяють використовувати маніпулятор для задач дистанційного керування.

Ключові слова: кіберрукавичка, взаємодія людини з комп'ютером, віртуальна реальність, фільтр Меджвіка, доповнена реальність.

Сідлецький Станіслав Валентинович – магістр кафедри автоматизації проектування енергетичних процесів та систем, Національний технічний університет України "Київський політехнічний інститут імені Ігоря Сікорського", ORCID: https://orcid.org/0009-0009-8516-8665

Демчишин Анатолій Анатолійович – к.т.н., доцент кафедри цифрових технологій в енергетиці, Національний технічний університет України Київський політехнічний інститут імені Ігоря Сікорського", ORCID: https://orcid.org/0000-0001-7754-0185

Каленюк Олександр Сергійович – к.т.н., старший викладач кафедри цифрових технологій в енергетиці, Національний технічний університет України "Київський політехнічний інститут імені Ігоря Сікорського", ORCID: https://orcid.org/0009-0009-3141-4840

Донець Андрій Георгійович – к.ф.-м.н., доцент кафедри цифрових технологій в енергетиці, Національний технічний університет України "Київський політехнічний інститут імені Ігоря Сікорського", ORCID: https://orcid.org/0000-0002-8122-051X

Sidletskyi Stanislav Valentynovych – graduate student of the department of automation of design of energy processes and systems, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute” , ORCID: https://orcid.org/0009-0009-8516-8665

Demchyshyn Anatoliy Anatoliyovych – candidate of engineering sciences, assistant professor, department of digital technologies in energy, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute” , ORCID: https://orcid.org/0000-0001-7754-0185

Kaleniuk Oleksandr Sergiyovych - candidate of physical and mathematical sciences, assistant professor, department of digital technologies in energy, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”, ORCID: https://orcid.org/0009-0009-3141-4840

Donets Andrii Georgievich – candidate of physical and mathematical sciences, assistant professor, department of digital technologies in energy, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”, ORCID: https://orcid.org/0000-0002-8122-051X